Bayesian Environment Representation, Prediction, and Criticality Assessment for Driver Assistance Systems

Vom Fachbereich
Elektrotechnik und Informationstechnik
der Technischen Universität Darmstadt
zur Erlangung des akademischen Grades
eines Doktor-Ingenieurs (Dr.-Ing.)
genehmigte Dissertation

von

Matthias Schreier, M.Sc.

geboren am 30. Dezember 1985 in Schweinfurt

Referent:	Prof. Dr.-Ing. J. Adamy
Korreferent:	Prof. Dr. rer. nat. H. Winner
Tag der Einreichung:	21. September 2015
Tag der mündlichen Prüfung:	10. Dezember 2015

D17
Darmstadt 2015

Preface

This dissertation is the result of my work at the Control Methods and Robotics Lab of the Institute of Automatic Control and Mechatronics, TU Darmstadt. It was embedded in the PRORETA 3 project – a research cooperation with Continental.

At first, I want to thank my supervisor Prof. Jürgen Adamy for his support in all matters. He provided a very pleasant working climate in his lab with great freedom to develop and realize own ideas. I am also grateful to Prof. Hermann Winner for his acceptance to act as a second referee and his motivating and constructive feedback in the PRORETA meetings.

I further owe special thanks to Dr. Volker Willert for many discussions, for his straightforward, friendly character, and the positive reinforcement he gave me throughout my work. Eric Bauer, Felix Lotz, and Matthias Pfromm were the best PRORETA teammates I could hope for. Thanks for numerous conversations and amusing joint free time "voucher" activities!

I am also deeply indebted to all Continental colleagues who were part of PRORETA, namely Ronald Bayer, Thomas Berthold, Stephan Cieler, Johannes Eck, Alfred Eckert, Christina Friedrich, Dr. Ralph Grewe, Wolfgang Günther, Stefan Hegemann, Dr. Andree Hohm, Sybille Huth, Dr. Stephan Kirstein, Matthias Komar, Dr. Benedikt Lattke, Thomas Lingenau, Dr. Stefan Lüke, Benjamin Mathe, Guido Meier-Arendt, Jürgen Pfeiffer, Dr. Peter Rieth, Jonas Schönichen, Dr. Otmar Schreiner, Adam Swoboda, Christian Thur, Dirk Waldbauer, and Jörg Witthaus.

All my past colleagues, I want to thank for the great rmr time, especially Sebastian Bernhard, Florian Damerow, Thomas Guthier, Benjamin Reichardt, and Moritz Schneider for proofreading the manuscript and Valentina Ansel for creating the appealing 3D images of this work. I also want to acknowledge the work of all my former students, who provided me with another valuable perspective on my research topics.

Last but not least, there are my parents to be grateful to. They always supported me in all my decisions and showed exceptional interest in everything I did. The same holds for Marina, whom I deeply thank for her extensive support and understanding in all matters.

Darmstadt, in September 2015 Matthias Schreier

On voit par cet Essai, que la théorie des probabilités n'est au fond, que le bon sens réduit au calcul.

Pierre-Simon Laplace

Contents

Abbreviations and Symbols

Abbreviations

ADAS	Advanced Driver Assistance System
AFFP	Accelerator Force Feedback Pedal
BBF	Binary Bayes Filter
BN	Bayesian Network
BOF	Bayesian Occupancy Filter
CA	Constant Acceleration; Cooperative Automation
CAS	Collision Avoidance System
CCP	Critical Collision Probability
CPHD	Cardinalized Probability Hypothesis Density
CPT	Conditional Probability Table
CTRA	Constant Turn Rate and Acceleration
CTRV	Constant Turn Rate and Velocity
CV	Constant Velocity
DATMO	Detection and Tracking of Moving Objects
DBF	Discrete Bayes Filter
DBN	Dynamic Bayesian Network
DBSCAN	Density Based Spatial Clustering for Applications with Noise
EKF	Extended Kalman Filter
FN	False Negative
FP	False Positive
FR	Follow Road
FV	Follow Vehicle
GNNF	Global Nearest Neighbor Filter
GPB	Generalized Pseudo-Bayesian
HF	Histogram Filter
HMI	Human Machine Interface
HMM	Hidden Markov Model
IF	Information Filter
IMM	Interacting Multiple Model

IMM-PDAF	Interacting Multiple Model Probabilistic Data Association Filter
IMM-UK-PDAF	Interacting Multiple Model Unscented Kalman Probabilistic Data Association Filter
JPDAF	Joint Probabilistic Data Association Filter
KF	Kalman Filter
LAT	Lateral Motion
LC	Lane Change
LE	Lane Existence
LGS	Linear Gaussian System
LON	Longitudinal Motion
MAP	Maximum A Posteriori
MHT	Multi Hypothesis Tracking
MMSE	Minimum Mean Square Error
NNSF	Nearest Neighbor Standard Filter
$\mathrm{OE}_{\mathrm{fro}}$	Object Existence in front
OpenGL	Open Graphics Library
PDA	Probabilistic Data Association
PDAF	Probabilistic Data Association Filter
PDF	Probability Density Function
PF	Particle Filter
PFS	Parametric Free Space
PMD	Predicted-Minimum-Distance
PMF	Probability Mass Function
PSD	Power Spectral Density
RBPF	Rao-Blackwellized Particle Filter
SA	Situation Awareness
SC	Safety Corridor
SLAM	Simultaneous Localization and Mapping
SLAMMOT	Simultaneous Localization, Mapping, and Moving Object Tracking
TB	Target Brake
TE	Turn Existence
TLC	Time-To-Line-Crossing
TN	True Negative
TP	True Positive
TPMD	Time-To-Predicted-Minimum-Distance
TR	Trash maneuver class
TTB	Time-To-Brake
TTC	Time-To-Collision

TTCCP	Time-To-Critical-Collision-Probability
TTK	Time-To-Kickdown
TTO$_{\mathrm{fro}}$	Time-To-Object in front
TTR	Time-To-React
TTS	Time-To-Steer
TTU	Time-To-Turning
TTX	Time-To-X
TU	Turn
UKF	Unscented Kalman Filter
VSMM	Variable Structure Multiple Model

Notation

x	Scalar
$\hat{x}$	Estimated quantity
$\hat{x}^{\mathrm{B}}$	Bayesian point estimate
$\hat{x}^{\mathrm{M}}$	MMSE estimate
$\hat{x}^{\mathrm{MAP}}$	MAP estimate
$\hat{x}_k^-$	Quantity predicted to time step k
$\tilde{x}$	Sample drawn from a distribution with pdf $p(x)$
$\dot{x}$	Time derivative of x
$\boldsymbol{x}$	Column vector
$\boldsymbol{x}^T$	Row vector
$\boldsymbol{x}_{0:k}$	Sequence of vectors $\boldsymbol{x}_{0:k} = \{\boldsymbol{x}_0, \boldsymbol{x}_1, \dots, \boldsymbol{x}_{k-1}, \boldsymbol{x}_k\}$
$\{\boldsymbol{x}_i\}_{i=1}^N$	Set of N vectors
$\boldsymbol{\mathcal{X}}$	Set of vectors ordered column-wise in matrix form
$\boldsymbol{X}$	Matrix
$\boldsymbol{X}^T$	Transpose of matrix $\boldsymbol{X}$
$\boldsymbol{X}^{-1}$	Inverse of matrix $\boldsymbol{X}$
$\lvert\boldsymbol{X}\rvert$	Determinant of matrix $\boldsymbol{X}$
$\sqrt{\boldsymbol{X}}$	Matrix square root of $\boldsymbol{X}$ of the form $\boldsymbol{X} = \sqrt{\boldsymbol{X}}\sqrt{\boldsymbol{X}}^T$
$\boldsymbol{X} \succ \boldsymbol{0}$	Matrix $\boldsymbol{X}$ is symmetric and positive definite
$\boldsymbol{X} \prec \boldsymbol{0}$	Matrix $\boldsymbol{X}$ is symmetric and negative definite
$\boldsymbol{X} \oplus \boldsymbol{x}$	Addition of the vector $\boldsymbol{x}$ to each column of matrix $\boldsymbol{X}$
$\boldsymbol{0}$	Zero vector or matrix
$\emptyset$	Empty set
$\mathbb{1}$	Identity vector or matrix

$\operatorname{diag}(\cdot)$	Diagonal matrix with $\operatorname{diag}(x_1, \ldots, x_n) = \begin{pmatrix} x_1 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & x_n \end{pmatrix}$
$\mathbb{E}(\cdot)$	Expectation operator
$\int_{\boldsymbol{x}} (\cdot) \mathrm{d}\boldsymbol{x}$	Integration over the whole range of $\boldsymbol{x}$, e.g. if $\boldsymbol{x} \in \mathbb{R}^n$, then $\int_{\boldsymbol{x}} (\cdot) \mathrm{d}\boldsymbol{x} = \int_{\mathbb{R}^n} (\cdot) \mathrm{d}\boldsymbol{x}$
$f(\cdot)$	Scalar function
$\boldsymbol{f}(\cdot)$	Vector function
$\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}$	Derivative of a scalar function $f(\boldsymbol{x})$ with respect to $\boldsymbol{x} \in \mathbb{R}^n$ in denominator layout with $\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}} = (\frac{\partial f(\boldsymbol{x})}{\partial x_1} \cdots \frac{\partial f(\boldsymbol{x})}{\partial x_n})^T$
$\frac{\partial \boldsymbol{f}(\boldsymbol{x})}{\partial \boldsymbol{x}}$	Jacobian matrix of a vector function $\boldsymbol{f}(\boldsymbol{x})$
$p(x)$	Pdf of a continuous random variable x
$p(\boldsymbol{x})$	Pdf of a continuous random vector $\boldsymbol{x}$
$p(X)$	Pmf of a discrete random variable X; For binary X: Probability that X is true
$p(\neg X)$	For binary X: Probability that X is false
$p(X_j)$	Probability that a discrete random variable X is in state j: $p(X_j) = p(X = j)$
$p(\boldsymbol{x}, \boldsymbol{y})$	Joint pdf of $\boldsymbol{x}$ and $\boldsymbol{y}$
$p(\boldsymbol{x} \mid \boldsymbol{y})$	Conditional pdf of $\boldsymbol{x}$ given $\boldsymbol{y}$
$p(X \mid Y)$	Conditional pmf of X given Y written as a CPT

Important Functions and Transformations

$\Gamma(\cdot)$	Gamma function
$\delta(\cdot)$	Dirac delta distribution
$\boldsymbol{f}(\cdot)$	System function
$\boldsymbol{h}(\cdot)$	Measurement function
$I_\mathrm{C}(\cdot)$	Collision indicator function
$l(\cdot)$	Log odds ratio
$L(\cdot)$	Loss function
$\mathcal{L}^{-1}(\cdot)$	Inverse Laplace transformation
$\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{P})$	Multivariate Gaussian pdf of a random vector $\boldsymbol{x}$ with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{P}$

$p(\boldsymbol{x}_k \mid \boldsymbol{z}_{1:k})$	Filtering distribution
$p(\boldsymbol{x}_k \mid \boldsymbol{x}_{k-1})$	Transition density
$p(\boldsymbol{z}_k \mid \boldsymbol{x}_k)$	Measurement likelihood
$p(X \mid \boldsymbol{z}_k)$	Inverse sensor model

Symbols

Latin Capital Letters

$A_{\mathrm{b,min}}$	Area of minimum oriented bounding rectangle
A_{f}	Area of newly free rectangle
A_{h}	Area of free space hole
A_{o}	Area of newly occupied rectangle
A_{th}	Threshold on area quotient in merging step
$\boldsymbol{A}$	System matrix of a discrete-time system
$\boldsymbol{A}_{\mathrm{c}}$	System matrix of a continuous-time system
$B(s)$	B-spline basis function with curve parameter s
$\boldsymbol{B}$	Input matrix of a discrete-time system
$\boldsymbol{B}_{\mathrm{c}}$	Input matrix of a continuous-time system
$C_{\mathrm{I},k}(T_{\mathrm{P}})$	Collision event for time instant $k + T_{\mathrm{P}}$
$C_k(T_{\mathrm{P}})$	Combined collision event for prediction horizon $\{k : k + T_{\mathrm{P}}\}$
$C_{\mathrm{PFS},k}(T_{\mathrm{P}})$	Collision event for prediction horizon $\{k : k + T_{\mathrm{P}}\}$ with respect to PFS map
$C_{\mathrm{V}_i,k}(T_{\mathrm{P}})$	Collision event for prediction horizon $\{k : k + T_{\mathrm{P}}\}$ with respect to i-th vehicle
$\boldsymbol{C}$	Cross covariance matrix between predicted measurements and states
$\mathcal{D}_{\mathcal{S}}(\mathcal{F}')$	Dilated set of $\mathcal{F}'$ with structuring element $\mathcal{S}$
E	Ego vehicle
$\mathcal{E}_{\mathcal{S}}(\mathcal{F})$	Eroded set of $\mathcal{F}$ with structuring element $\mathcal{S}$
$\mathcal{F}$	Set of free cells
$\mathcal{F}'$	Relevant eroded subset of free cells
$\boldsymbol{H}$	Measurement matrix
$\boldsymbol{I}$	Information matrix
$\boldsymbol{K}$	Kalman gain
L	Number of spline spans
M	Motion mode; Driving maneuver
M_{c}	Center of circular arc in TU maneuver

$\mathcal{M}$	Model set
$\mathcal{M}_\mathrm{s}$	Model subset
N	Number of Monte Carlo samples
N_b	Number of B-spline basis functions in PFS map
N_c	Number of circles in PFS map
N_DB	Minimal point number in DBSCAN
N_k	Number of B-spline knots
N_min	Minimum number of detections
N_p	Number of outer boundary cells for B-spline tracking
N_r	Number of oriented rectangles in PFS map
N_v	Number of validated measurements
N_z	Number of measurements in tracking
$\mathbb{N}$	Set of non-zero natural numbers
$\mathbb{N}^0$	Set of natural numbers including zero
$\mathcal{O}_\mathcal{S}(\mathcal{F})$	Opened set of $\mathcal{F}$ with structuring element $\mathcal{S}$
P_D	Detection probability
P_G	Gate probability
P_{y_R}	Variance in Ornstein-Uhlenbeck process
$\boldsymbol{P}$	Error covariance matrix
$\boldsymbol{P}^\star$	Mixed initial covariance matrix in IMM Filter
Q_n	Noise intensity
$\boldsymbol{Q}$	Process noise covariance matrix
$\boldsymbol{Q}_\mathrm{n}$	Noise intensity matrix
$\mathbb{R}$	Set of real numbers
$\boldsymbol{R}$	Measurement noise covariance matrix
S	Spline span
S_M	Neighborhood size of median filter
$\mathcal{S}_{\boldsymbol{p}}$	Structuring element with origin placed at $\boldsymbol{p}$
$\mathcal{S}(\boldsymbol{x})$	Set of occupied points of an object with configuration $\boldsymbol{x} = (x, y, \psi)^T$
$\mathcal{S}(\boldsymbol{x}_\mathrm{PFS})$	Set of occupied points of a PFS map with configuration $\boldsymbol{x}_\mathrm{PFS}$
$\boldsymbol{S}$	Innovation covariance matrix
T	Sampling time
T_c	Time constant in Ornstein-Uhlenbeck process
T_P	Prediction time steps
T_r	Reference maneuver time in LC maneuver
$\boldsymbol{U}(s)$	Matrix of concatenated B-spline basis function vectors with curve parameter s

V	Volume of the validation region; Vehicle
$V_\mathrm{U}(q)$	Volume of the q-dimensional unit hypersphere
$\mathcal{V}$	Validation region
$\boldsymbol{\mathcal{X}}$	Matrix of sigma points in UKF state prediction
$\boldsymbol{\mathcal{X}}^*$	Matrix of sigma points in UKF state prediction
$\mathcal{Z}$	Measurement set
$\mathcal{Z}_\mathrm{v}$	Validated measurement set
$\boldsymbol{\mathcal{Z}}$	Matrix of sigma points in UKF measurement prediction

Latin Lowercase Letters

a	Acceleration in driving direction
a_max	Maximum acceleration
a_min	Minimum acceleration
$a_\mathrm{c,max}$	Maximum centripetal acceleration in TU maneuver
$a_\mathrm{f,max}$	Maximum acceleration in FV maneuver
$a_\mathrm{f,min}$	Minimum acceleration in FV maneuver
$a_\mathrm{R,lat}$	Lateral acceleration perpendicular to road course
$a_\mathrm{R,lon}$	Longitudinal acceleration along road course
$a_\mathrm{R,lon,r}$	Longitudinal reference acceleration along road course
b	Abbreviation in PDA
$\boldsymbol{b}(s)$	Vector of B-spline basis functions
c	Cell length
c_m	Roundness metric
C	Vehicle corner point
d	B-spline order
Δd	Safety distance in TB maneuver
d_fo	Distance between newly free and occupied rectangle
d_r	Reference safety distance in TB maneuver
$d_\mathrm{R,fro}$	Distance to vehicle in front along road course
$d_\mathrm{R,TU}$	Distance to turn in TU maneuver
d_th	Distance threshold in merging step
e	Abbreviation in PDA
f	Number of vehicles except ego vehicle
$\boldsymbol{i}$	Information vector
i	Running index
j	Running index
k	Time step; Knot position
l	Object length in tracking

l_{n}	Ray length along normals of free space curve
l_{r}	Reference maneuver length in LC maneuver
$\boldsymbol{l}$	Line segment
m	Running index
m_i	Grid cell i
m_{th}	Threshold on free probability for grid segmentation
$\boldsymbol{m}$	Conventional occupancy grid map
$\boldsymbol{m}_{\mathrm{opt}}$	Optimized occupancy grid map
n	State vector dimension
$\boldsymbol{n}(s)$	Normal vector of B-spline curve with curve parameter s
p	Input vector dimension
p_{h}	Perimeter of free space hole
$p(m_i)$	Occupancy probability of i-th grid cell
p_{o}	Percentage of occupied cells in merging step
p_{th}	Threshold on occupied cell percentage in merging step
q	Measurement vector dimension
q_{th}	Threshold on temporal difference grid map
$\boldsymbol{q}$	De Boor point
q_x	De Boor point x-coordinate
q_y	De Boor point y-coordinate
r	Number of prediction models
r_{c}	Turn radius in TU maneuver
r_{m}	Rectangularity metric
r_{DB}	Neighborhood radius in DBSCAN
r_{SE}	Radius of disc-shaped structuring element
$\boldsymbol{r}(s)$	B-Spline curve with curve parameter s
s	Laplace variable; Curve parameter
u	Input scalar
$\boldsymbol{u}$	Input vector
v	Velocity in driving direction
v_{rel}	Relative velocity to object in front
$v_{\mathrm{R,lat}}$	Lateral velocity perpendicular to road course
$v_{\mathrm{R,lon}}$	Longitudinal velocity along road course
$v_{\mathrm{R,lon,max}}$	Maximum velocity in TU maneuver
$\boldsymbol{v}$	Measurement noise vector; Innovation vector
w	Process noise scalar; Object width in tracking
w_{c}	Continuous-time process noise scalar
w_{L}	Lane width
w_{V}	Vehicle width

$w_i^{(m)}, w_i^{(c)}$	Weights of i-th sigma point in UKF
$\boldsymbol{w}$	Process noise vector
$\boldsymbol{w}_\mathrm{c}$	Continuous-time process noise vector
x	x-coordinate of object center in map-fixed system (Tracking) or global system (Prediction)
x_I	Intersection point road coordinate between road tangents in TU maneuver
x_R	x-coordinate of vehicle center in road-fixed system
$x_\mathrm{R,s}$	Maneuver start coordinate in LC maneuver
$\boldsymbol{x}$	State vector
$\boldsymbol{x}^\star$	Mixed initial state vector in IMM Filter
$\boldsymbol{x}_\mathrm{c}$	State vector of circles in PFS map
$\boldsymbol{x}_\mathrm{P}$	B-Spline control vector in PFS map
$\boldsymbol{x}_\mathrm{PFS}$	State vector of PFS map
$\boldsymbol{x}_\mathrm{r}$	State vector of oriented rectangles in PFS map
y	y-coordinate of object center in map-fixed system (Tracking) or global system (Prediction)
y_R	y-coordinate of vehicle center in road-fixed system
$\boldsymbol{z}$	Measurement vector

Greek Letters

α	Angle in TU maneuver
α_U	Sigma point scaling parameter in UKF
β	Measurement-to-target association probability
β_U	Sigma point scaling parameter in UKF
γ_G	Gate threshold
γ_U	Sigma point scaling parameter in UKF
$\boldsymbol{\gamma}$	Noise gain vector
θ	Association event in PDA
κ	Kulpa's perimeter correction factor
κ_U	Sigma point scaling parameter in UKF
λ	Model likelihood
λ_U	Sigma point scaling parameter in UKF
μ	Mode probability
$\mu_{i\|j}$	Mixing probability
μ_th	Mode probability threshold
$\boldsymbol{\mu}$	Mode probability vector
$\boldsymbol{\Pi}$	Model transition probability matrix

$\sigma_{\Delta a}$	Standard deviation of acceleration increments in discrete-time CTRA model
$\sigma_{\Delta a_{\mathrm{R,lon}}}$	Standard deviation of acceleration increments in discrete-time CA model
$\sigma_{\Delta\omega}$	Standard deviation of yaw rate increments in discrete-time CTRA model
$\sigma_{\psi_{\mathrm{R}}}$	Standard deviation of yaw angle
$\sigma_{y_{\mathrm{R}}}$	Limiting standard deviation in Ornstein-Uhlenbeck process
$\sigma_{y_{\mathrm{R,s}}}$	Standard deviation of lateral maneuver origin in LC maneuver
τ	Time gap in FV maneuver
τ_{r}	Reference time gap in FV maneuver
ϕ_{f}	Angle of newly free rectangle
ϕ_{o}	Angle of newly occupied rectangle
ϕ_{th}	Threshold on angle difference in merging step
ψ	Yaw angle with respect to global coordinate system
ψ_{R}	Yaw angle with respect to road course
ω	Yaw rate

Abstract

Advanced Driver Assistance Systems (ADAS) already make a major contribution to driving safety. To further increase this contribution, it is, however, vital that future intelligent vehicles perceive, predict, and assess their environment more comprehensively. In this context, the present dissertation approaches the questions i) how to represent the driving environment adequately within an environment model, ii) how to obtain such a representation, and iii) how to predict the future traffic scene evolution for proper criticality assessment. Bayesian inference provides the common theoretical framework of all designed methods.

Based on the shortcomings of existing environment representations, a novel parametric representation of general driving environments is first introduced in this work. It consists of a combination of dynamic object maps for moving objects and so-called Parametric Free Space (PFS) maps for static environment structures. PFS maps model the environment by a closed curve around the vehicle, which encloses relevant drivable free space. The representation compactly describes all essential information contained in common occupancy grid maps, suppresses irrelevant details, and consistently separates between static and dynamic environment objects.

A novel method for grid mapping in dynamic road environments provides the basis to realize this representation. Therein, dynamic cell hypothesis are detected, clustered, and subsequently tracked and classified with an adaptive Bayesian multiple model filter for jump Markov nonlinear systems – the so-called Interacting Multiple Model Unscented Kalman Probabilistic Data Association Filter (IMM-UK-PDAF). The intermediate result is a dynamic object map and an optimized grid of the static driving environment. From the optimized grid, relevant free space is then extracted by methods of image analysis, and robustly converted to a PFS map in a final B-Spline contour tracking step. Evaluations and experiments, which were performed with an experimental vehicle equipped with radars and a stereo camera in real driving environments, confirm the advantages of the real-time capable approach.

The so-obtained representation additionally forms the basis of a novel method for long-term trajectory prediction and criticality assessment.

Therein, a three-layered Bayesian network is used to infer current driving maneuvers of traffic participants initially. A trash maneuver class allows the detection of irrational driving behavior and the seamless application from highly-structured to non-structured environments. Subsequently, maneuver-based prediction models in form of stochastic processes are presented and employed to predict the vehicle configurations under consideration of uncertainties in the maneuver executions. Finally, the criticality time metric Time-To-Critical-Collision-Probability (TTCCP) is introduced as a generalization of the time metric Time-To-Collision (TTC) for arbitrary, uncertain, multi-object driving environments and longer prediction horizons. The TTCCP considers all uncertain, maneuver-based predictions and is estimated via Monte Carlo simulations. Simulations confirm its potential to suppress false warnings, to generate timely true warnings, and to generate warnings in critical almost-collision situations effectively.

All methods are part of the driver assistance system PRORETA 3, which has been co-developed in the context of this thesis. It constitutes a novel, integrated approach to collision avoidance and vehicle automation and thereby makes a valuable contribution to realize the Vision Zero – the vision of a future without traffic deaths.

Kurzfassung

Fahrerassistenzsysteme leisten bereits heute einen bedeutenden Beitrag zur Sicherheit im Straßenverkehr. Um diesen Beitrag weiter zu erhöhen, müssen zukünftige intelligente Fahrzeuge ihre Umgebung jedoch noch eingehender wahrnehmen, prädizieren und bewerten. In diesem Kontext behandelt die vorliegende Dissertation die Fragen, i) wie die Fahrumgebung geeignet in einem Umfeldmodell repräsentiert werden kann, ii) wie eine solche Repräsentation realisierbar ist und iii) wie die zukünftige Entwicklung der Verkehrssituation sowie deren Kritikalität abgeschätzt werden kann. Bayessche Inferenzverfahren bilden das gemeinsame theoretische Gerüst aller hierzu entworfenen Methoden.

Ausgehend von den Limitierungen bestehender Umgebungsrepräsentationen wird in dieser Arbeit zunächst eine neue parametrische Repräsentation allgemeiner Fahrumgebungen eingeführt. Sie besteht aus einer Kombination aus dynamischen Objektkarten für sich bewegende Objekte und sogenannten parametrischen Freiraumkarten für statische Umgebungsstrukturen. Letztere modellieren die Umgebung mittels einer geschlossenen Kurve, die das Fahrzeug umgibt und die relevante befahrbare Freiräume einschließt. Die Repräsentation stellt alle wesentlichen Informationen, die auch in den verbreiteten Belegungsgitterkarten enthalten sind, kompakt dar, unterdrückt irrelevante Details und unterscheidet konsistent zwischen dynamischen und statischen Objekten.

Eine neue Methode zur Erstellung von Belegungsgitterkarten in dynamischen Fahrumgebungen bildet die Basis zur Realisierung dieser Repräsentation. Hierbei werden dynamische Zellhypothesen detektiert, gruppiert und objektbasiert mit einem adaptiven Bayesschen Mehrmodellfilter für nichtlineare Markov-Sprungprozesse – dem sogenannten Interacting Multiple Model Unscented Kalman Probabilistic Data Association Filter (IMM-UK-PDAF) – zeitlich verfolgt und klassifiziert. Das Zwischenresultat ist eine dynamische Objektkarte sowie eine optimierte Belegungsgitterkarte der statischen Umgebung. Aus der optimierten Gitterkarte werden daraufhin relevante Freiräume mit Methoden der Bildverarbeitung extrahiert und im Rahmen eines nachgeschalteten B-Spline-Konturverfolgungsschritts robust in eine parametrische Freiraumkarte überführt. Evaluationen und Experimente, die in realen Fahrumgebungen mit einem mit Radarsensoren und Stereokamera ausgerüsteten Versuchsfahrzeug ausgeführt wurden, bestätigen die Vorteile des echtzeitfähigen Ansatzes.

Die so erzeugte Umgebungsrepräsentation dient darüber hinaus als Basis für ein neues Langzeit-Trajektorienprädiktions- und Kritikalitätsbewertungsverfahren. Den Ausgangspunkt hierfür bildet ein dreischichtiges Bayessches Netz, das genutzt wird, um auf die aktuellen Fahrmanöver der Verkehrsteilnehmer zu schließen. Eine Restmanöverklasse erlaubt zusätzlich die Erkennung irrationaler Fahrerhandlungen sowie die nahtlose Anwendbarkeit von hochstrukturierten bis hin zu unstrukturierten Fahrumgebungen. Weiterhin werden manöverbasierte Prädiktionsmodelle in Form stochastischer Prozesse vorgestellt und zur Vorhersage der Fahrzeugkonfigurationen unter Berücksichtigung von Unsicherheiten in der Manöverausführung genutzt. Abschließend wird das Kritikalitätszeitmaß Time-To-Critical-Collision-Probability (TTCCP) als Erweiterung des Zeitmaßes Time-To-Collision (TTC) für beliebige unsichere Fahrumgebungen mit mehreren Objekten eingeführt, das auch für längere Prädiktionshorizonte geeignet ist. Die TTCCP bezieht alle unsicheren, manöverbasierten Vorhersagen mit ein und wird mittels einer Monte-Carlo-Simulation geschätzt. Simulationen bestätigen das Potential des Ansatzes Fehlwarnungen zu unterdrücken, korrekte Warnungen frühzeitig zu erzeugen sowie auch in kritischen Beinahezusammenstoßsituationen effektiv zu warnen.

Anwendung finden die Methoden im Assistenzsystem PRORETA 3, das im Rahmen dieser Arbeit mitentwickelt wurde. Dieses stellt einen integralen Ansatz zur Kollisionsvermeidung und Fahrzeugautomatisierung dar und leistet damit seinerseits einen wertvollen Beitrag zur Realisierung der Vision Zero – der Vision einer Zukunft ohne Verkehrstote.

1 Introduction

More than 90 % of all traffic accidents are at least partly caused by human error [248]. This fact clearly motivates the need of driver assistance systems to turn the *Vision Zero* into reality – a vision of a future in which no one will be killed or seriously injured on the road [258]. Remarkable success has already been achieved in lowering the number of traffic accidents and deaths over the last decades. Fig. 1.1 illustrates the relative change of the number of motorized vehicles, injured persons, and deaths in German road traffic over the years 1970–2014 based on [238]. Although the number of vehicles more than doubled in this time interval, the number of injured persons decreased by a factor of 1.5 and the number of deaths even decreased by more than a factor of 6.

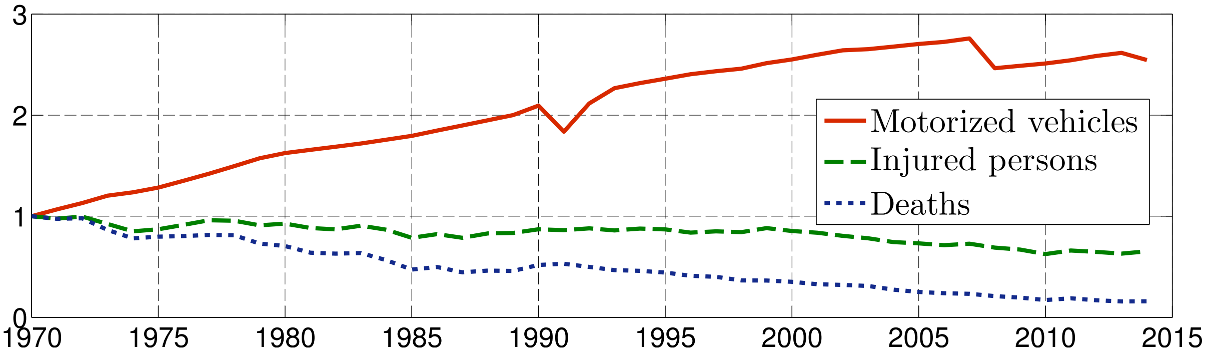


Figure 1.1: Relative change of the number of motorized vehicles, injured persons, and deaths in German road traffic over the years 1970–2014 based on [238].

This promising development has provably been fostered by passive safety systems, such as seatbelts and airbags, as well as by early active safety systems, e.g. anti-lock braking (since 1987) or electronic stability control (since 1995) [231]. However, passive systems can only mitigate the effects of occurring accidents, while such early active systems are limited to interventions that can be triggered purely based on the ego vehicle's states, which are estimated with the help of proprioceptive sensors. Environment knowledge is not taken into account. Therefore, so-called *Advanced Driver Assistance Systems* (ADAS) have been developed, which additionally consider the state of the environment observed by exteroceptive sensors, such as radars, lidars, sonars, or cameras. Examples already in market include collision warning and intervention systems, lane departure warnings, lane keeping support systems, lane change assists, adaptive cruise control, traffic-jam assists, vision enhancing systems, park assists, and many more [48, 273]. To date, each of these assistants mainly operates as an independent, individually designed unit [70]. Future ADAS, in contrast, will act cooperatively [70], and the development will probably culminate in holistic ADAS that are not composed of individually designed functions any more but directly suitable for a large variety of different traffic situations. This thesis originates from the development of such an integrated ADAS.

One of the main challenges in this regard is the choice and development of an adequate environment model, i.e. a suitable representation of the driving environment. The environment model provides a holistic view of the current traffic scene and constitutes a consistent knowledge base for all assistance functions [67]. It thus permits the decoupling of the function from the specific sensor setup. In this context, the first main question addressed in this thesis is the following:

1. How can *relevant* environment information be represented in a *consistent*, *compact*, *easily interpretable* way with *sufficient detail* to realize future, integrated ADAS?

Since it is of limited use to find an ideal representation that cannot be generated, the second main question addressed is:

2. How can the environment representation be generated *online* in a *robust*, *computationally efficient*, *sensor-independent* way?

These two questions relate to the representation and estimation of the *current* state of the environment. For correct decisions, warnings, and interventions, it is, however, important not just to represent this current state but to predict the uncertain future evolution of the traffic scene. Only then it is possible to correctly assess the situation's criticality. The third main question this thesis is based on is thus:

3. How can a system for *long-term*, *multi-object* trajectory prediction and criticality assessment in *arbitrary* driving environments be realized that takes *uncertainties* into account?

These questions will be motivated in greater detail in the corresponding chapters.

1.1 Contributions

The main contributions of this thesis, which directly relate to the mentioned questions, can be summarized as follows:

- Novel metric environment representation for intelligent vehicles by combining dynamic object maps with introduced *Parametric Free Space* (PFS) maps [223, 226].
- Novel method for grid mapping and tracking in dynamic road environments [225, 226].
    - Novel grid-based dynamic object detection and reconstruction by using knowledge about newly available free space.
    - Novel filter combination for tracking jump Markov nonlinear systems in clutter.
    - Novel multiple model tracking filter-based classification of dynamic grid cell hypothesis.
- Novel method for grid-based free space detection and estimation [221, 223, 226].
    - Novel free space detection by morphological image analysis applied to occupancy grid maps.
    - Novel B-spline contour tracking of arbitrary free space shapes.
- Novel method for probabilistic, long-term trajectory prediction and criticality assessment [222, 224].
    - Novel three-layered Bayesian network designed for driving maneuver detection.
    - Novel probabilistic, maneuver-based trajectory prediction models.
    - Novel criticality metric *Time-To-Critical-Collision-Probability* (TTCCP) as an extension of the *Time-To-Collision* (TTC) for arbitrary, uncertain, multi-object driving environments.

All contributions harmonize naturally with each other as they are based on the same underlying Bayesian theory. A combination of all methods has been integrated in a developed environment simulator, while environment representation-related algorithms have additionally been fully implemented and tested on an experimental vehicle. They are part of the holistic driver assistance system PRORETA 3 [25, 56, 274, 275], which the author co-developed in the course of this thesis.

Individual contributions have already been made available to the *Intelligent Transportation Systems* and the *Robotics and Automation* community in the author's publications [25, 217–226, 274, 275], which are summarized in appendix B, p. 217.

The next section describes the structure of this thesis and highlights how the chapters and contributions interrelate.

1.2 Thesis Structure

The thesis structure is shown in Fig. 1.2. Chapters that contain novel contributions are highlighted in gray. After the introductory Chapter 1, common metric environment representations are presented and discussed with respect to their suitability for future ADAS in Chapter 2, which closes with the description of the proposed novel representation. Since Bayesian inference plays a major role in its realization, all relevant theoretical foundations are presented subsequently in Chapter 3 with a focus on the employed filters and their applications in target tracking and occupancy grid mapping. Chapter 4 is devoted to a novel grid mapping and tracking approach in *dynamic* road environments, which results in a dynamic object map and an optimized grid map that only contains static environment structures. In Chapter 5, this optimized grid is the basis for the extraction of so-called *Parametric Free Space* (PFS) maps, which compactly represent the static driving environment in form of drivable free space. This completes the generation process of the proposed metric environment representation. The final representation then serves as the foundation of a novel long-term prediction and criticality assessment approach, which is introduced and described in detail in Chapter 6.

Up to that point, all presented methods are general and can be employed for different ADAS and intelligent vehicles. Chapter 7, in contrast, embeds the methods into the novel, holistic ADAS PRORETA 3, which not only provides safety warnings and interventions in a large variety of different traffic scenarios but also a maneuver-based comfort driving mode in form

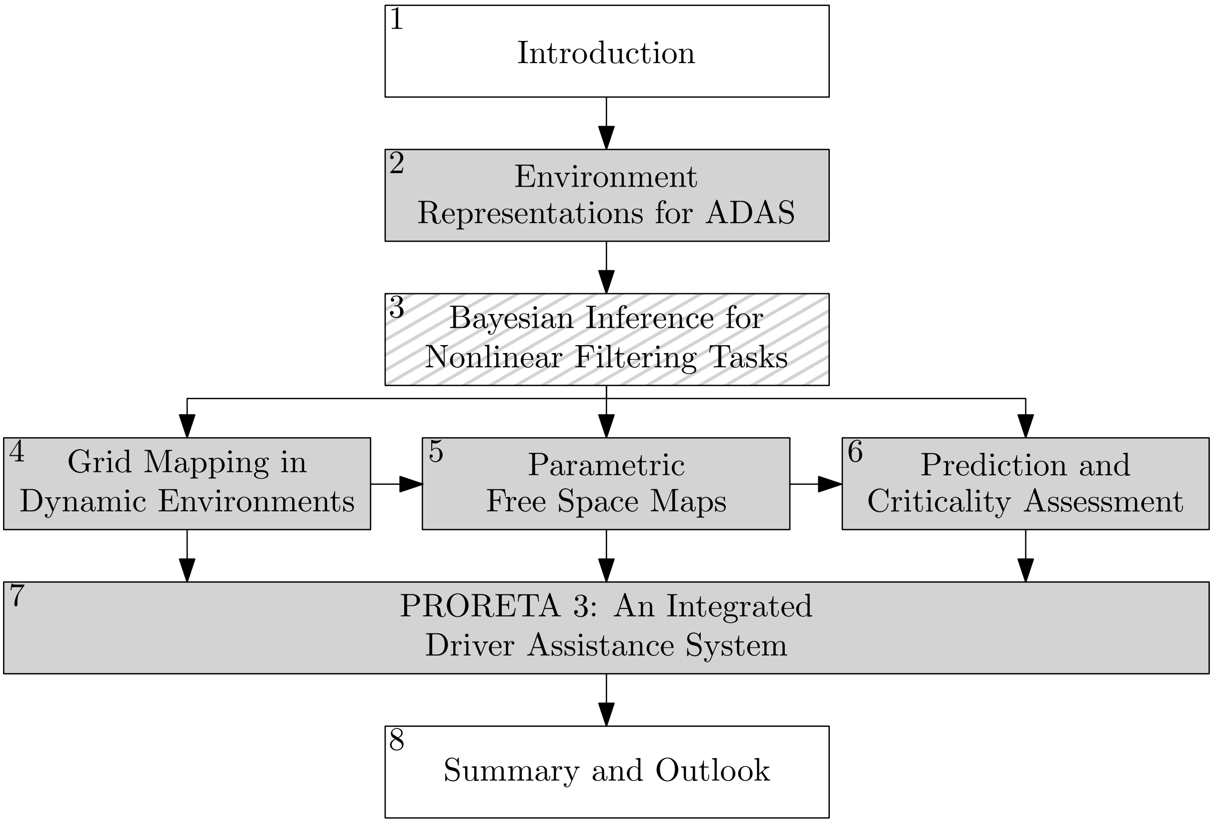

1
Introduction
2
Environment Representations for ADAS
3
Bayesian Inference for Nonlinear Filtering Tasks
4
Grid Mapping in Dynamic Environments
5
Parametric Free Space Maps
6
Prediction and Criticality Assessment
7
PRORETA 3: An Integrated Driver Assistance System
8
Summary and Outlook

Figure 1.2: Thesis structure with possible reading paths. Chapters containing majors contributions are shaded in gray, whereas chapters with minor novelties are shown hatched.

of a cooperative automation between vehicle and driver. The main results of this thesis along with possible future work are finally summarized in Chapter 8.

Although the best reading experience is given if all chapters are read one after each other, Chapters 4-6 have carefully been designed to be self-contained so that readers only interested in the specific areas *grid mapping in dynamic environments*, *PFS maps and free space detection*, or *prediction and criticality assessment* can directly jump to the corresponding topic of interest. Therefore, related work on these individual topics is directly located in the corresponding chapters.

2 Environment Representations for ADAS

The present chapter first gives an overview of common metric environment representations and compares them with respect to their suitability for future ADAS. Subsequently, the proposed representation is introduced, which combines dynamic object maps with a novel description of static driving environments in form of so-called *Parametric Free Space* (PFS) maps.

2.1 Introduction and Motivation

The real world in all its facets is exceptionally complex. Any system that operates therein thus needs to perform some kind of abstraction and build a simplified model of its environment. Such an *environment model* is an internal representation of the surroundings of a mobile robot, e.g. an intelligent vehicle. The model always has to be adapted to the tasks and the type of environment the system is located in [49]. In principle, an environment representation should be as compact as possible and as general as needed for the given tasks. Overly simplified models involve the danger that specific tasks cannot be performed since the model does not include the required information, whereas overly detailed models obscure relevant information and complicate subsequent processing.

In this context, one distinguishes between *topological* and *metric* representations. Topological representations provide high-level, qualitative information about spatial relations of world entities without specifying their exact locations in space. For example, the knowledge that there is another lane next to the current lane is part of such a topological map, which is normally represented as a graph. Metric representations, in contrast, encode spatial information quantitatively in a coordinate frame and are thus a prerequisite for tasks like trajectory planning. Both types are often used in parallel in one system. The present work focuses on *metric* representations, which in turn differ greatly in their level of detail.

In the field of intelligent vehicles, it is a known fact [92] that solely sparse object-based environment models – as used in today's adaptive cruise control or emergency braking systems – are not sufficient for the realization of future ADAS such as *Collision Avoidance Systems* (CAS) in arbitrary environments [25] or general lateral vehicle guidance systems, e.g. in narrow inner-city roads [172], in construction sites [155], or with respect to arbitrary elevated objects [25]. However, an established interface for the required dense environment models for these near-future ADAS functions does not exist [92], and there is no clear consensus which representation is the most suitable. Ideally, the model should

- be compact enough for transmission between electronic control units with limited transmission bandwidth,
- be sufficiently general to permit the realization of a large variety of future ADAS functions,
- suppress irrelevant environment details to facilitate situation interpretation and planning,
- represent free space explicitly to permit safety-related trajectory planning,
- be generable online in a computationally inexpensive, robust way with all exteroceptive sensors used in the automotive domain,
- be able to cope with dynamic objects and to represent them adequately, and
- allow to incorporate sensor uncertainties.

In recent times, grid-based representations [76] have increasingly been used, e.g. for realizing CAS with an incorporation of available free space [205], lateral vehicle guidance [155], or overtaking assistants [105]. However, the disadvantages of grid-based representations, namely the required transmission bandwidth and memory resources [92], might limit their application in near-future ADAS due to the slow data transfer over low data rate busses or vehicle-to-vehicle communication interfaces. Moreover, grid-based representations suffer from discretization effects, contain a large amount of irrelevant details such as unreachable free space areas, and have only limited capability to represent dynamic objects.

Therefore, a novel environment representation is proposed in this thesis to cope with these issues. Due to its widespread usage, it directly builds upon a conventional grid mapping algorithm. The resulting grid, however, only serves as an intermediate step for the generation of two parametric, continuous representations: a dynamic object map for dynamic entities as well as a so-called Parametric Free Space (PFS) map for the static environment. Before these are introduced in detail, common metric environment representations are first discussed in the following.

2.2 Related Work

In the discussion of metric environment models, several relations have to be kept in mind. The lower its abstraction level, the more general is an environment model. It contains more details because fewer model assumptions are incorporated. On the downside, less abstract models have higher transmission bandwidth requirements and, even more important, are harder to interpret by ADAS algorithms as these have to cope with irrelevant details. In contrast, the higher the abstraction level of an environment model, the fewer details are preserved. Hence, the required bandwidth is reduced, but the risk that specific functions cannot be realized is increased. Therefore, one has to strike a balance with the aim to find an abstract, generic representation that still permits the realization of future ADAS functions but without containing irrelevant details. In general, the more structured the environment, the more abstract is the possible generic representation.

Fig. 2.1 illustrates common metric environment models from mobile robotics and the automotive domain. These are discussed with respect to their suitability for future ADAS in the following, starting with the highest abstraction level.

Feature maps (Fig. 2.1a) are the most compact representation considered. They represent the environment either by a set of point landmarks (landmark-based maps [245]), by a collection of lines (line maps [49]), or, more general, by geometric shapes (object maps [245]). Feature maps are only suitable in at least semi-structured environments that can be fully described by only a small vocabulary of geometric elements [245]. Moreover, they have to cope with the error-prone correspondence problem of associating individual measurements with already detected features. For a successful association, a sparse set of distinctive features is required, and the resulting maps lack detailed geometric descriptions as a conse-

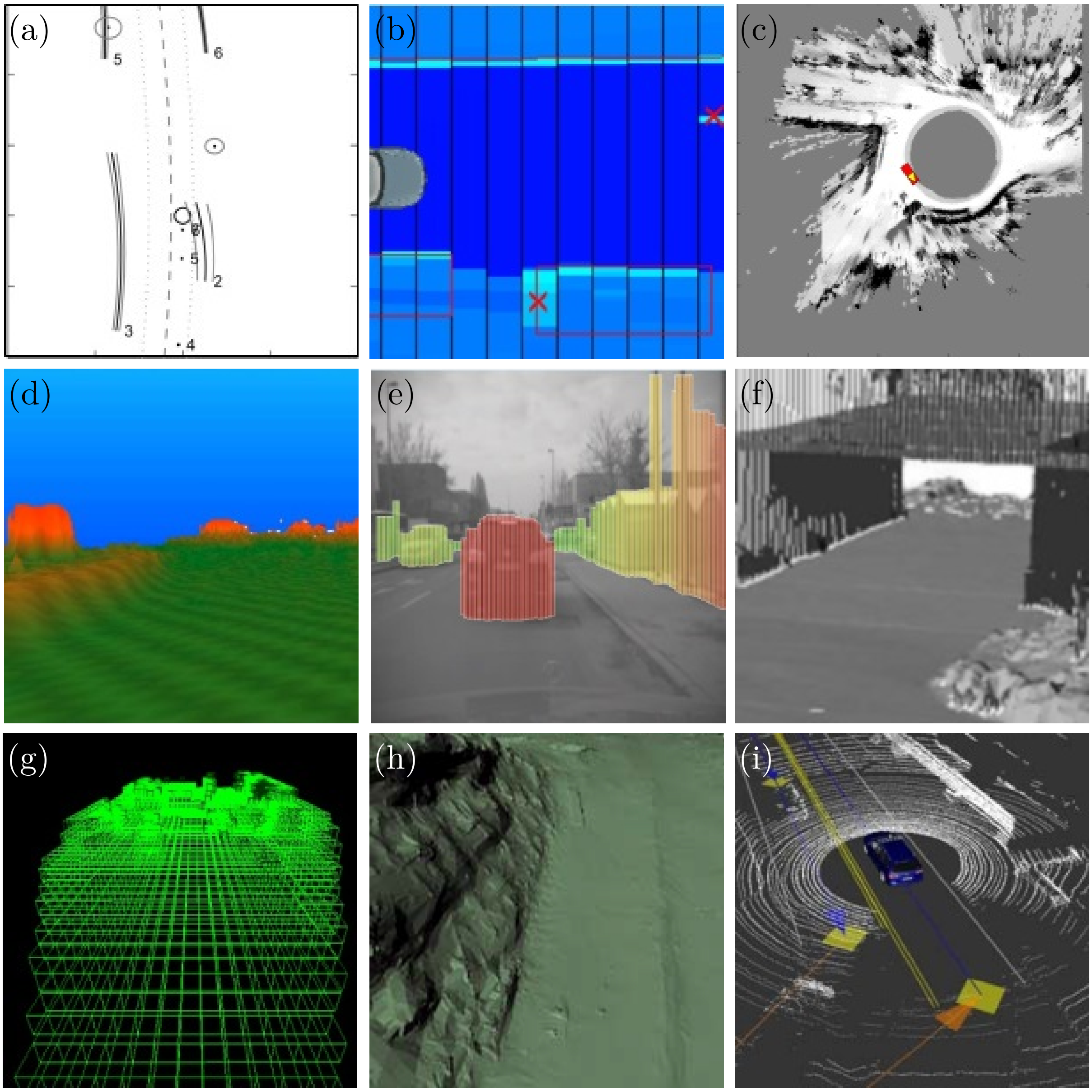


Figure 2.1: Environment representations of various levels of abstractness from most abstract (a) to most general (i): feature map [160] (a), interval map [267] (b), 2D grid map [223] (c), elevation map [61] (d), stixel world [194] (e), multi-level surface map [249] (f), voxel grid [47] (g), mesh [49] (h), raw sensor data [193] (i). Images taken from the mentioned sources.

quence [245]. Nevertheless, a combination of point and line maps is used in [160] for the representation of static driving environments. Such descriptions may be sufficient for road boundaries but not for less structured areas. Another negative property is that feature maps cannot represent free space explicitly. This is, however, considered a particularly important criterion for future CAS because these systems need to distinguish between areas that have just not yet been observed and areas with real free space evidence to plan safe maneuvers. Although feature-based maps do not suffer from discretization effects and are very compact, the mentioned limitations outweigh the advantages. Hence, these types of maps are hardly suitable for future ADAS on their own. However, the common object lists, which belong to the class of object maps, remain beneficial for object-based tracking of dynamic entities such as other vehicles, e.g. as realized in [21, 64, 154, 170, 177, 201, 240], because these can be compactly described by predefined, basic geometric shapes.

2D interval maps [267] (Fig. 2.1b) are a more general representation than feature maps. The space around the vehicle is discretized in longitudinal direction, while the lateral components are stored as continuous values in the form of point and interval cells. The rationale behind this approach is the observation that ADAS functions for longitudinal traffic require a higher perception accuracy in lateral than in longitudinal direction [267]. In [41], interval maps are used to compactly represent free space in front of the ego vehicle by equidistant intervals or rectangles, respectively. An interval map-based precrash function is further demonstrated in [267]. 2D interval maps can be built quickly and require low bandwidth but have so far only been directly mapped via precise laser scanners.[1] Moreover, the dependence of the data structure on the vehicle orientation may complicate subsequent environment recognition and interpretation algorithms such as automatic junction or roundabout detections.

2D occupancy grid maps (Fig. 2.1c) belong to the next lower abstraction level. Here, the environment is tessellated into a finite number of cells, each of which is assigned a probability of being occupied, depending on the sensor readings [76]. Occupancy grid maps do not have to cope with the correspondence problem, can be constructed with limited computational resources by all automotive environment sensors including radars, lidars, or stereo cameras, and are able to handle free space information explicitly. On the downside, they suffer from high bandwidth and memory require-

[1] Alternatively, the interval-based representation can also be extracted from occupancy grid map-based representations as described in [40].

ments [92] in their standard form, which might limit their application in near-future ADAS due to the slow data transfer over low data rate busses or vehicle-to-vehicle communication interfaces. These high bandwidth requirements can, however, be reduced by altering the data structure of the grid with the aid of compression techniques. Examples include the use of well-known hierarchical structures such as quadtrees, as realized in [134], or the approach described in [91, 92], in which a coarse cell quantization is combined with a run-length encoding of temporal difference grids for removing spatial and temporal redundancies during grid transmission. Independent of the bandwidth requirements, occupancy grids suffer from discretization effects, contain a large amount of irrelevant details such as unreachable free space areas, and are not directly suitable for representing dynamic environments as the conventional mapping assumes a static world. To cope with the latter issue, hybrid environment models that combine grid-based representations of static parts with sparse, object-based representations of dynamic objects can be employed [42, 73, 92, 104, 186]. Alternatively, the grid cells' state space can be extended by velocity components [57].[2] In recent times, grid-based representations have increasingly been used in the ADAS domain, e.g. for realizing CAS with an incorporation of available free space [205], lateral vehicle guidance [155], grid map-based road boundary and road course estimations [65,107,131,243,269], an overtaking assistant [105], precise localization by matching a digital road map to the grid [132], and parking space detection [212].

2.5D maps offer an even more detailed representation of the driving environment, which, however, cannot be created by all sensors employed in the ADAS domain, e.g. not by automotive radars. The simplest version of 2.5D maps are *elevation maps* (Fig. 2.1d), which store the height of the terrain surface in cells of a discrete grid [96]. Elevation maps, however, lack the ability to represent vertical or overhanging structures such as bridges [49] and do not provide a robust temporal filtering in their standard form.[3] They have been used as an intermediate step for stereo camera-based grid-mapping [183], for the detection of road surfaces and obstacles [188], and for the generation of dynamic polygonal maps [257].

The so-called *stixel world* [194] (Fig. 2.1e) provides another intermediate

[2] This topic is picked up again in Section 4, in which related work on mapping dynamic environments is discussed in detail and a novel, computationally efficient method for grid mapping in dynamic environments is outlined.

[3] Very recently, a promising, but computationally demanding, particle-based solution for tracking elevation maps has been presented [61], which could enable their use in dynamic environments.

level 2.5D representation of the driving environment. Vertical surfaces in front of the ego vehicle are approximated by adjacent rectangular sticks of a certain width and height but without depth information. The stixel world can be generated in two different ways. One option is to construct an occupancy grid for an initial free space computation by dynamic programming first, and subsequently estimate the height of the stixels that limit the detected free space [12]. The other option is to perform one single global optimization step as explained in [195]. The stixel world can handle dynamic environments, but the result is not a real map because only stixel information of the current frame (scan) is kept in an ego vehicle-fixed coordinate system. Moreover, free space is only represented implicitly.

Another 2.5D representation, the *multi-level surface map* (Fig. 2.1f), has been introduced in [249] as an extension of elevation maps. The cells of multi-level surface maps contain a list of so-called surface patches, each of which reflect the possibility to traverse the 3D environment at a specific height. Multi-level surface maps represent 3D structures as height values over a grid in analogy to elevation maps but additionally allow for the storage of more than one vertical structure. Consequently, overhanging structures such as bridges or tunnels can be represented correctly. However, only positive sensor data of detected objects is recorded, and occupancy values can thus never be decreased. This property makes them unsuitable for dynamic road environments because obsolete moving object information is never deleted. Moreover, unknown areas are not represented in this kind of map.

In contrast to the mentioned 2.5D representations, full 3D representations can describe arbitrarily shaped three-dimensional objects. Prominent examples include *voxel grids* (Fig. 2.1g) as straightforward extensions of two-dimensional cell representations to the height dimension. Methods for constructing voxel grids can be subdivided into i) truly probabilistic ones, which have a 3D occupancy grid map as a result [108, 175, 213], and ii) methods that just employ a voxel grid to coarsen the level of detail of 3D point clouds. The latter is, for example, used in [47] as an intermediate step for stereo vision-based obstacle detection and in [123] for the detection of dynamic objects based on 3D warping. Full probabilistic 3D grid maps are computationally expensive to construct and require high bandwidth for transmission even if memory-efficient, hierarchical data structures such as octrees are employed as in [108, 213]. This limits their applicability for near-future ADAS, in which the environment representation has to be transferred between many electronic control units. Voxel methods of the second category produce voxel grids faster but lack the explicit description of free space, do not use probabilistic sensor models, and do not provide a way of fusing sensor information.

Meshes (Fig. 2.1h), as another example of full 3D representations, can encode any combination of surfaces and are reducible to a relatively compact form by mesh simplification algorithms [49]. Their main limitations lie in the correct extraction of surfaces from raw data as well as in the robust detection of discontinuities [49], which are common in road environments, e.g. in form of other vehicles or buildings. Moreover, they are incapable of handling dynamic changes in the environment. To date, meshes do not play any role in the ADAS domain.

Raw sensor data models such as 3D point sets (clouds) (Fig. 2.1i) have the lowest abstraction level. For future ADAS, they are not suitable because of the high required bandwidth that grows with each new scan, the difficult interpretation, and the strong dependence on the sensor modality [55]. Moreover, raw sensor data models do not provide a representation of free and unknown space either.

Table 2.1 summarizes the properties of the discussed metric environment models, namely their level of detail (L), performance (P), compactness (C), free space describability (F), capability of handling and describing moving objects (M), sensor independence (I), and represented dimension (D). The properties of the proposed representation, which is introduced in the next section, are additionally already shown in the last row for completeness.

Table 2.1: Comparison of metric environment representations.
−− very low; − low; 0 medium; + high; ++ very high

Representation	L	P	C	F	M	I	D
Landmark-based map	−−	++	++	−	+	+	2
Line map	−	+	++	−	0	+	2
Object map	−	+	++	−	++	+	2-3
Interval map	−	++	++	0	+	++	2
Occ. grid map (2D)	0	+	0	+	0	++	2
Elevation map	+	0	0	−	−	−	2.5
Stixel world	+	0	+	0	+	−	2.5
Multi-level surface map	++	−	0	−	−	−	2.5
Occ. grid map (3D)	++	−−	−−	+	0	−	3
Mesh	++	−−	0	−	−−	−	3
Raw sensor data	++	++	−−	−	−	−−	2-3
Proposed representation	0	+	++	+	++	++	2

Conclusion

Different common environment representations have been discussed with respect to their suitability for future ADAS. In conclusion, *sparse* dynamic object maps for the representation of dynamic entities, in combination with the information contained in *dense* 2D grid maps seem to provide a sufficient level of detail to realize a large variety of future ADAS. For example, lateral vehicle guidance with respect to obstacles, or evasive steering assists that consider drivable free space, can be accomplished with such a combination as they do not require 2.5D or 3D information.[4] However, in the eyes of the author, the information contained in these grid maps is not represented in its optimal, *compact* form for transmission and interpretation and is linked too strongly to the discrete cell representation. Therefore, a compact, continuous, parametric description is proposed also for arbitrary static driving environments, which additionally facilitates environment interpretation by suppressing irrelevant details.

2.3 Proposed Representation

The proposed metric environment representation for near-future ADAS is illustrated on an exemplary driving scene in Fig. 2.2. Excerpts have already been published by the author in [223, 226].

The representation consists of a combination of *dynamic object maps* for representing dynamic entities with so-called *Parametric Free Space (PFS) maps* for encoding relevant static parts of the environment. This separation is beneficial for subsequent situation interpretation algorithms that need to treat static and dynamic objects in a different way. The dynamic object map (white) just contains the oncoming vehicle abstracted by a bounding rectangle with an estimated state vector. The PFS map, in contrast, is a continuous, 2D bird's-eye view representation of the local static environment around the ego vehicle, which does not model the world by discrete cells but by a combination of a parametric curve and geometric primitives. An important difference to other parametric maps, such as object maps, is that not objects of the environment are described explicitly but rather the opposite – the absence of objects in form of *free space.* Free space is defined as the whole drivable space for the vehicle in this context,

[4] Note that there might be some future ADAS functions for which a 2D representation is not sufficient, e.g. for driver warnings if a bridge is too low to pass safely for trucks. However, even in these cases one could argue that a 2D representation that models areas under the bridge as obstacles is acceptable.

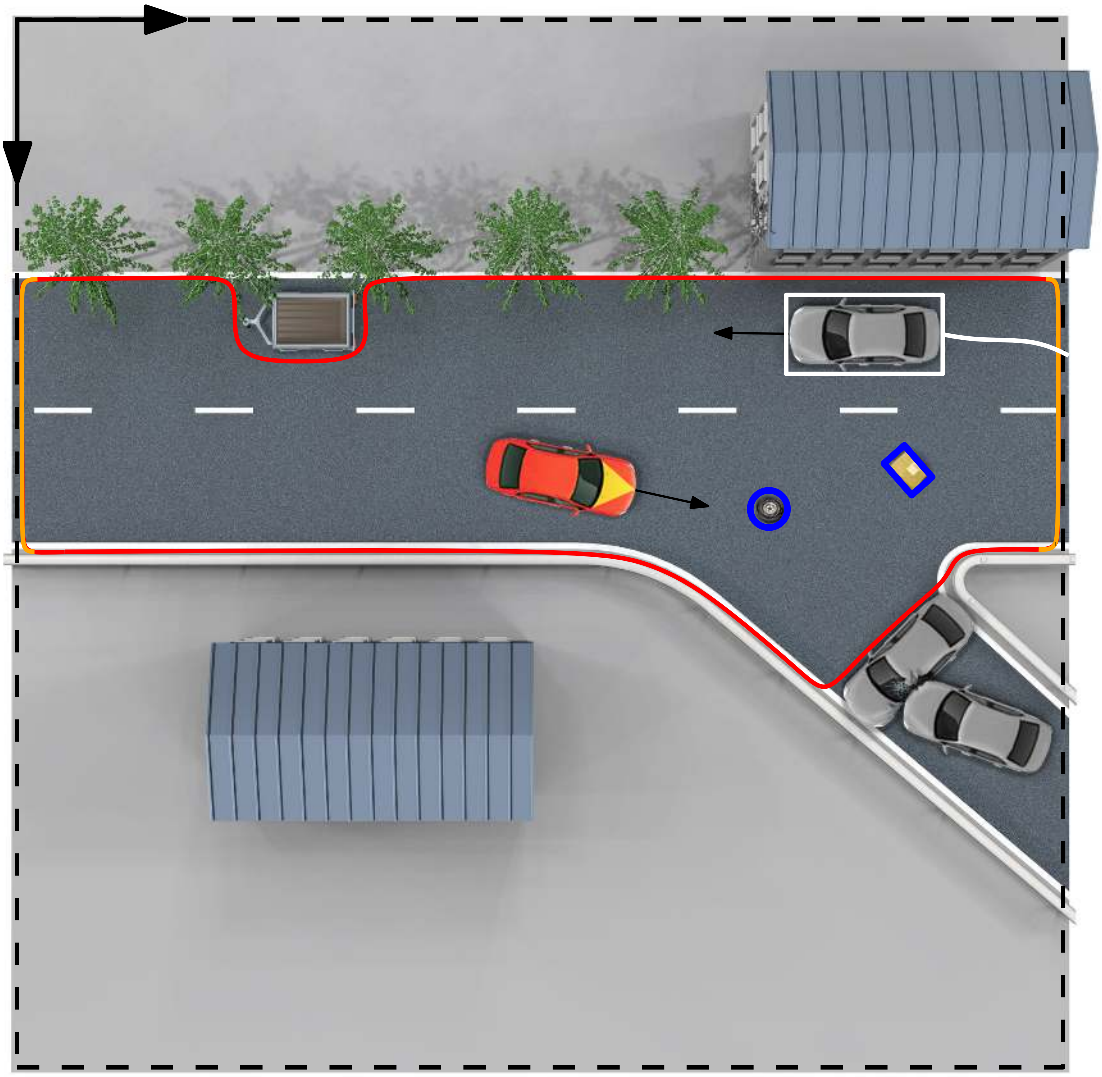


Figure 2.2: Proposed environment representation within an exemplary driving scene with a tree-lined road, a parked trailer, an oncoming vehicle, a blocked right turning, a guardrail, a tire, a box, two houses, and the red ego vehicle in the middle. The dynamic object map is shown in white, the novel PFS map description of the static environment is visualized in red, orange, and blue.

independent of the current lane. The motivation for this free space description is twofold. First, objects in the driving environment are mostly not visible from all sides because the ego vehicle's environment sensors can normally only detect the parts that lie directly in the line of sight of the sensor. Since the driving environment contains too many unknowns to make adequate assumptions about the hidden parts of the objects[5], they

[5] Note that this especially holds for arbitrary static objects. Moving objects can usually be assumed to have a specific shape such as a bounding box.

cannot be described by object maps. Second, an explicit representation of detected free space is important for safety-related trajectory planning.

As the outer boundary of the free space itself can be arbitrarily shaped, a flexible, closed curve is chosen to describe it in a parametric form. The advantage of a closed curve is that scenarios with multiple obstacles all-around the ego vehicle, such as crowded parking lots, can also be treated. If the sensors detect a significant amount of free space all-around an object, then this inner free space boundary – which is greater or equal to the outer object boundary – is represented by simple geometric primitives, e.g. by bounding rectangles or circles. This has proven to be sufficient because inner objects that render such a mapping possible are normally small compared to the large outer free space boundary. In the unlikely event that free space is detected all-around an unstructured object of large dimensions, the PFS map will just contain a little less free space than actually available and is thus conservative in this regard. Furthermore, PFS maps only describe *relevant* free space not merely to achieve an even more compact representation but rather to facilitate situation interpretation tasks. Free areas are considered as irrelevant if they are unreachable or smaller than a vehicle's size because future ADAS will most likely not need this specific information.

In addition to the metric description, location-related semantic information, which encodes supplemental knowledge, is attached to the free space boundary. Examples include the description of the area behind the free space with the corresponding labels "obstacle boundary" or "unknown environment boundary". In Fig. 2.2, the parametric curve of the PFS map encloses the available free space that is delimited by either obstacle boundaries (red) or boundaries to unknown environment (orange). Small passages between narrowly spaced trees as well as the trailer that is parked in front of them belong to the continuous outer free space curve. This also holds for the guardrails on the right-hand side of the road and the two crashed vehicles that block the right turn. In contrast, the two small obstacles on the road – a tire and a box – are captured by the geometric primitives description (blue) because sufficient free space is available and detected around them. Note that the house behind the guardrail is not included in the PFS map because it is located behind the outer free space boundary and therefore unreachable for the ego vehicle.

The integration of the two maps (PFS map and dynamic object map) into the world model is illustrated in Fig. 2.3. In combination, they provide the metric representation that is extended by additional environment information such as drivable lanes, traffic signs, and topological maps to

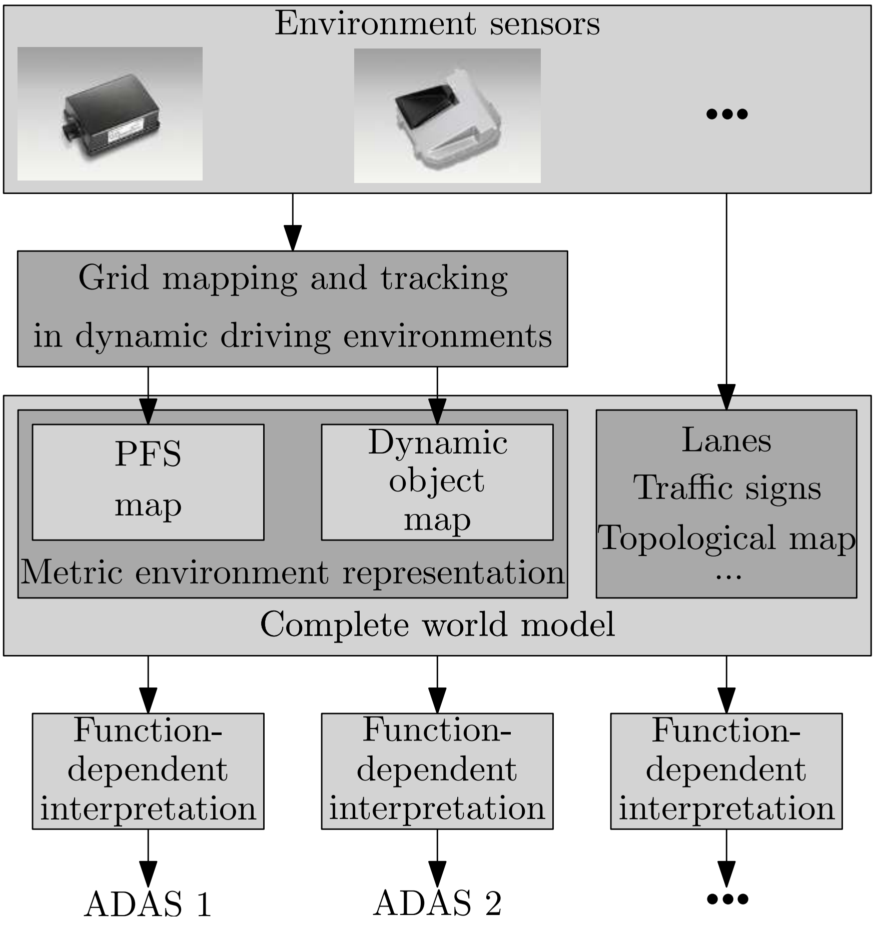


Figure 2.3: Integration of PFS and dynamic object maps in the world model.

form the complete world model, which decouples the ADAS function from the specific sensor configuration. This world model is subsequently interpreted by function-dependent situation interpretation algorithms for different ADAS.

Due to the mentioned advantages and the widespread usage of grid map-based representations, a method is presented in this thesis that obtains both the dynamic object map and the PFS map in a consistent way by first building a grid map in dynamic driving environments as an intermediate representation. This intermediate grid decouples subsequent steps from specific sensor configurations. Moreover, the consistency between static and dynamic world is always ensured with this approach due to the common underlying grid.

In order to realize the representation, various variants of Bayesian inference are employed throughout this work, which include different mapping and tracking filters. Therefore, all relevant theoretical foundations on this topic are first presented in the following chapter.

3 Bayesian Inference for Nonlinear Filtering Tasks

Bayesian inference is a theoretically well-founded method for reasoning under uncertainty. The following sections first provide the theoretical foundations of Bayesian filters as a special case of Bayesian inference in dynamic systems. These filters form the basis of two filtering applications, which play a major role in the generation of the proposed environment representation: target tracking and occupancy grid mapping – both presented subsequently.

3.1 Bayesian Filtering

Starting with the derivation of the optimal Bayesian filter, this section transitions to the specification of the filter components by means of stochastic state space systems and the extraction of point estimates from the filter solution. Thereafter, all later-employed filter algorithms, which realize the optimal filter solution in a computationally tractable way, are presented.

3.1.1 Optimal Bayesian Filter

A general form of Bayesian inference in dynamic systems is concerned with the calculation of the posterior *probability density function* (pdf) $p(\boldsymbol{x}_{0:k}|\boldsymbol{z}_{1:k})$, also called conditional distribution or belief, of a hidden state sequence $\boldsymbol{x}_{0:k} = \{\boldsymbol{x}_0, \ldots, \boldsymbol{x}_{k-1}, \boldsymbol{x}_k\}$ given an observed measurement sequence $\boldsymbol{z}_{1:k} = \{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_{k-1}, \boldsymbol{z}_k\}$. In this pdf, the complete probabilistic knowledge about the state is subsumed. If a relationship between states and measurements is known and prior knowledge about states is available in form of a prior distribution $p(\boldsymbol{x}_{0:k})$, then Bayes' theorem[1] [26] allows to

[1] Thomas Bayes formulated the theory only for uniform priors. Pierre-Simon Laplace independently rediscovered the theory in the nowadays known general form and published it in his famous essay [139].

calculate the posterior pdf according to

$$p(\boldsymbol{x}_{0:k}|\boldsymbol{z}_{1:k}) = \frac{\overbrace{p(\boldsymbol{z}_{1:k}|\boldsymbol{x}_{0:k})}^{\text{Likelihood}}\,\overbrace{p(\boldsymbol{x}_{0:k})}^{\text{Prior}}}{\underbrace{p(\boldsymbol{z}_{1:k})}_{\text{Normalization}}}. \tag{3.1}$$

The likelihood function[2] determines the probability that the observed measurement sequence is $\boldsymbol{z}_{1:k}$ given that the hidden state sequence is $\boldsymbol{x}_{0:k}$, whereas the denominator contains a normalization factor, which ensures that $p(\boldsymbol{x}_{0:k}|\boldsymbol{z}_{1:k})$ is a valid pdf and integrates to one. The normalization can be calculated from the numerator by marginalization[3], therefore

$$p(\boldsymbol{z}_{1:k}) = \int_{\boldsymbol{x}_k} \dots \int_{\boldsymbol{x}_0} p(\boldsymbol{z}_{1:k}|\boldsymbol{x}_{0:k}) p(\boldsymbol{x}_{0:k}) \mathrm{d}\boldsymbol{x}_0 \dots \mathrm{d}\boldsymbol{x}_k. \tag{3.2}$$

In online inference, measurements are received one by one over time. Consequently, a recursive solution is needed that updates the state distribution at each time step. For this purpose, the elements in (3.1) are rewritten in terms of the following factorizations

$$p(\boldsymbol{z}_{1:k}|\boldsymbol{x}_{0:k}) = p(\boldsymbol{z}_k|\boldsymbol{z}_{1:k-1}, \boldsymbol{x}_{0:k}) p(\boldsymbol{z}_{1:k-1}|\boldsymbol{x}_{0:k}), \tag{3.3a}$$

$$p(\boldsymbol{x}_{0:k}) = p(\boldsymbol{x}_k|\boldsymbol{x}_{0:k-1}) p(\boldsymbol{x}_{0:k-1}), \tag{3.3b}$$

$$p(\boldsymbol{z}_{1:k}) = p(\boldsymbol{z}_k|\boldsymbol{z}_{1:k-1}) p(\boldsymbol{z}_{1:k-1}). \tag{3.3c}$$

As measurements at time $k-1$ do not depend on object states at later times, the factorization of the likelihood function (3.3a) is simplified to

$$p(\boldsymbol{z}_{1:k}|\boldsymbol{x}_{0:k}) = p(\boldsymbol{z}_k|\boldsymbol{z}_{1:k-1}, \boldsymbol{x}_{0:k}) p(\boldsymbol{z}_{1:k-1}|\boldsymbol{x}_{0:k-1}). \tag{3.4}$$

Substituting these factorizations into (3.1) and rearranging yields the general recursive form

$$p(\boldsymbol{x}_{0:k}|\boldsymbol{z}_{1:k}) = \frac{p(\boldsymbol{z}_k|\boldsymbol{z}_{1:k-1}, \boldsymbol{x}_{0:k}) p(\boldsymbol{x}_k|\boldsymbol{x}_{0:k-1})}{p(\boldsymbol{z}_k|\boldsymbol{z}_{1:k-1})} \underbrace{\frac{p(\boldsymbol{z}_{1:k-1}|\boldsymbol{x}_{0:k-1}) p(\boldsymbol{x}_{0:k-1})}{p(\boldsymbol{z}_{1:k-1})}}_{p(\boldsymbol{x}_{0:k-1}|\boldsymbol{z}_{1:k-1})}. \tag{3.5}$$

[2] The term likelihood function for the pdf $p(\boldsymbol{z}_{1:k}|\boldsymbol{x}_{0:k})$ is only used if it is viewed as a function of $\boldsymbol{x}_{0:k}$ for a specific, fixed $\boldsymbol{z}_{1:k}$ instead of the traditional way as a function of $\boldsymbol{z}_{1:k}$. Note that it is not a pdf over $\boldsymbol{x}_{0:k}$ under this point of view as the likelihood function does not integrate to unity to make it a valid pdf.

[3] For better readability, $\int_{\boldsymbol{x}_k} (\cdot) \mathrm{d}\boldsymbol{x}_k$ is the integration over the whole range of $\boldsymbol{x}_k$, e.g. if $\boldsymbol{x}_k \in \mathbb{R}^n$, then $\int_{\boldsymbol{x}_k} (\cdot) \mathrm{d}\boldsymbol{x}_k = \int_{\mathbb{R}^n} (\cdot) \mathrm{d}\boldsymbol{x}_k$.

The second factor in (3.5) is, according to a second application of Bayes' theorem, equal to $p(\boldsymbol{x}_{0:k-1}|\boldsymbol{z}_{1:k-1})$.

For the design of computationally feasible filters, two simplifying assumptions are common. First, measurements at a given time are assumed to depend only on states of this time. They are conditionally independent of previous measurement and state histories and thus $p(\boldsymbol{z}_k|\boldsymbol{z}_{1:k-1}, \boldsymbol{x}_{0:k})$ simplifies to $p(\boldsymbol{z}_k|\boldsymbol{x}_k)$. Second, the system to be filtered is supposed to obey the Markov assumption, also called the complete state assumption [247], which means that the present state only depends on the previous state. The present state $\boldsymbol{x}_k$ is thus conditionally independent of the older state sequence $\boldsymbol{x}_{0:k-2}$ given the last state $\boldsymbol{x}_{k-1}$. Consequently, $p(\boldsymbol{x}_k|\boldsymbol{x}_{0:k-1}) = p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$ holds.[4] These conditional independence assumptions are visualized in form of a *Dynamic Bayesian Network* (DBN) in Fig. 3.1, which, in this particular form, is known as a first-order *Hidden Markov Model*[5] (HMM) [203].

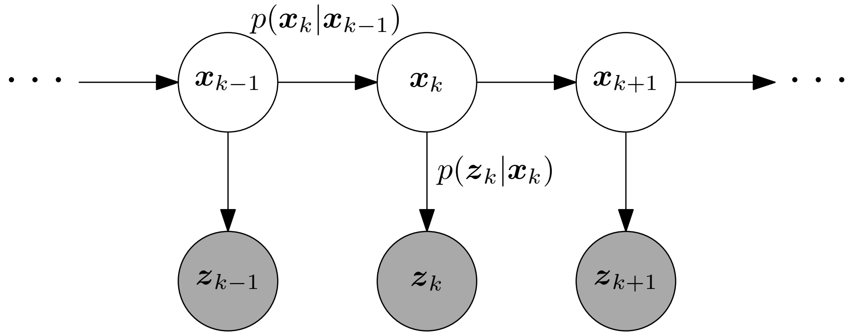


Figure 3.1: First-order HMM with observable nodes shaded in gray. In filtering tasks, measurements are only available up to time step k.

With these assumptions, (3.5) can be formulated as

$$p(\boldsymbol{x}_{0:k}|\boldsymbol{z}_{1:k}) = \frac{p(\boldsymbol{z}_k|\boldsymbol{x}_k)}{p(\boldsymbol{z}_k|\boldsymbol{z}_{1:k-1})} p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}) p(\boldsymbol{x}_{0:k-1}|\boldsymbol{z}_{1:k-1}). \tag{3.6}$$

In Bayesian optimal *filtering*, however, not the full joint posterior density $p(\boldsymbol{x}_{0:k}|\boldsymbol{z}_{1:k})$ is of interest but the marginal posterior state density

[4] This assumption is less of a restriction than it first sounds because a Markov process of arbitrary order can always be transformed into a vector valued first-order Markov process similar to that higher-order differential equations can always be reformulated as first-order vector differential equations.

[5] Some authors, e.g. [178, 210], use the term HMM only for discrete state spaces.

$p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k})$ at the current time step k, known as the filtering distribution.[6] This filtering distribution can be derived from (3.6) by marginalization over all states of all previous time steps [53], therefore

$$\begin{aligned}
p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}) &= \int_{\boldsymbol{x}_{k-1}} \dots \int_{\boldsymbol{x}_0} p(\boldsymbol{x}_{0:k}|\boldsymbol{z}_{1:k}) \mathrm{d}\boldsymbol{x}_0 \dots \mathrm{d}\boldsymbol{x}_{k-1} \\
&= \frac{p(\boldsymbol{z}_k|\boldsymbol{x}_k)}{p(\boldsymbol{z}_k|\boldsymbol{z}_{1:k-1})} \int_{\boldsymbol{x}_{k-1}} \dots \int_{\boldsymbol{x}_0} p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}) p(\boldsymbol{x}_{0:k-1}|\boldsymbol{z}_{1:k-1}) \mathrm{d}\boldsymbol{x}_0 \dots \mathrm{d}\boldsymbol{x}_{k-1} \\
&= \frac{p(\boldsymbol{z}_k|\boldsymbol{x}_k)}{p(\boldsymbol{z}_k|\boldsymbol{z}_{1:k-1})} \\
&\quad \int_{\boldsymbol{x}_{k-1}} \left(p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}) \underbrace{\int_{\boldsymbol{x}_{k-2}} \dots \int_{\boldsymbol{x}_0} p(\boldsymbol{x}_{0:k-1}|\boldsymbol{z}_{1:k-1}) \mathrm{d}\boldsymbol{x}_0 \dots \mathrm{d}\boldsymbol{x}_{k-2}}_{p(\boldsymbol{x}_{k-1}|\boldsymbol{z}_{1:k-1})} \right) \mathrm{d}\boldsymbol{x}_{k-1} .
\end{aligned} \tag{3.7}$$

Since $p(\boldsymbol{x}_{0:k-1}|\boldsymbol{z}_{1:k-1}) = p(\boldsymbol{x}_{k-1}, \boldsymbol{x}_{0:k-2}|\boldsymbol{z}_{1:k-1})$, the inner integrals over $\boldsymbol{x}_0$ through $\boldsymbol{x}_{k-2}$ simplify to $p(\boldsymbol{x}_{k-1}|\boldsymbol{z}_{1:k-1})$ via marginalization and the recursive Bayesian filtering solution, the posterior state pdf or filtering distribution, is finally given by [18, 53, 208]

$$p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}) = \frac{\overbrace{p(\boldsymbol{z}_k|\boldsymbol{x}_k)}^{\text{Measurement Likelihood}}}{\underbrace{p(\boldsymbol{z}_k|\boldsymbol{z}_{1:k-1})}_{\text{Normalization}}} \overbrace{\underbrace{\int_{\boldsymbol{x}_{k-1}} p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}) p(\boldsymbol{x}_{k-1}|\boldsymbol{z}_{1:k-1}) \mathrm{d}\boldsymbol{x}_{k-1}}_{p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1})}}^{\text{Prior (Chapman-Kolmogorov)}} . \tag{3.8}$$

It is this formula that all Bayesian filtering and tracking algorithms try to compute. It can be interpreted as combining a prediction or time update step with a correction or measurement update step. The prediction step is realized by the Chapman-Kolmogorov equation that yields a predicted, prior state pdf $p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1})$ for time step k by multiplying the posterior state pdf of the previous time step $p(\boldsymbol{x}_{k-1}|\boldsymbol{z}_{1:k-1})$ with the transition density $p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$ and integrating over all states $\boldsymbol{x}_{k-1}$ of this previous time step. This prior thus includes the knowledge of all measurements up

[6] In opposition to filtering distributions, prediction distributions $p(\boldsymbol{x}_{k+m}|\boldsymbol{z}_{1:k})$ are marginal distributions of future states, $m \in \mathbb{N}$ steps after the current time step k, whereas smoothing distributions are marginal distributions of the form $p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k+m})$ and thus calculated given measurements beyond the time k for which the state distribution is searched.

to time $k-1$. The subsequent correction step uses the current measurement $\boldsymbol{z}_k$ to correct this prior at time step k via the measurement likelihood function $p(\boldsymbol{z}_k|\boldsymbol{x}_k)$. After normalization, the posterior state pdf $p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k})$ is available. It remains to specify the transition density $p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$, the measurement likelihood $p(\boldsymbol{z}_k|\boldsymbol{x}_k)$, and the initial state pdf $p(\boldsymbol{x}_0)$ to start the recursion.

Stochastic State Space Systems

In a general sense, this specification can be realized in form of a discrete-time, stochastic state space system

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k) + \boldsymbol{w}_k, \tag{3.9a}$$

$$\boldsymbol{z}_k = \boldsymbol{h}(\boldsymbol{x}_k) + \boldsymbol{v}_k, \tag{3.9b}$$

with input vector $\boldsymbol{u}_k \in \mathbb{R}^p$, state vector $\boldsymbol{x}_k \in \mathbb{R}^n$, measurement vector $\boldsymbol{z}_k \in \mathbb{R}^q$, process noise vector $\boldsymbol{w}_k \in \mathbb{R}^n$, measurement noise vector $\boldsymbol{v}_k \in \mathbb{R}^q$, system function $\boldsymbol{f}(\cdot)$ and measurement function $\boldsymbol{h}(\cdot)$ at each time step k. The noise sequences are mutually independent, zero-mean, and white[7], with process noise covariance matrix $\boldsymbol{Q}_k \in \mathbb{R}^{n\times n}$ and measurement noise covariance matrix $\boldsymbol{R}_k \in \mathbb{R}^{q\times q}$. The transition density is induced by the system dynamic equation (3.9a), while the measurement likelihood is generated from the measurement equation (3.9b) according to [53, 166]

$$p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}) = p_{\boldsymbol{w}_{k-1}}\big(\boldsymbol{x}_k - \boldsymbol{f}(\boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1})\big), \tag{3.10a}$$

$$p(\boldsymbol{z}_k|\boldsymbol{x}_k) = p_{\boldsymbol{v}_k}\big(\boldsymbol{z}_k - \boldsymbol{h}(\boldsymbol{x}_k)\big), \tag{3.10b}$$

in which $p_{\boldsymbol{w}_{k-1}}(\cdot)$ and $p_{\boldsymbol{v}_k}(\cdot)$ are the pdfs of the corresponding process and measurement noise vectors, respectively.

In the case of linear time-invariant systems, (3.9) reduces to

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{u}_k + \boldsymbol{w}_k, \tag{3.11a}$$

$$\boldsymbol{z}_k = \boldsymbol{H}\boldsymbol{x}_k + \boldsymbol{v}_k, \tag{3.11b}$$

with system matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, input matrix $\boldsymbol{B} \in \mathbb{R}^{n\times p}$, and measurement matrix $\boldsymbol{H} \in \mathbb{R}^{q\times n}$. With the additional assumption of Gaussian noise

[7] This property ensures that we have a Markov process.

sequences, the probabilistic counterpart of (3.11) is then given by[8] [30]

$$p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}) = \mathcal{N}(\boldsymbol{x}_k; \boldsymbol{A}\boldsymbol{x}_{k-1} + \boldsymbol{B}\boldsymbol{u}_{k-1}, \boldsymbol{Q}_{k-1}), \tag{3.12a}$$

$$p(\boldsymbol{z}_k|\boldsymbol{x}_k) = \mathcal{N}(\boldsymbol{z}_k; \boldsymbol{H}\boldsymbol{x}_k, \boldsymbol{R}_k). \tag{3.12b}$$

Measurement equations are normally directly specified in discrete-time. System dynamic equations, in contrast, are often more conveniently derived and given in the continuous-time form[9]

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t), t) + \boldsymbol{w}_{\mathrm{c}}(t), \quad \boldsymbol{x}(t_0) = \boldsymbol{x}_0, \tag{3.13}$$

or via

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}_{\mathrm{c}}\boldsymbol{x}(t) + \boldsymbol{B}_{\mathrm{c}}\boldsymbol{u}(t) + \boldsymbol{w}_{\mathrm{c}}(t), \quad \boldsymbol{x}(t_0) = \boldsymbol{x}_0, \tag{3.14}$$

in the case of linear time-invariant systems with continuous-time system matrix $\boldsymbol{A}_{\mathrm{c}} \in \mathbb{R}^{n\times n}$, continuous-time input matrix $\boldsymbol{B}_{\mathrm{c}} \in \mathbb{R}^{n\times p}$, and continuous-time, zero-mean, white noise process $\boldsymbol{w}_{\mathrm{c}}(t) \in \mathbb{R}^n$ with noise intensity matrix[10] $\boldsymbol{Q}_{\mathrm{n}}(t) \in \mathbb{R}^{n\times n}$.

Continuous-time systems like these have to be discretized during the filter design and implementation process. While (3.13) can only be discretized exactly in special cases and, therefore, often approximate schemes have to be employed [2], it is always possible to discretize (3.14) exactly. Assuming $\boldsymbol{u}(t)$ to be constant during each sampling interval (zero-order

[8] The notation $\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{P})$ is used for a multivariate Gaussian pdf of a random vector $\boldsymbol{x}$ with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{P}$ with $\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{P}) = \frac{1}{\sqrt{|2\pi\boldsymbol{P}|}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\boldsymbol{P}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})}$ [208].

[9] From a formal point of view, stochastic differential equations are to be written as $\mathrm{d}\boldsymbol{x}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t), t)\mathrm{d}t + \mathrm{d}\boldsymbol{W}(t)$, $\boldsymbol{x}(t_0) = \boldsymbol{x}_0$, with continuous-time Wiener process $\boldsymbol{W}(t)$, the integral of white noise, as the time-derivative $\frac{\mathrm{d}\boldsymbol{x}(t)}{\mathrm{d}t}$ may not exist [152, 187]. This is due to the non-existent derivative of the nowhere differentiable Wiener process. Another correct way to write (3.13) is in form of the corresponding stochastic integral equation $\boldsymbol{x}(t) = \boldsymbol{x}_0 + \int_0^t \boldsymbol{f}(\boldsymbol{x}(\tau), \boldsymbol{u}(\tau), \tau)\mathrm{d}\tau + \int_0^t \mathrm{d}\boldsymbol{W}(\tau)$, in which the second integral is a stochastic Itô integral. The given notation is, however, common in the engineering literature so that we sacrifice rigor for readability.

[10] It should be emphasized that $\boldsymbol{Q}_{\mathrm{n}}(t)$ is a noise intensity matrix and not a covariance matrix as often falsely claimed. The corresponding covariance (function) matrix $\boldsymbol{Q}_{\mathrm{c}}(t, \tau)$ of the (non-existent) continuous-time, zero-mean, white noise process is formally given by $\boldsymbol{Q}_{\mathrm{c}}(t, \tau) = \mathbb{E}(\boldsymbol{w}_{\mathrm{c}}(t)\boldsymbol{w}_{\mathrm{c}}^T(\tau)) = \boldsymbol{Q}_{\mathrm{n}}(t)\delta(t-\tau)$ and differs from $\boldsymbol{Q}_{\mathrm{n}}(t)$ also in unit as the Dirac delta distribution $\delta(t-\tau)$ has unit $\frac{1}{\mathrm{s}}$, the inverse of time [87]. If the noise intensity matrix $\boldsymbol{Q}_{\mathrm{n}}(t)$ is constant over time (stationary white noise), it is termed the *Power Spectral Density* (PSD) matrix. Note that continuous-time, white noise is just a useful mathematical concept. It can never be generated exactly in reality due to its infinite power in contrast to discrete-time white noise sequences.

hold) and introducing $\boldsymbol{u}_k = \boldsymbol{u}(t_0 + kT)$ and $\boldsymbol{x}_k = \boldsymbol{x}(t_0 + kT)$ with $k \in \mathbb{N}^0$ and sampling time T, then the matrices of the discrete-time equivalent (3.11a) of (3.14) are given by the relations

$$\boldsymbol{A} = e^{\boldsymbol{A}_\mathrm{c} T} = \left[\mathcal{L}^{-1}\left\{(\mathbb{1}s - \boldsymbol{A}_\mathrm{c})^{-1}\right\}\right]_{t=T}, \tag{3.15a}$$

$$\boldsymbol{B} = \boldsymbol{B}_\mathrm{c}\int_0^T e^{\boldsymbol{A}_\mathrm{c}\alpha}\mathrm{d}\alpha, \tag{3.15b}$$

with inverse Laplace transformation $\mathcal{L}^{-1}(\cdot)$ and Laplace variable s [161]. The integrated effect of the (stationary) continuous-time, white noise process can exactly be reproduced in discrete-time via the correspondence[11] [81, 87]

$$\boldsymbol{Q}_k = \int_0^T e^{\boldsymbol{A}_\mathrm{c}\alpha}\boldsymbol{Q}_\mathrm{n} e^{\boldsymbol{A}_\mathrm{c}^T\alpha}\mathrm{d}\alpha, \tag{3.16}$$

whereas the approximate discretization

$$\boldsymbol{Q}_k \approx \boldsymbol{Q}_\mathrm{n} T \tag{3.17}$$

yields reasonable results for small – in relation to the system dynamics – sampling times [232].

Bayesian Point Estimates

Subsequent applications often just make use of point estimates instead of the full posterior distribution (3.8). According to Bayesian decision theory [27], different Bayesian point estimates $\hat{\boldsymbol{x}}_k^\mathrm{B}$ can be calculated from $p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k})$. These minimize the expectation of a chosen scalar loss function $L(\boldsymbol{x}_k, \hat{\boldsymbol{x}}_k)$, the Bayesian expected loss, given all measurements [27], therefore

$$\hat{\boldsymbol{x}}_k^\mathrm{B} = \arg\min_{\hat{\boldsymbol{x}}_k} \mathbb{E}(L(\boldsymbol{x}_k, \hat{\boldsymbol{x}}_k)|\boldsymbol{z}_{1:k}) = \arg\min_{\hat{\boldsymbol{x}}_k}\int_{\boldsymbol{x}_k} L(\boldsymbol{x}_k, \hat{\boldsymbol{x}}_k)p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k})\mathrm{d}\boldsymbol{x}_k. \tag{3.18}$$

The quadratic loss function

$$L(\boldsymbol{x}_k, \hat{\boldsymbol{x}}_k) = (\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k)^T(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k) \tag{3.19}$$

[11] Often, (3.16) is given in the equivalent form $\boldsymbol{Q}_k = \int_0^T e^{\boldsymbol{A}_\mathrm{c}(T-\tau)}\boldsymbol{Q}_\mathrm{n} e^{\boldsymbol{A}_\mathrm{c}^T(T-\tau)}\mathrm{d}\tau$.

results in the unbiased[12] *Minimum Mean Square Error* (MMSE) estimate $\hat{\boldsymbol{x}}_k^{\mathrm{M}}$, the posterior mean, given by the conditional expectation[13]

$$\hat{\boldsymbol{x}}_k^{\mathrm{M}} = \mathbb{E}(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}) = \int_{\boldsymbol{x}_k} \boldsymbol{x}_k p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k})\mathrm{d}\boldsymbol{x}_k, \tag{3.20}$$

with associated error covariance matrix

$$\begin{aligned}\boldsymbol{P}_k &= \mathbb{E}\left((\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k^{\mathrm{M}})(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k^{\mathrm{M}})^T|\boldsymbol{z}_{1:k}\right) \\ &= \int_{\boldsymbol{x}_k} \left((\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k^{\mathrm{M}})(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k^{\mathrm{M}})^T\right) p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k})\mathrm{d}\boldsymbol{x}_k.\end{aligned} \tag{3.21}$$

The MMSE estimate is located at the center of mass of the posterior state pdf and is therefore a weighted average of all values of the pdf. Another common loss function is

$$L(\boldsymbol{x}_k, \hat{\boldsymbol{x}}_k) = \begin{cases} 0, & \boldsymbol{x}_k = \hat{\boldsymbol{x}}_k, \\ 1, & \boldsymbol{x}_k \neq \hat{\boldsymbol{x}}_k, \end{cases} \tag{3.22}$$

which results in the *Maximum A Posteriori* (MAP) estimate

$$\hat{\boldsymbol{x}}_k^{\mathrm{MAP}} = \arg\max_{\boldsymbol{x}_k} \left(p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k})\right) \tag{3.23}$$

located at the peak of the posterior state pdf. In the Gaussian case, these estimates coincide, whereas they can substantially differ in multimodal distributions as visualized in Fig. 3.2. In this case, point estimates may be misleading as much information is lost by mapping the complete pdf onto a single vector.

[12] An estimate is unbiased if $\mathbb{E}(\hat{\boldsymbol{x}}) = \mathbb{E}(\boldsymbol{x})$ holds. This is indeed the case for the MMSE estimate given by (3.20) due to $\mathbb{E}(\hat{\boldsymbol{x}}_k^{\mathrm{M}}) = \mathbb{E}(\mathbb{E}(\boldsymbol{x}_k|\boldsymbol{z}_{1:k})) = \mathbb{E}(\boldsymbol{x}_k)$, which directly results from the law of iterated expectations.

[13] This can easily be shown by setting the first derivative of the Bayesian expected loss to zero. Therefore, $\frac{\partial \mathbb{E}(L(\boldsymbol{x}_k,\hat{\boldsymbol{x}}_k)|\boldsymbol{z}_{1:k})}{\partial \hat{\boldsymbol{x}}_k} = \frac{\partial \mathbb{E}(((\boldsymbol{x}_k-\hat{\boldsymbol{x}}_k)^T(\boldsymbol{x}_k-\hat{\boldsymbol{x}}_k))|\boldsymbol{z}_{1:k})}{\partial \hat{\boldsymbol{x}}_k} = \frac{\partial(\mathbb{E}(\boldsymbol{x}_k^T\boldsymbol{x}_k|\boldsymbol{z}_{1:k})-\mathbb{E}(\boldsymbol{x}_k^T|\boldsymbol{z}_{1:k})\hat{\boldsymbol{x}}_k-\hat{\boldsymbol{x}}_k^T\mathbb{E}(\boldsymbol{x}_k|\boldsymbol{z}_{1:k})+\hat{\boldsymbol{x}}_k^T\hat{\boldsymbol{x}}_k)}{\partial \hat{\boldsymbol{x}}_k} = -2\mathbb{E}(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}) + 2\hat{\boldsymbol{x}}_k = \boldsymbol{0}$, which results in $\hat{\boldsymbol{x}}_k^{\mathrm{M}} = \mathbb{E}(\boldsymbol{x}_k|\boldsymbol{z}_{1:k})$. As $\frac{\partial^2\mathbb{E}(L(\boldsymbol{x}_k,\hat{\boldsymbol{x}}_k)|\boldsymbol{z}_{1:k})}{\partial \hat{\boldsymbol{x}}_k^2} = 2\cdot\mathbb{1} \succ \boldsymbol{0}$ holds, this is a minimum.

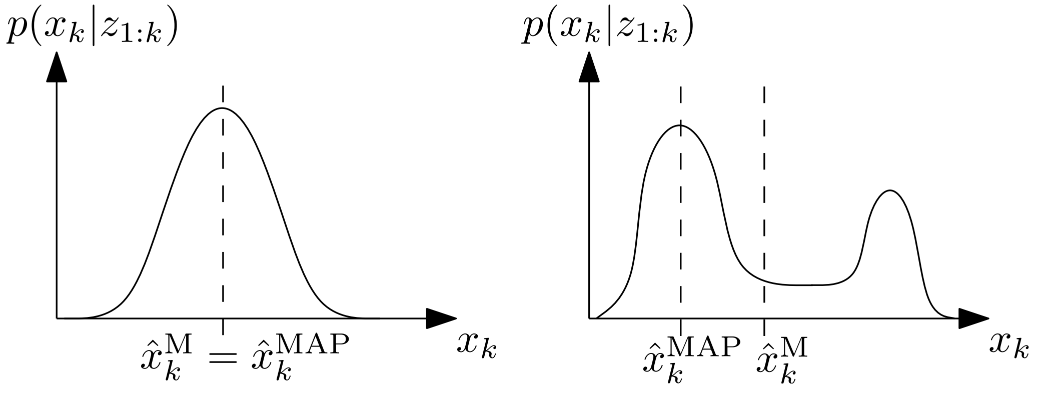


Figure 3.2: MMSE and MAP point estimates on two exemplary posterior densities.

Filter Algorithms

Because a solution of (3.8) is generally not possible in closed form, many kinds of filter algorithms have been developed, which try to solve this equation in different ways.[14]

- *Kalman Filters* (KF) [120] and *Information Filters* (IF) provide exact, closed form solutions under *Linear Gaussian System* (LGS) assumptions.

- *Extended Kalman Filters* (EKF) [6, 18] are based on a Taylor series linearization of the system.

- *Unscented Kalman Filters* (UKF) [118] are based on the unscented transformation to propagate a set of sigma points through the nonlinearities and retrieve a Gaussian representation afterwards.

- *Interacting Multiple Model* (IMM) Filters [36] approximate the posterior distribution by a Gaussian mixture.

- *Generalized Pseudo-Bayesian* (GPB) Filters [18] approximate the posterior distribution by a Gaussian mixture, but are either less accurate (GPB1 Filter) or computationally more demanding (GPB2 Filter) than IMM Filters.

[14] The list is limited to well-known algorithms and not intended to be exhaustive. Filters that go beyond traditional Bayesian solutions such as random set-based solutions and their approximations, e.g. *Cardinalized Probability Hypothesis Density* (CPHD) Filters [165], are not included.

- *Histogram Filters* (HF) [247] approximate the posterior distribution by a discrete distribution over a finite grid, represented as a histogram.
- *Discrete Bayes Filters* (DBF) [247] provide exact inference results for discrete state spaces in form of a *probability mass function* (pmf). They are also termed the forward-pass in discrete HMMs. *Binary Bayes Filters* (BBF) are a special class of DBFs, in which the state space is only composed of two discrete, static states.
- *Particle Filters* (PF) (Sequential Monte Carlo Methods) [8, 69, 90] approximate the posterior distribution by a finite number of particles, which can represent arbitrary, multimodal distributions.
- *Rao-Blackwellized Particle Filters* (RBPF) [68] use Monte Carlo integration for only a subset of state variables while the rest is filtered in parametric form, e.g. with KFs.

In Table 3.1, the mentioned filter algorithms are compared with respect to their posterior representation (PR), whether the filtering is realized in a parametric form (P), whether they can provide exact solutions to (3.8) under additional assumptions (E), and with respect to their processing speed (PS).

As we later use KFs/IFs for spline tracking, a combination of IMMs and UKFs for online tracking and classification of dynamic grid cell hypothesis,

Table 3.1: Bayesian filter algorithms.
−− very low; − low; 0 medium; + high; ++ very high

Filter	PR	P	E	PS
KF/IF	Gaussian pdf	✓	✓(LGS)	++
EKF	Gaussian pdf	✓	✗	+
UKF	Gaussian pdf	✓	✗	+
IMM	Gaussian mixture pdf	✓	✗	0
GPB1	Gaussian mixture pdf	✓	✗	0
GPB2	Gaussian mixture pdf	✓	✗	−
HF	Histogram	✗	✗	−−
DBF	Arbitrary pmf	✗	✓(discrete states)	++
PF	Samples	✗	✗	−−
RBPF	Samples+Gaussian pdf	Partly	✗	−

and BBFs for grid mapping, these filters are presented in more detail in the following subsections.

3.1.2 Kalman Filter

The well-known KF [120] provides an exact filtering solution to (3.8) for the sub-class of linear, but possibly time-variant, stochastic systems of the form

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_k \boldsymbol{x}_k + \boldsymbol{B}_k \boldsymbol{u}_k + \boldsymbol{w}_k, \tag{3.24a}$$

$$\boldsymbol{z}_k = \boldsymbol{H}_k \boldsymbol{x}_k + \boldsymbol{v}_k, \tag{3.24b}$$

with input vector $\boldsymbol{u}_k \in \mathbb{R}^p$, state vector $\boldsymbol{x}_k \in \mathbb{R}^n$, measurement vector $\boldsymbol{z}_k \in \mathbb{R}^q$, process noise vector $\boldsymbol{w}_k \in \mathbb{R}^n$, measurement noise vector $\boldsymbol{v}_k \in \mathbb{R}^q$, system matrix $\boldsymbol{A}_k \in \mathbb{R}^{n\times n}$, input matrix $\boldsymbol{B}_k \in \mathbb{R}^{n\times p}$, and measurement matrix $\boldsymbol{H}_k \in \mathbb{R}^{q\times n}$ under the prerequisite that the noise sequences are mutually independent, zero-mean, white[15], and Gaussian[16], with process noise covariance matrix $\boldsymbol{Q}_k \in \mathbb{R}^{n\times n}$ and measurement noise covariance matrix $\boldsymbol{R}_k \in \mathbb{R}^{q\times q}$, and that the initial state density is Gaussian. Under these assumptions, all occurring pdfs are Gaussian, which can compactly be specified by the first two moments, mean and covariance.

In analogy to the optimal Bayesian filter, the KF can be interpreted in form of a prediction and correction step. First, the a posteriori state estimates of the last time step $\hat{\boldsymbol{x}}_{k-1}$ and associated error covariances $\boldsymbol{P}_{k-1}$ are predicted to the current time step k via

$$\hat{\boldsymbol{x}}_k^- = \boldsymbol{A}_{k-1} \hat{\boldsymbol{x}}_{k-1} + \boldsymbol{B}_{k-1} \boldsymbol{u}_{k-1}, \tag{3.25a}$$

$$\boldsymbol{P}_k^- = \boldsymbol{A}_{k-1} \boldsymbol{P}_{k-1} \boldsymbol{A}_{k-1}^T + \boldsymbol{Q}_{k-1}, \tag{3.25b}$$

to obtain the prior pdf at time step k, compactly represented by its mean $\hat{\boldsymbol{x}}_k^-$ and covariance $\boldsymbol{P}_k^-$. This is a direct result of solving the Chapman-

[15] In the case of autocorrelated and therefore non-white process noise, the Markov assumption is violated. However, this violation can be resolved by designing a preceding prewhitening system so that the combination of the original and the prewhitening system is driven by white noise again [18]. Similar techniques exist for autocorrelated measurement noise [18].

[16] It can be shown that the KF is the best *linear* estimator with respect to minimizing the mean square error even in the non-Gaussian case, see [6]. In this more general case, however, the filter does not provide the exact solution to (3.8) anymore.

Kolmogorov equation in (3.8) exactly for the Gaussian densities

$$\begin{aligned} p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}) &= \mathcal{N}(\boldsymbol{x}_k; \boldsymbol{A}_{k-1}\boldsymbol{x}_{k-1} + \boldsymbol{B}_{k-1}\boldsymbol{u}_{k-1}, \boldsymbol{Q}_{k-1}), && (3.26a) \\ p(\boldsymbol{x}_{k-1}|\boldsymbol{z}_{1:k-1}) &= \mathcal{N}(\boldsymbol{x}_{k-1}; \hat{\boldsymbol{x}}_{k-1}, \boldsymbol{P}_{k-1}), && (3.26b) \end{aligned}$$

see e.g. [52], which leads to

$$p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1}) = \mathcal{N}(\boldsymbol{x}_k; \hat{\boldsymbol{x}}_k^-, \boldsymbol{P}_k^-). \tag{3.27}$$

Second, the measurement prediction step

$$\begin{aligned} \hat{\boldsymbol{z}}_k^- &= \boldsymbol{H}_k \hat{\boldsymbol{x}}_k^-, && (3.28a) \\ \boldsymbol{S}_k &= \boldsymbol{H}_k \boldsymbol{P}_k^- \boldsymbol{H}_k^T + \boldsymbol{R}_k && (3.28b) \end{aligned}$$

is carried out with the help of (3.24b) to obtain predicted measurements $\hat{\boldsymbol{z}}_k^-$ with associated innovation covariance matrix $\boldsymbol{S}_k \in \mathbb{R}^{q\times q}$. This is a direct result of writing the normalization factor in (3.8) in terms of

$$p(\boldsymbol{z}_k|\boldsymbol{z}_{1:k-1}) = \int_{\boldsymbol{x}_k} p(\boldsymbol{z}_k|\boldsymbol{x}_k) p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1}) \mathrm{d}\boldsymbol{x}_k \tag{3.29}$$

and solving it for the Gaussian densities

$$\begin{aligned} p(\boldsymbol{z}_k|\boldsymbol{x}_k) &= \mathcal{N}(\boldsymbol{z}_k; \boldsymbol{H}_k\boldsymbol{x}_k, \boldsymbol{R}_k), && (3.30a) \\ p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1}) &= \mathcal{N}(\boldsymbol{x}_k; \hat{\boldsymbol{x}}_k^-, \boldsymbol{P}_k^-), && (3.30b) \end{aligned}$$

which results in

$$p(\boldsymbol{z}_k|\boldsymbol{z}_{1:k-1}) = \mathcal{N}(\boldsymbol{z}_k; \hat{\boldsymbol{z}}_k^-, \boldsymbol{S}_k). \tag{3.31}$$

In the subsequent correction step, predicted, a priori states can be updated to retrieve the a posteriori MMSE state estimate[17] $\hat{\boldsymbol{x}}_k^{\mathrm{M}}$ and associated error covariance matrix $\boldsymbol{P}_k$ according to

$$\begin{aligned} \hat{\boldsymbol{x}}_k^M &= \hat{\boldsymbol{x}}_k^- + \boldsymbol{K}_k \underbrace{(\boldsymbol{z}_k - \hat{\boldsymbol{z}}_k^-)}_{\boldsymbol{v}_k}, && (3.32a) \\ \boldsymbol{P}_k &= (\mathbb{1} - \boldsymbol{K}_k\boldsymbol{H}_k)\boldsymbol{P}_k^- = \boldsymbol{P}_k^- - \boldsymbol{K}_k\boldsymbol{S}_k\boldsymbol{K}_k^T, && (3.32b) \end{aligned}$$

with innovation $\boldsymbol{v}_k \in \mathbb{R}^q$ and Kalman gain $\boldsymbol{K}_k \in \mathbb{R}^{n\times q}$ given by

$$\boldsymbol{K}_k = \boldsymbol{P}_k^- \boldsymbol{H}_k^T \boldsymbol{S}_k^{-1}. \tag{3.33}$$

[17] This is also the MAP estimate due to the Gaussian posterior.

This correction step is obtained by plugging the predicted density (3.27), the likelihood function (3.30a) and the normalization factor (3.31) into (3.8), which results in

$$p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}) = \mathcal{N}(\boldsymbol{x}_k; \hat{\boldsymbol{x}}_k^{\mathrm{M}}, \boldsymbol{P}_k). \tag{3.34}$$

For an actual implementation, the algebraically equivalent, but numerically more stable, Joseph form of the covariance update

$$\boldsymbol{P}_k = (\mathbb{1} - \boldsymbol{K}_k\boldsymbol{H}_k)\boldsymbol{P}_k^-(\mathbb{1} - \boldsymbol{K}_k\boldsymbol{H}_k)^T + \boldsymbol{K}_k\boldsymbol{R}_k\boldsymbol{K}_k^T \tag{3.35}$$

should be used instead of (3.32b) because it preserves the positive-definiteness of the updated covariance matrix [34, 87].

Observing (3.32a) and (3.33), it becomes obvious that the KF balances the influence of the measurement and the prediction according to their uncertainties. If the state prediction is good (“small” $\boldsymbol{P}_k^-$, e.g. due to “small” $\boldsymbol{Q}_{k-1}$) and the measurement uncertainty is large (“large” $\boldsymbol{S}_k$ due to “large” $\boldsymbol{R}_k$), the Kalman gain will be “small”. Therefore, the filter strongly trusts the prediction. The filter bandwidth is lower in this case. Conversely, worse predictions and better measurements result in a larger Kalman gain and the filter follows the measurements more strongly. The filter bandwidth is higher in this case.

From a computational point of view, the KF correction step is the bottleneck for high-dimensional measurement spaces as it requires the inversion of the innovation covariance matrix $\boldsymbol{S}_k$, a $q \times q$-matrix in (3.33) [247]. Therefore, the standard KF formulation is best suitable for rather low-dimensional measurement vectors. Otherwise, the IF presented in the following is advantageous.

3.1.3 Information Filter

The IF is mathematically equivalent to the KF and thus based on the same underlying assumptions. In contrast to the KF, it represents all occurring Gaussian pdfs not in terms of their moments, i.e. mean $\boldsymbol{x}$ and covariance matrix $\boldsymbol{P}$, but by the dual, canonical parametrization in form of an information vector $\boldsymbol{i} \in \mathbb{R}^n$ and an information matrix $\boldsymbol{I} \in \mathbb{R}^{n \times n}$. The information matrix is given by the inverse of the covariance matrix

$$\boldsymbol{I} = \boldsymbol{P}^{-1}, \tag{3.36}$$

while the relation between mean and information vector follows

$$\boldsymbol{i} = \boldsymbol{P}^{-1}\boldsymbol{x} = \boldsymbol{I}\boldsymbol{x}. \tag{3.37}$$

Similar to the KF, the IF can be interpreted in terms of a prediction and correction step. The predicted information vector and the predicted information matrix are given by

$$\hat{\boldsymbol{i}}_k^- = \boldsymbol{I}_k^-(\boldsymbol{A}_{k-1}\boldsymbol{I}_{k-1}^{-1}\hat{\boldsymbol{i}}_{k-1} + \boldsymbol{B}_{k-1}\boldsymbol{u}_{k-1}), \tag{3.38a}$$

$$\boldsymbol{I}_k^- = (\boldsymbol{A}_{k-1}\boldsymbol{I}_{k-1}^{-1}\boldsymbol{A}_{k-1}^T + \boldsymbol{Q}_{k-1})^{-1}, \tag{3.38b}$$

which are corrected by

$$\hat{\boldsymbol{i}}_k = \boldsymbol{H}_k^T\boldsymbol{R}_k^{-1}\boldsymbol{z}_k + \hat{\boldsymbol{i}}_k^-, \tag{3.39a}$$

$$\boldsymbol{I}_k = \boldsymbol{H}_k^T\boldsymbol{R}_k^{-1}\boldsymbol{H}_k + \boldsymbol{I}_k^- \tag{3.39b}$$

upon arrival of a new measurement at time step k [247]. Complete ignorance of the initial state distribution can easily be specified by setting both the information vector and the information matrix to zero, which would require an infinitely large covariance matrix in the standard KF formulation. However, the components of the information vector do not directly permit a physical interpretation, so that both state vector and covariance have to be recovered from (3.36 and 3.37) for final output purpose. This is possible as long as $\boldsymbol{I}_k$ is invertible.

From a computational point of view, KFs and IFs have opposing strengths and weaknesses [247]. On the one hand, the IF prediction step is more complex as it involves the inversion of two $n \times n$-matrices in (3.38b). On the other hand, its correction step is computationally simpler [247].[18] The IF is therefore advantageous if the dimension of the measurement vector is much larger than the dimension of the state vector. If a linear system assumption is not justified, both KF and IF cannot be used and alternatives such as the EKF or UKF are necessary, which are explained in the following.

3.1.4 Extended Kalman Filter

The EKF [6, 18] is a well-known filter for estimating states of nonlinear stochastic systems of the form

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k) + \boldsymbol{w}_k, \tag{3.40a}$$

$$\boldsymbol{z}_k = \boldsymbol{h}(\boldsymbol{x}_k) + \boldsymbol{v}_k, \tag{3.40b}$$

[18] The calculation time for the inversion of the measurement noise covariance matrix $\boldsymbol{R}_k$ is negligible because it is normally a diagonal matrix or at least time-invariant and therefore only needs to be inverted once.

with input vector $\boldsymbol{u}_k \in \mathbb{R}^p$, state vector $\boldsymbol{x}_k \in \mathbb{R}^n$, measurement vector $\boldsymbol{z}_k \in \mathbb{R}^q$, process noise vector $\boldsymbol{w}_k \in \mathbb{R}^n$, measurement noise vector $\boldsymbol{v}_k \in \mathbb{R}^q$, system function $\boldsymbol{f}(\cdot)$, and measurement function $\boldsymbol{h}(\cdot)$. The prerequisite is that the noise sequences are mutually independent, zero-mean, white, and Gaussian with process noise covariance matrix $\boldsymbol{Q}_k \in \mathbb{R}^{n\times n}$ and measurement noise covariance matrix $\boldsymbol{R}_k \in \mathbb{R}^{q\times q}$, and that the initial state density is Gaussian.

Due to the nonlinearities involved in $\boldsymbol{f}(\cdot)$ and $\boldsymbol{h}(\cdot)$, the posterior pdf becomes distorted and is no longer Gaussian. As the EKF inherits the Gaussian posterior representation from the KF, it provides only an approximate solution to the filtering task and becomes less accurate at stronger nonlinearities and larger uncertainties [247]. Its main idea is to approximate the nonlinear functions by their first-order[19] Taylor series expansions. This requires knowledge of the Jacobian matrix of the system function

$$\boldsymbol{A}_{k-1} = \left[\frac{\partial \boldsymbol{f}(\boldsymbol{x},\boldsymbol{u})}{\partial \boldsymbol{x}}\right]_{\boldsymbol{x}=\hat{\boldsymbol{x}}_{k-1},\boldsymbol{u}_{k-1}} \tag{3.41}$$

as well as of the measurement function

$$\boldsymbol{H}_k = \left[\frac{\partial \boldsymbol{h}(\boldsymbol{x})}{\partial \boldsymbol{x}}\right]_{\boldsymbol{x}=\hat{\boldsymbol{x}}_k^-} . \tag{3.42}$$

The Jacobian of the system function is evaluated at the a posteriori state estimate of the last time step, while the Jacobian of the measurement function is evaluated at the predicted, a priori state estimate of the current time step.[20] The state prediction then follows

$$\hat{\boldsymbol{x}}_k^- = \boldsymbol{f}(\hat{\boldsymbol{x}}_{k-1}, \boldsymbol{u}_{k-1}), \tag{3.43a}$$

$$\boldsymbol{P}_k^- = \boldsymbol{A}_{k-1}\boldsymbol{P}_{k-1}\boldsymbol{A}_{k-1}^T + \boldsymbol{Q}_{k-1}, \tag{3.43b}$$

while the measurement prediction is given by

$$\hat{\boldsymbol{z}}_k^- = \boldsymbol{h}(\hat{\boldsymbol{x}}_k^-), \tag{3.44a}$$

$$\boldsymbol{S}_k = \boldsymbol{H}_k\boldsymbol{P}_k^-\boldsymbol{H}_k^T + \boldsymbol{R}_k. \tag{3.44b}$$

[19] Higher-order EKFs may be obtained by retaining more terms of the Taylor series expansions. A second-order EKF, for example, requires knowledge of Jacobian and Hessian matrices.

[20] An EKF variant called iterated EKF linearizes the nonlinear measurement function around the updated (corrected) state instead of the predicted state [18]. This, however, provides greater advantages only if the states that induce the nonlinearity in the measurement equation are directly measurable [146].

The subsequent correction step is identical to the standard KF, see (3.32).

Although the EKF provides a simple and often employed approximation, it suffers from several inherent drawbacks. First, the derivation of Jacobians is error-prone or even impossible if the system or measurement function is not differentiable. Second, the EKF can diverge in the case of linearizations at false operating points, e.g. if the initial conditions are chosen wrongly [17, 113]. The UKF presented in the following does not suffer from the first drawback and often performs better with regard to false initial conditions [53].

3.1.5 Unscented Kalman Filter

The UKF, originally introduced in [118], is based on the intuition that it is easier to approximate a probability distribution than it is to approximate an arbitrary nonlinear function or transformation [118]. It uses the so-called unscented transformation to propagate a set of weighted sigma points through a nonlinear function without any analytical linearization. Hereby, the UKF captures higher-order moments of a distribution than the linearization approach of the EKF. The UKF prediction results in approximations that are accurate to the third-order Taylor series expansion for Gaussian inputs for all nonlinearities and to at least the second-order for non-Gaussian inputs [118]. The standard EKF only has first-order accuracy [252]. Figure 3.3 illustrates the difference between how EKFs and UKFs propagate a pdf in a one-dimensional example.

In the following, the nowadays common version of the UKF that uses the *scaled* unscented transformation [117] is presented, which ensures that the estimated covariance matrix of the transformed pdf remains positive semi-definite. Under consideration of additive noise, the UKF state prediction step is then given by [118][21]

$$\boldsymbol{\mathcal{X}}_{k-1} = \left(\hat{\boldsymbol{x}}_{k-1} \quad \gamma_{\mathrm{U}}\sqrt{\boldsymbol{P}_{k-1}} \oplus \hat{\boldsymbol{x}}_{k-1} \quad -\gamma_{\mathrm{U}}\sqrt{\boldsymbol{P}_{k-1}} \oplus \hat{\boldsymbol{x}}_{k-1}\right), \tag{3.45a}$$

$$\boldsymbol{\mathcal{X}}^*_{i,k} = \boldsymbol{f}(\boldsymbol{\mathcal{X}}_{i,k-1}, \boldsymbol{u}_{k-1}), \qquad \forall i \in \{0, \dots, 2n\}, \tag{3.45b}$$

$$\hat{\boldsymbol{x}}^-_k = \sum_{i=0}^{2n} w_i^{(m)} \boldsymbol{\mathcal{X}}^*_{i,k}, \tag{3.45c}$$

$$\boldsymbol{P}^-_k = \sum_{i=0}^{2n} w_i^{(c)} \left(\boldsymbol{\mathcal{X}}^*_{i,k} - \hat{\boldsymbol{x}}^-_k\right)\left(\boldsymbol{\mathcal{X}}^*_{i,k} - \hat{\boldsymbol{x}}^-_k\right)^T + \boldsymbol{Q}_{k-1}. \tag{3.45d}$$

[21] For shorter notation, the operation of adding a column vector $\boldsymbol{x}$ to each column of a matrix $\boldsymbol{X}$ is written as $\boldsymbol{X} \oplus \boldsymbol{x}$.

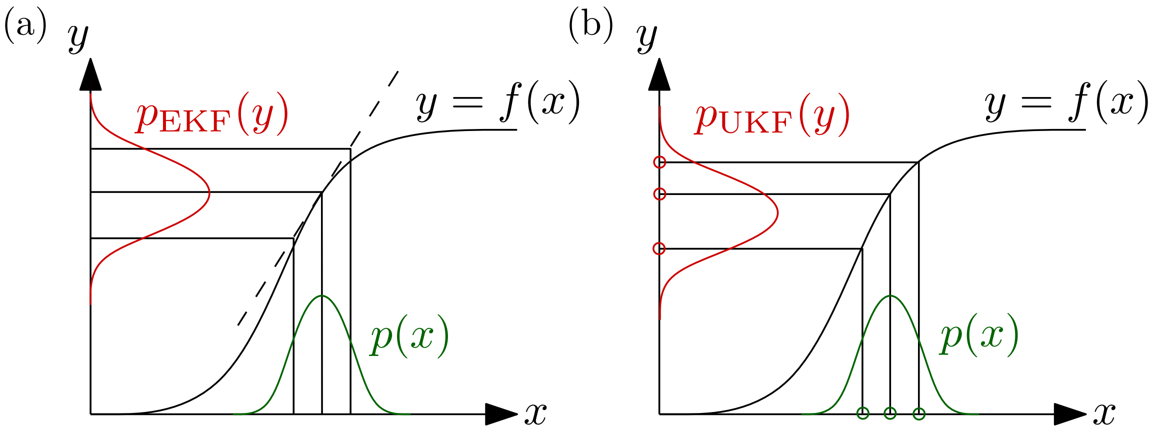


Figure 3.3: Difference between EKF and UKF propagation. The EKF (a) linearizes the nonlinear function $f(x)$ around the mean of the Gaussian distribution and propagates its mean and covariance through the linearized function to obtain the propagated Gaussian pdf $p_{\mathrm{EKF}}(y)$. The UKF (b), in contrast, chooses a set of sigma points (green circles) from the Gaussian distribution, propagates them through the nonlinear function, and retrieves a Gaussian approximation $p_{\mathrm{UKF}}(y)$ from the propagated sigma points (red circles). Note that the true distribution $p(y)$ is not a Gaussian anymore.

First, $2n+1$ sigma points $\boldsymbol{\mathcal{X}}_{i,k-1}$ with $i \in \{0,\ldots,2n\}$, subsumed in the matrix $\boldsymbol{\mathcal{X}}_{k-1} = (\boldsymbol{\mathcal{X}}_{0,k-1} \ldots \boldsymbol{\mathcal{X}}_{2n,k-1}) \in \mathbb{R}^{n\times(2n+1)}$, are chosen based on a square-root decomposition of the a posteriori error covariance of the last time step in (3.45a).[22] Each sigma point has two associated scalar weights $w_i^{(m)}$ and $w_i^{(c)}$ given by

$$w_0^{(m)} = \frac{\lambda_{\mathrm{U}}}{n+\lambda_{\mathrm{U}}}, \quad w_0^{(c)} = \frac{\lambda_{\mathrm{U}}}{n+\lambda_{\mathrm{U}}} + (1-\alpha_{\mathrm{U}}^2+\beta_{\mathrm{U}}), \tag{3.46a}$$

$$w_i^{(m)} = w_i^{(c)} = \frac{1}{2(n+\lambda_{\mathrm{U}})}, \qquad \forall i \in \{1,\ldots,2n\}, \tag{3.46b}$$

$$\lambda_{\mathrm{U}} = \alpha_{\mathrm{U}}^2(n+\kappa_{\mathrm{U}}) - n, \tag{3.46c}$$

$$\gamma_{\mathrm{U}} = \sqrt{n+\lambda_{\mathrm{U}}}, \tag{3.46d}$$

with scaling parameters $\alpha_{\mathrm{U}}, \beta_{\mathrm{U}}, \kappa_{\mathrm{U}}, \lambda_{\mathrm{U}}, \gamma_{\mathrm{U}}$ [247]. The parameter α_{U} determines the spread of the sigma points and should ideally be a small number in the interval $0 \leq \alpha_{\mathrm{U}} \leq 1$ to avoid sampling significant non-local effects at

[22] Here, it is assumed that the matrix square root $\sqrt{\boldsymbol{P}}$ of $\boldsymbol{P}$ is of the form $\boldsymbol{P} = \sqrt{\boldsymbol{P}}\sqrt{\boldsymbol{P}}^T$ so that the columns of $\sqrt{\boldsymbol{P}}$ are used to calculate the sigma points. If, otherwise, $\boldsymbol{P} = \sqrt{\boldsymbol{P}}^T\sqrt{\boldsymbol{P}}$ holds, the rows of $\sqrt{\boldsymbol{P}}$ have to be used. The matrix square root is calculated by a numerically stable Cholesky decomposition.

stronger nonlinearities. The parameter κ_U has to be chosen non-negative to guarantee the positive semi-definiteness of the covariance matrix and is normally set to zero for state estimation, while the non-negative scaling parameter β_U is used to incorporate prior knowledge of the state distribution with $\beta_\mathrm{U} = 2$ being optimal for Gaussian distributions [252]. These sigma points are propagated through the system function $\boldsymbol{f}(\cdot)$ without approximation in (3.45b), and the mean and variance are retrieved with the help of the calculated weights in (3.45c) and (3.45d).

For the subsequent measurement prediction step given by [247]

$$\boldsymbol{\mathcal{X}}_k = \left(\hat{\boldsymbol{x}}_k^- \quad \gamma_\mathrm{U}\sqrt{\boldsymbol{P}_k^-} \oplus \hat{\boldsymbol{x}}_k^- \quad -\gamma_\mathrm{U}\sqrt{\boldsymbol{P}_k^-} \oplus \hat{\boldsymbol{x}}_k^-\right), \tag{3.47a}$$

$$\boldsymbol{\mathcal{Z}}_{i,k} = \boldsymbol{h}(\boldsymbol{\mathcal{X}}_{i,k}), \qquad \forall i \in \{0, \dots, 2n\}, \tag{3.47b}$$

$$\hat{\boldsymbol{z}}_k^- = \sum_{i=0}^{2n} w_i^{(m)} \boldsymbol{\mathcal{Z}}_{i,k}, \tag{3.47c}$$

$$\boldsymbol{S}_k = \sum_{i=0}^{2n} w_i^{(c)} \left(\boldsymbol{\mathcal{Z}}_{i,k} - \hat{\boldsymbol{z}}_k^-\right)\left(\boldsymbol{\mathcal{Z}}_{i,k} - \hat{\boldsymbol{z}}_k^-\right)^T + \boldsymbol{R}_k, \tag{3.47d}$$

new sigma points $\boldsymbol{\mathcal{X}}_{i,k}$ are chosen in (3.47a), propagated through the measurement function in $\boldsymbol{h}(\cdot)$ in (3.47b), and the predicted, a priori measurement vector $\hat{\boldsymbol{z}}_k^-$ with associated innovation covariance matrix $\boldsymbol{S}_k$ can finally be recovered from the propagated sigma points $\boldsymbol{\mathcal{Z}}_{i,k}$ via (3.47c) and (3.47d).

For the correction step, the Kalman gain is calculated as

$$\boldsymbol{K}_k = \boldsymbol{C}_k \boldsymbol{S}_k^{-1}, \tag{3.48}$$

with the help of the cross covariance matrix $\boldsymbol{C}_k$ between predicted measurements and states given by

$$\boldsymbol{C}_k = \sum_{i=0}^{2n} w_i^{(c)} \left(\boldsymbol{\mathcal{X}}_{i,k} - \hat{\boldsymbol{x}}_k^-\right)\left(\boldsymbol{\mathcal{Z}}_{i,k} - \hat{\boldsymbol{z}}_k^-\right)^T. \tag{3.49}$$

The rest is identical to the KF correction step, see (3.32).

Finally, it should be noted that there also exist other kinds of so-called sigma point KFs, e.g. the Central Difference KF presented in [114]. These, however, only differ in the way the sigma points are calculated [253]. From a computational point of view, the UKF is of the same order of magnitude as the EKF [119] while providing more accurate state estimates [232]. For linear Gaussian systems, the results of the UKF are identical to those obtained by the KF [247].

3.1.6 Multiple Model Optimal Bayesian Filter

So far, we considered a real system to behave according to exactly one specified process model or system function. However, in many applications, a system changes its dynamic behavior over time and it is only possible to specify a set of possibly structurally different models, each of which describe one of these behaviors. If it is unknown which of these process models is the correct one at a specific time, then structural uncertainty arises. Examples are target tracking applications with so-called maneuvering targets, which change their motion behavior over time, e.g. from moving straight with nearly constant velocity up to performing a coordinated turn. If systems like these are filtered with a single model, two problems occur. First, the hypothesized model does not match the true maneuvering system to be filtered and tracked, which results in suboptimal state estimates. Second, and more severe, the data association becomes error-prone. This is because the predicted measurement lies further away from the true measurement, which is especially problematic in clutter situations with false alarms – a problem that will be treated in more detail in Section 3.2. Therefore, the true so-called motion mode or behavior pattern of an object has to be estimated together with its states simultaneously.[23]

Various approaches to cope with this problem can be found in the literature, which are summarized under the term multiple model methods, see for example [18,53,153]. In the Bayesian framework, this general inference task is equal to estimate the joint pdf

$$p(\boldsymbol{x}_k, M_k|\boldsymbol{z}_{1:k}) \tag{3.50}$$

given all measurements $\boldsymbol{z}_{1:k}$, with continuous random variable $\boldsymbol{x}_k$, representing the object's states, and discrete random variable M_k for the motion mode.[24]

[23] Approaches that partially try to cope with these problems by sticking to single model filtering can be found in [151]. They include, amongst others, process noise adaption and input estimation. Process noise adaption approaches raise the process noise as soon as a maneuver is detected to make the filter follow the measurements more closely and to prevent track losses. However, the gain is limited because a maneuver cannot be adequately modeled by additional white noise in general [53]. Input estimation approaches, in contrast, try to estimate the unknown input vector online. Disadvantageously, errors in this estimation have a direct, negative effect on the state estimate. As these methods are inferior to true multiple model methods, for a comparison see [18], they are not considered further.

[24] Throughout this work, continuous random variables are written in lower case and discrete random variables in capital letters.

The next section provides the extension of the optimal Bayesian filter to this hybrid estimation problem as well as a well-known approximation in form of the IMM Filter used in this work.

Extension of the Optimal Bayesian Filter

From a modeling point of view, systems with structural uncertainty can be described by jump Markov nonlinear systems, whose states and mode probabilities are to be estimated recursively. In this context, the term mode probability refers to the probability of the event that a hypothesized model matches the true behavior pattern or mode of a system [153]. We consider the j-th model of a fixed model set[25] $\mathcal{M} = \left\{M_j\right\}_{j=1}^{r}$ to be given by the nonlinear stochastic state space model

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}_j(\boldsymbol{x}_k, \boldsymbol{u}_k) + \boldsymbol{w}_{j,k}, \tag{3.51a}$$

$$\boldsymbol{z}_k = \boldsymbol{h}_j(\boldsymbol{x}_k) + \boldsymbol{v}_{j,k}, \tag{3.51b}$$

with input vector $\boldsymbol{u}_k \in \mathbb{R}^p$, state vector $\boldsymbol{x}_k \in \mathbb{R}^n$, measurement vector $\boldsymbol{z}_k \in \mathbb{R}^q$, process noise vector $\boldsymbol{w}_{j,k} \in \mathbb{R}^n$, measurement noise vector $\boldsymbol{v}_{j,k} \in \mathbb{R}^q$, system function $\boldsymbol{f}_j(\cdot)$, and measurement function $\boldsymbol{h}_j(\cdot)$ at each time step k. The noise sequences are mutually independent, zero-mean, and white with process noise covariance matrix $\boldsymbol{Q}_{j,k} \in \mathbb{R}^{n\times n}$ and measurement noise covariance matrix $\boldsymbol{R}_{j,k} \in \mathbb{R}^{q\times q}$.

The evolution of the hybrid system among the r models is supposed to obey a first-order Markov chain with time-invariant Markovian model transition probability matrix

$$\boldsymbol{\Pi} = \begin{pmatrix} p_{11} & \cdots & p_{r1} \\ \vdots & \ddots & \vdots \\ p_{1r} & \cdots & p_{rr} \end{pmatrix} \in \mathbb{R}^{r\times r}, \tag{3.52}$$

in which p_{ij} denotes the probability that a mode transition occurs from model i to model j [17], whereas the diagonal elements p_{ii} are the probabilities of a transition from model i to itself. The transition probabilities p_{ij} for $i \neq j$ are designed according to $\sum_{j\neq i} p_{ij} = 1 - p_{ii}$ to model the possible mode jumps [17]. The DBN of this jump Markov system[26] is shown in Figure 3.4.

[25] If even the number of models is allowed to vary over time, it is termed a *Variable Structure Multiple Model* (VSMM) [150, 153].

[26] We consider a so-called left-continuous mode jump process. The model M_k is active during the sampling time interval $]k-1; k]$.

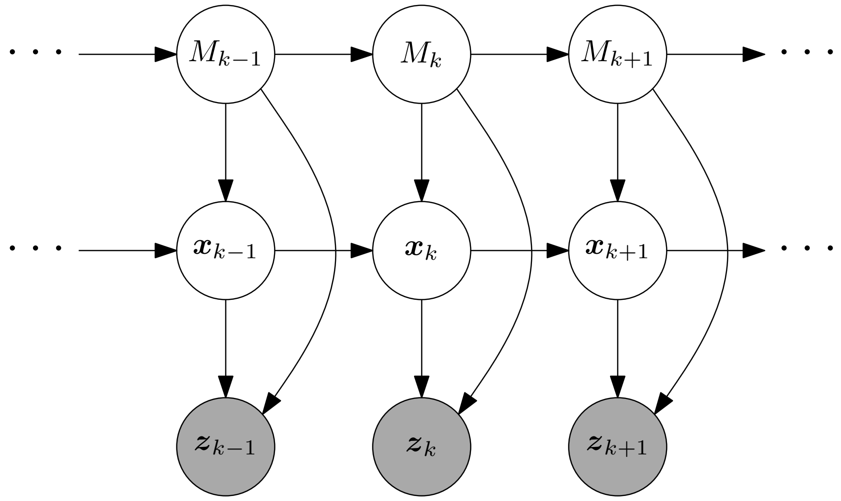


Figure 3.4: Jump Markov process with observable nodes shaded in gray. In filtering tasks, measurements are only available up to time step k.

By decomposing (3.50) into

$$p(\boldsymbol{x}_k, M_k|\boldsymbol{z}_{1:k}) = p(\boldsymbol{x}_k|M_k, \boldsymbol{z}_{1:k})p(M_k|\boldsymbol{z}_{1:k}), \tag{3.53}$$

the filtering task can be interpreted as a two stage estimation process. The so-called mode learning stage $p(M_k|\boldsymbol{z}_{1:k})$ and the state inference stage $p(\boldsymbol{x}_k|M_k, \boldsymbol{z}_{1:k})$ [262]. Although the optimal Bayesian filter presented in Section 3.1.1 is still valid by considering a combined, hybrid state space consisting of M_k and $\boldsymbol{x}_k$, the relations become more evident by explicitly distinguishing between these two random variables. With the abbreviation $p(M_{j,k}) = p(M_k = j)$ for the probability that the discrete random variable M_k is in mode j, the conditional state density $p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k})$ can be found by marginalizing out M_k, which results in

$$\begin{aligned}
p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}) &= \sum_{j=1}^{r} p(\boldsymbol{x}_k, M_{j,k}|\boldsymbol{z}_{1:k}) = \sum_{j=1}^{r} p(\boldsymbol{x}_k|M_{j,k}, \boldsymbol{z}_{1:k}) \underbrace{p(M_{j,k}|\boldsymbol{z}_{1:k})}_{\mu_{j,k}} \\
&= \sum_{j=1}^{r} \mu_{j,k} \frac{p(\boldsymbol{z}_k|\boldsymbol{x}_k, M_{j,k}, \boldsymbol{z}_{1:k-1})}{p(\boldsymbol{z}_k|M_{j,k}, \boldsymbol{z}_{1:k-1})} p(\boldsymbol{x}_k|M_{j,k}, \boldsymbol{z}_{1:k-1}) \\
&= \sum_{j=1}^{r} \mu_{j,k} \frac{p(\boldsymbol{z}_k|\boldsymbol{x}_k, M_{j,k}, \boldsymbol{z}_{1:k-1})}{p(\boldsymbol{z}_k|M_{j,k}, \boldsymbol{z}_{1:k-1})} \\
&\quad \int_{\boldsymbol{x}_{k-1}} p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, M_{j,k}, \boldsymbol{z}_{1:k-1}) p(\boldsymbol{x}_{k-1}|M_{j,k}, \boldsymbol{z}_{1:k-1}) \mathrm{d}\boldsymbol{x}_{k-1}
\end{aligned} \tag{3.54}$$

after an application of Bayes theorem in the second line and finally writing $p(\boldsymbol{x}_k|M_{j,k}, \boldsymbol{z}_{1:k-1})$ according to the Chapman-Kolmogorov equation, see (3.8), p. 21, in analogy to the single model case.

With the imposed conditional independence assumptions visualized in the DBN, (3.54) reduces to

$$\begin{aligned} p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}) = \sum_{j=1}^{r} \mu_{j,k} \frac{\overbrace{p(\boldsymbol{z}_k|\boldsymbol{x}_k, M_{j,k})}^{\text{Measurement Likelihood}}}{\underbrace{p(\boldsymbol{z}_k|M_{j,k}, \boldsymbol{z}_{1:k-1})}_{\text{Normalization}}} \\ \underbrace{\overbrace{\int_{\boldsymbol{x}_{k-1}} p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, M_{j,k}) p(\boldsymbol{x}_{k-1}|M_{j,k}, \boldsymbol{z}_{1:k-1}) \mathrm{d}\boldsymbol{x}_{k-1}}^{p(\boldsymbol{x}_k|M_{j,k},\boldsymbol{z}_{1:k-1})}}_{\text{Prior (Chapman-Kolmogorov)}}, \end{aligned} \tag{3.55}$$

in which $\mu_{j,k} = p(M_{j,k}|\boldsymbol{z}_{1:k})$ is the mode probability of model j. The transition density as well as the measurement likelihood are induced by the j-th state space model according to

$$p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}, M_{j,k}) = p_{\boldsymbol{w}_{j,k-1}}\big(\boldsymbol{x}_k - \boldsymbol{f}_j(\boldsymbol{x}_{k-1}, \boldsymbol{u}_{k-1})\big), \tag{3.56a}$$

$$p(\boldsymbol{z}_k|\boldsymbol{x}_k, M_{j,k}) = p_{\boldsymbol{v}_{j,k}}\big(\boldsymbol{z}_k - \boldsymbol{h}_j(\boldsymbol{x}_k)\big), \tag{3.56b}$$

similar to the single model case. The second pdf of the integrand in (3.55) is now expanded and factorized according to

$$\begin{aligned} p(\boldsymbol{x}_{k-1}|M_{j,k}, \boldsymbol{z}_{1:k-1}) &= \sum_{i=1}^{r} p(\boldsymbol{x}_{k-1}, M_{i,k-1}|M_{j,k}, \boldsymbol{z}_{1:k-1}) \\ &= \sum_{i=1}^{r} p(\boldsymbol{x}_{k-1}|M_{i,k-1}, M_{j,k}, \boldsymbol{z}_{1:k-1}) \underbrace{p(M_{i,k-1}|M_{j,k}, \boldsymbol{z}_{1:k-1})}_{\mu_{(i|j),k-1}} \\ &= \sum_{i=1}^{r} \mu_{(i|j),k-1} p(\boldsymbol{x}_{k-1}|M_{i,k-1}, \boldsymbol{z}_{1:k-1}), \end{aligned} \tag{3.57}$$

in which the last step follows from the fact that states $\boldsymbol{x}_{k-1}$ at time step $k-1$ do not depend on models of time step k. The so-called conditional mode probabilities or mixing probabilities $\mu_{(i|j),k-1}$ in (3.57) are themselves calculated via Bayes theorem as

$$\begin{aligned}\mu_{(i|j),k-1} &= \frac{p(M_{j,k}|M_{i,k-1},\boldsymbol{z}_{1:k-1})p(M_{i,k-1}|\boldsymbol{z}_{1:k-1})}{p(M_{j,k}|\boldsymbol{z}_{1:k-1})} \\ &= \frac{p_{ij}\mu_{i,k-1}}{\sum_{i=1}^{r} p_{ij}\mu_{i,k-1}} = \frac{p_{ij}\mu_{i,k-1}}{\mu_{j,k}^{-}},\end{aligned} \tag{3.58}$$

in which the abbreviation $\mu_{j,k}^{-} = p(M_{j,k}|\boldsymbol{z}_{1:k-1})$ stands for the predicted mode probability of mode j. This predicted mode probability is calculated from all a posteriori mode probabilities of the last time step and elements from $\boldsymbol{\Pi}$.

The normalization factor in (3.55) can then be written as

$$\begin{aligned}p(\boldsymbol{z}_k|M_{j,k},\boldsymbol{z}_{1:k-1}) &= \int_{\boldsymbol{x}_k} p(\boldsymbol{z}_k,\boldsymbol{x}_k|M_{j,k},\boldsymbol{z}_{1:k-1})\mathrm{d}\boldsymbol{x}_k \\ &= \int_{\boldsymbol{x}_k} p(\boldsymbol{z}_k|\boldsymbol{x}_k,M_{j,k})p(\boldsymbol{x}_k|M_{j,k},\boldsymbol{z}_{1:k-1})\mathrm{d}\boldsymbol{x}_k\end{aligned} \tag{3.59}$$

by applying the conditional independence assumption of a measurement $\boldsymbol{z}_k$ from the older measurement sequence $\boldsymbol{z}_{1:k-1}$ given a state $\boldsymbol{x}_k$ and model $M_{j,k}$ as visible in the DBN.

It remains to find a recursion for the mode probabilities. By applying Bayes theorem to $\mu_{j,k}$ in (3.55), we reach

$$\mu_{j,k} = p(M_{j,k}|\boldsymbol{z}_{1:k}) = \frac{\overbrace{p(\boldsymbol{z}_k|M_{j,k},\boldsymbol{z}_{1:k-1})}^{\text{Model Likelihood } \lambda_{j,k}}\overbrace{p(M_{j,k}|\boldsymbol{z}_{1:k-1})}^{\mu_{j,k}^{-}}}{p(\boldsymbol{z}_k|\boldsymbol{z}_{1:k-1})}. \tag{3.60}$$

The model likelihood $\lambda_{j,k} = p(\boldsymbol{z}_k|M_{j,k},\boldsymbol{z}_{1:k-1})$ is equal to the normalizer (3.59) of the conditional state density, whereas the normalizer $p(\boldsymbol{z}_k|\boldsymbol{z}_{1:k-1})$ in (3.60) can be written in terms of the numerator as

$$p(\boldsymbol{z}_k|\boldsymbol{z}_{1:k-1}) = \sum_{j=1}^{r} p(\boldsymbol{z}_k|M_{j,k},\boldsymbol{z}_{1:k-1})p(M_{j,k}|\boldsymbol{z}_{1:k-1}) = \sum_{j=1}^{r} \lambda_{j,k}\mu_{j,k}^{-}. \tag{3.61}$$

Now that all terms in (3.55) are specified, the MMSE point estimate $\boldsymbol{x}_k^{\mathrm{M}}$ and the associated error covariance matrix $\boldsymbol{P}_k$ can be calculated from the first line of (3.54) according to

$$\hat{\boldsymbol{x}}_k^{\mathrm{M}} = \int_{\boldsymbol{x}_k} \boldsymbol{x}_k p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k})\mathrm{d}\boldsymbol{x}_k = \int_{\boldsymbol{x}_k} \boldsymbol{x}_k \sum_{j=1}^{r} p(\boldsymbol{x}_k|M_{j,k},\boldsymbol{z}_{1:k})\mu_{j,k}\mathrm{d}\boldsymbol{x}_k, \tag{3.62a}$$

$$\boldsymbol{P}_k = \int_{\boldsymbol{x}_k} \Big((\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k^{\mathrm{M}})(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k^{\mathrm{M}})^T\Big) \sum_{j=1}^{r} p(\boldsymbol{x}_k|M_{j,k},\boldsymbol{z}_{1:k})\mu_{j,k}\mathrm{d}\boldsymbol{x}_k, \tag{3.62b}$$

or in the more convenient form

$$\hat{\boldsymbol{x}}_k^{\mathrm{M}} = \sum_{j=1}^{r} \Big(\mu_{j,k} \int_{\boldsymbol{x}_k} \boldsymbol{x}_k p(\boldsymbol{x}_k|M_{j,k},\boldsymbol{z}_{1:k})\mathrm{d}\boldsymbol{x}_k\Big), \tag{3.63a}$$

$$\boldsymbol{P}_k = \sum_{j=1}^{r} \Big(\mu_{j,k} \int_{\boldsymbol{x}_k} \Big((\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k^{\mathrm{M}})(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k^{\mathrm{M}})^T\Big) p(\boldsymbol{x}_k|M_{j,k},\boldsymbol{z}_{1:k})\mathrm{d}\boldsymbol{x}_k\Big) \tag{3.63b}$$

by rearranging summation and integrals [53]. This *optimal* Bayesian filtering solution is, however, intractable even under LGS assumptions. This is because even under this assumption, an exponentially increasing number of (Gaussian) mixture components are induced by (3.57) over time, requiring r^k filters, one matched to each possible mode sequence [18]. Therefore, approximate schemes have to be employed. The most prominent approximation is the IMM Filter presented in the following, which yields an excellent compromise between computational complexity and achievable filtering results [167].

3.1.7 Interacting Multiple Model Filter

The IMM Filter, originally introduced in [36], provides an approximate solution to the sub-class of jump Markov linear systems, also called switching Kalman Filter models, in which the j-th model of a model set $\mathcal{M} = \{M_j\}_{j=1}^{r}$ is given by

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_{j,k}\boldsymbol{x}_k + \boldsymbol{B}_{j,k}\boldsymbol{u}_k + \boldsymbol{w}_{j,k}, \tag{3.64a}$$

$$\boldsymbol{z}_k = \boldsymbol{H}_{j,k}\boldsymbol{x}_k + \boldsymbol{v}_{j,k}, \tag{3.64b}$$

with input vector $\boldsymbol{u}_k \in \mathbb{R}^p$, state vector $\boldsymbol{x}_k \in \mathbb{R}^n$, measurement vector $\boldsymbol{z}_k \in \mathbb{R}^q$, process noise vector $\boldsymbol{w}_{j,k} \in \mathbb{R}^n$, measurement noise vector

$\boldsymbol{v}_{j,k} \in \mathbb{R}^q$, system matrix $\boldsymbol{A}_{j,k} \in \mathbb{R}^{n\times n}$, input matrix $\boldsymbol{B}_{j,k} \in \mathbb{R}^{n\times p}$, and measurement matrix $\boldsymbol{H}_{j,k} \in \mathbb{R}^{q\times n}$ under the prerequisite that the noise sequences are mutually independent, zero-mean, white, and Gaussian with process noise covariance matrix $\boldsymbol{Q}_{j,k} \in \mathbb{R}^{n\times n}$ and measurement noise covariance matrix $\boldsymbol{R}_{j,k} \in \mathbb{R}^{q\times q}$. The initial state density for each model is considered Gaussian and model transitions are equivalent to the optimal Bayesian filter with Markovian model transition probability matrix $\boldsymbol{\Pi} \in \mathbb{R}^{r\times r}$. A visual overview of the involved filtering steps is given in Fig. 3.5, which shows how the r individual filters, each matched to one model from $\mathcal{M}$, interact with each other.

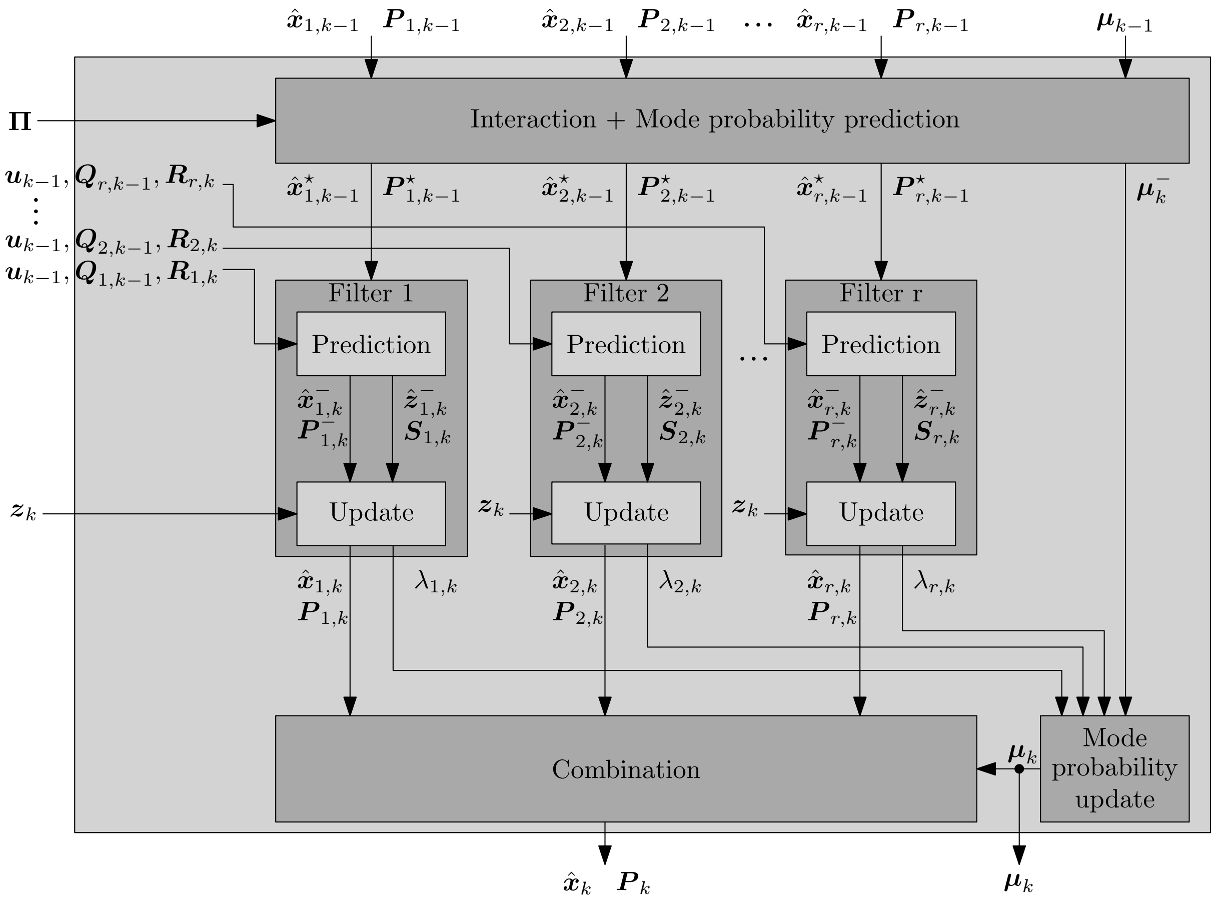

$\hat{\boldsymbol{x}}_{1,k-1}$ $\boldsymbol{P}_{1,k-1}$
$\hat{\boldsymbol{x}}_{2,k-1}$ $\boldsymbol{P}_{2,k-1}$
...
$\hat{\boldsymbol{x}}_{r,k-1}$ $\boldsymbol{P}_{r,k-1}$
$\boldsymbol{\mu}_{k-1}$
$\boldsymbol{\Pi}$
Interaction + Mode probability prediction
$\boldsymbol{u}_{k-1}, \boldsymbol{Q}_{r,k-1}, \boldsymbol{R}_{r,k}$
$\boldsymbol{u}_{k-1}, \boldsymbol{Q}_{2,k-1}, \boldsymbol{R}_{2,k}$
$\boldsymbol{u}_{k-1}, \boldsymbol{Q}_{1,k-1}, \boldsymbol{R}_{1,k}$
$\hat{\boldsymbol{x}}^\star_{1,k-1}$ $\boldsymbol{P}^\star_{1,k-1}$
$\hat{\boldsymbol{x}}^\star_{2,k-1}$ $\boldsymbol{P}^\star_{2,k-1}$
$\hat{\boldsymbol{x}}^\star_{r,k-1}$ $\boldsymbol{P}^\star_{r,k-1}$
$\boldsymbol{\mu}^-_k$
Filter 1
Filter 2
Filter r
Prediction
Prediction
Prediction
...
$\hat{\boldsymbol{x}}^-_{1,k}$ $\boldsymbol{P}^-_{1,k}$ $\hat{\boldsymbol{z}}^-_{1,k}$ $\boldsymbol{S}_{1,k}$
$\hat{\boldsymbol{x}}^-_{2,k}$ $\boldsymbol{P}^-_{2,k}$ $\hat{\boldsymbol{z}}^-_{2,k}$ $\boldsymbol{S}_{2,k}$
$\hat{\boldsymbol{x}}^-_{r,k}$ $\boldsymbol{P}^-_{r,k}$ $\hat{\boldsymbol{z}}^-_{r,k}$ $\boldsymbol{S}_{r,k}$
$\boldsymbol{z}_k$
Update
$\boldsymbol{z}_k$
Update
$\boldsymbol{z}_k$
Update
$\hat{\boldsymbol{x}}_{1,k}$ $\boldsymbol{P}_{1,k}$ $\lambda_{1,k}$
$\hat{\boldsymbol{x}}_{2,k}$ $\boldsymbol{P}_{2,k}$ $\lambda_{2,k}$
$\hat{\boldsymbol{x}}_{r,k}$ $\boldsymbol{P}_{r,k}$ $\lambda_{r,k}$
Combination
$\boldsymbol{\mu}_k$
Mode probability update
$\hat{\boldsymbol{x}}_k$ $\boldsymbol{P}_k$
$\boldsymbol{\mu}_k$

Figure 3.5: Overview of one cycle of the IMM Filter at time step k.

Interaction and Mode Probability Prediction

The main idea of the IMM Filter is to approximate both $p(\boldsymbol{x}_{k-1}|M_{i,k-1}, \boldsymbol{z}_{1:k-1})$ in (3.57) by a Gaussian and the resulting Gaussian mixture $p(\boldsymbol{x}_{k-1}|M_{j,k}, \boldsymbol{z}_{1:k-1})$ by a single Gaussian distribution via moment matching at each time step[27], therefore

$$\underbrace{p(\boldsymbol{x}_{k-1}|M_{j,k},\boldsymbol{z}_{1:k-1})}_{\approx\mathcal{N}(\boldsymbol{x}_{k-1};\hat{\boldsymbol{x}}^\star_{j,k-1},\boldsymbol{P}^\star_{j,k-1})} = \sum_{i=1}^{r} \mu_{(i|j),k-1}\underbrace{p(\boldsymbol{x}_{k-1}|M_{i,k-1},\boldsymbol{z}_{1:k-1})}_{\approx\mathcal{N}(\boldsymbol{x}_{k-1};\hat{\boldsymbol{x}}_{i,k-1},\boldsymbol{P}_{i,k-1})}, \tag{3.65}$$

with

$$\hat{\boldsymbol{x}}^\star_{j,k-1} = \sum_{i=1}^{r} \mu_{(i|j),k-1}\hat{\boldsymbol{x}}_{i,k-1}, \tag{3.66a}$$

$$\boldsymbol{P}^\star_{j,k-1} = \sum_{i=1}^{r} \mu_{(i|j),k-1}\left(\boldsymbol{P}_{i,k-1} + (\hat{\boldsymbol{x}}_{i,k-1} - \hat{\boldsymbol{x}}^\star_{j,k-1})(\hat{\boldsymbol{x}}_{i,k-1} - \hat{\boldsymbol{x}}^\star_{j,k-1})^T\right). \tag{3.66b}$$

In this so-called interaction step, the individual state and covariance estimates of the underlying filters of the last time step, $\hat{\boldsymbol{x}}_{j,k-1}$ and $\boldsymbol{P}_{j,k-1}$, are probabilistically mixed to form a single initial state $\hat{\boldsymbol{x}}^\star_{j,k-1}$ and covariance $\boldsymbol{P}^\star_{j,k-1}$ for each filter j. The necessary mixing probabilities $\mu_{(i|j),k-1}$ are calculated identically to the optimal filter case (3.58) according to

$$\mu_{(i|j),k-1} = \frac{p_{ij}\mu_{i,k-1}}{\mu^-_{j,k}}, \tag{3.67}$$

with the help of the predicted, a priori mode probabilities $\boldsymbol{\mu}^-_k = (\mu^-_{1,k}\dots\mu^-_{r,k})^T$ of the current time step. These are themselves given by the prediction of the mode probabilities of the last time step with elements from $\boldsymbol{\Pi}$ as

$$\mu^-_{j,k} = \sum_{i=1}^{r} p_{ij}\mu_{i,k-1}, \tag{3.68}$$

see (3.58). Consequently, the initial state vector of each filter results from the state estimation of all filters of the last time step. The higher the

[27] The GPB1 Filter, in contrast, approximates the posterior Gaussian mixture state pdf $p(\boldsymbol{x}_{k-1}|\boldsymbol{z}_{1:k-1})$ by a Gaussian, which is used as the common initial pdf for all underlying filters for the current time step. This approach likewise results in r distinct, underlying filters such as the IMM approximation but is less effective due to the premature merging [53]. The computationally more demanding, but in comparison with the IMM Filter only slightly more effective GPB2 Filter approximates only $p(\boldsymbol{x}_{k-1}|M_{i,k-1},\boldsymbol{z}_{1:k-1})$ by a Gaussian, which results in r^2 underlying filters. This can be made obvious by substituting this approximation made in (3.65) into (3.55), p. 39, which yields a Gaussian mixture with r^2 components for $p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k})$.

mode probability of mode i at time step $k-1$ and the higher its transition probability to mode j, the stronger the influence of the previous state estimation of filter i on the initial state vector of filter j.

Model-Specific Filtering

Subsequently, these initial states $\hat{\boldsymbol{x}}^{\star}_{j,k-1}$ and covariances $\boldsymbol{P}^{\star}_{j,k-1}$ are predicted to the current time step k for each model individually to obtain predicted states $\hat{\boldsymbol{x}}^{-}_{j,k}$ and covariances $\boldsymbol{P}^{-}_{j,k}$ as well as predicted measurements $\hat{\boldsymbol{z}}^{-}_{j,k}$ and innovation covariances $\boldsymbol{S}_{j,k}$. These are then corrected with the current measurement $\boldsymbol{z}_k$ at time step k. As all densities involved in (3.55) are Gaussian, this model-specific filtering can be realized by individual KF prediction and correction steps according to Section 3.1.2, which yield a posteriori state and covariance estimates $\hat{\boldsymbol{x}}_{j,k}$ and $\boldsymbol{P}_{j,k}$ for each model.

Mode Probability Update

With the Gaussian model likelihoods given by[28]

$$\begin{aligned}\lambda_{j,k} &= p(\boldsymbol{z}_k|M_{j,k}, \boldsymbol{z}_{1:k-1}) = \mathcal{N}(\boldsymbol{z}_k; \hat{\boldsymbol{z}}^{-}_{j,k}, \boldsymbol{S}_{j,k}) \\ &= \frac{1}{\sqrt{|2\pi \boldsymbol{S}_{j,k}|}} e^{-\frac{1}{2}(\boldsymbol{z}_k - \hat{\boldsymbol{z}}^{-}_{j,k})^T \boldsymbol{S}^{-1}_{j,k}(\boldsymbol{z}_k - \hat{\boldsymbol{z}}^{-}_{j,k})},\end{aligned} \tag{3.69}$$

the mode probabilities can now be updated based on how well the measurements fit to the model just as in the optimal filter (3.60), therefore

$$\mu_{j,k} = \frac{\lambda_{j,k}\mu^{-}_{j,k}}{\sum_{i=1}^{r} \lambda_{i,k}\mu^{-}_{i,k}}. \tag{3.70}$$

Thus, the smaller the (squared) Mahalanobis distance [164] between the measurement prediction of model j given by $\hat{\boldsymbol{z}}^{-}_{j,k}$ and an actual measurement $\boldsymbol{z}_k$ in the exponent of (3.69), the higher the likelihood of model j, which itself results in a higher mode probability for this model according to (3.70).

[28] This result can be seen by observing the analogy of (3.59), p. 40, with (3.29), p. 29, which resulted in (3.31), p. 29.

Combination

For final output purpose, the individual updated filter states are combined to a final state and covariance estimate [17] via

$$\hat{\boldsymbol{x}}_k = \sum_{j=1}^{r} \mu_{j,k} \hat{\boldsymbol{x}}_{j,k}, \tag{3.71a}$$

$$\boldsymbol{P}_k = \sum_{j=1}^{r} \mu_{j,k} \left(\boldsymbol{P}_{j,k} + (\hat{\boldsymbol{x}}_{j,k} - \hat{\boldsymbol{x}}_k)(\hat{\boldsymbol{x}}_{j,k} - \hat{\boldsymbol{x}}_k)^T \right). \tag{3.71b}$$

Note that these combinations are not used in the subsequent filter recursion. The IMM Filter provides the basis for tracking and motion behavior classification of dynamic grid cell hypothesis within the proposed dynamic mapping approach described in Chapter 4.

3.1.8 Binary Bayes Filter

The last section focused on the approximate filtering solution of jump Markov systems with continuous state spaces. The BBF, in contrast, is a very special case of the optimal Bayesian filter, in which the state X to be estimated is *static*, thus not changing over time, and the state space is *discrete* and *binary*, consequently comprises only the two values true and false. The corresponding probabilities are shortly written as $p(X) = p(X = \text{true})$ and $p(\neg X) = p(X = \text{false})$ in the following. Fig. 3.6 shows a DBN visualization of the filtering task that can be solved by a BBF [247].

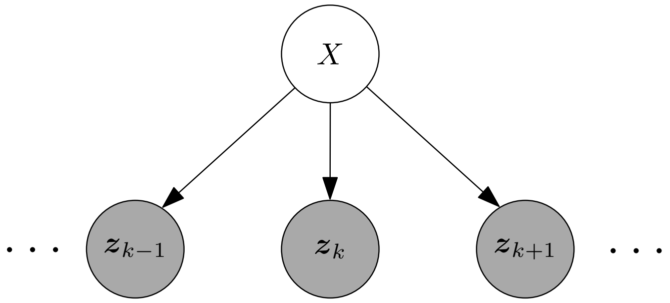


Figure 3.6: BBF model with observable nodes shaded in gray. A static, hidden state is estimated from a measurement sequence. In filtering tasks, measurements are only available up to time step k.

The BBF plays a major role for occupancy grid mapping algorithms in which each cell can only be either free or occupied, and is particularly suitable for situations, in which a so-called *inverse* measurement or *inverse* sensor model of the form $p(X|\boldsymbol{z}_k)$ can be specified instead of the familiar *forward* sensor models $p(\boldsymbol{z}_k|X)$ considered so far. The inverse sensor model specifies a distribution over the binary state variable X as a function of the measurement $\boldsymbol{z}_k$ and thus reasons from effects (the observed measurement) back to its causes (the true, hidden state) [247]. This is convenient in situations in which the measurements space is much more complex than the state space and a forward measurement model is hard to come up with [247].[29] Under the Markov assumption and a double application of Bayes' theorem, we obtain

$$\begin{aligned} p(X|\boldsymbol{z}_{1:k}) &= \frac{p(\boldsymbol{z}_k|X, \boldsymbol{z}_{1:k-1})p(X|\boldsymbol{z}_{1:k-1})}{p(\boldsymbol{z}_k|\boldsymbol{z}_{1:k-1})} = \frac{p(\boldsymbol{z}_k|X)p(X|\boldsymbol{z}_{1:k-1})}{p(\boldsymbol{z}_k|\boldsymbol{z}_{1:k-1})} \\ &= \frac{p(X|\boldsymbol{z}_k)p(\boldsymbol{z}_k)p(X|\boldsymbol{z}_{1:k-1})}{p(X)p(\boldsymbol{z}_k|\boldsymbol{z}_{1:k-1})} \end{aligned} \tag{3.72}$$

for the event $X =$ true. Exactly the same is done for the opposite event, which analogously results in

$$p(\neg X|\boldsymbol{z}_{1:k}) = \frac{p(\neg X|\boldsymbol{z}_k)p(\boldsymbol{z}_k)p(\neg X|\boldsymbol{z}_{1:k-1})}{p(\neg X)p(\boldsymbol{z}_k|\boldsymbol{z}_{1:k-1})}. \tag{3.73}$$

Dividing (3.72) by (3.73) allows to cancel the hard to calculate fraction $\frac{p(\boldsymbol{z}_k)}{p(\boldsymbol{z}_k|\boldsymbol{z}_{1:k-1})}$ and results in the so-called odds of the binary state

$$\begin{aligned} \frac{p(X|\boldsymbol{z}_{1:k})}{p(\neg X|\boldsymbol{z}_{1:k})} &= \frac{p(X|\boldsymbol{z}_k)}{p(\neg X|\boldsymbol{z}_k)} \frac{p(X|\boldsymbol{z}_{1:k-1})}{p(\neg X|\boldsymbol{z}_{1:k-1})} \frac{p(\neg X)}{p(X)} \\ &= \frac{p(X|\boldsymbol{z}_k)}{1 - p(X|\boldsymbol{z}_k)} \frac{p(X|\boldsymbol{z}_{1:k-1})}{1 - p(X|\boldsymbol{z}_{1:k-1})} \frac{1 - p(X)}{p(X)}, \end{aligned} \tag{3.74}$$

which is the ratio of the probability of the event $X =$ true divided by the probability of its negate $X =$ false [247]. To avoid truncation problems arising from probabilities close to zero or one and to turn multiplications into additions, the so-called log odds representation $l_k(X)$ is advantageous.

[29] It is, for example, difficult to specify a high-dimensional distribution over all camera images that show a vehicle, whereas it is much easier to define a function that calculates the probability of a vehicle existing or not existing (binary state space) in a camera image (high-dimensional measurement space).

It is defined as the logarithm of (3.74) [247], therefore

$$
\begin{aligned}
l_k(X) &= \log\frac{p(X|\boldsymbol{z}_k)}{1-p(X|\boldsymbol{z}_k)} + \log\frac{p(X|\boldsymbol{z}_{1:k-1})}{1-p(X|\boldsymbol{z}_{1:k-1})} + \log\frac{1-p(X)}{p(X)} \\
&= \log\frac{\overbrace{p(X|\boldsymbol{z}_k)}^{\text{Inverse sensor model}}}{1-p(X|\boldsymbol{z}_k)} + l_{k-1}(X) - \log\frac{\overbrace{p(X)}^{\text{Prior}}}{1-p(X)}.
\end{aligned}
\tag{3.75}
$$

Thus, the final filter result in form of the updated log odds ratio $l_k(X)$ at time step k is calculated by adding the log odds form of the inverse sensor model $p(X|\boldsymbol{z}_k)$ to the filter result of the previous time step $l_{k-1}(X)$ and subtracting the log odds ratio of the prior state probability $p(X)$. To start the recursion, $l_0(X)$ is specified with the help of the prior probability according to

$$l_0(X) = \log\frac{p(X)}{1-p(X)}. \tag{3.76}$$

If no prior state knowledge is available, complete ignorance can be expressed by setting the non-informative prior $p(X) = 0.5$. In this case, the last term in (3.75) vanishes. Under consideration of natural logarithms, the posterior state in its familiar form can finally be recovered from $l_k(X)$ by

$$p(X|\boldsymbol{z}_{1:k}) = 1 - \frac{1}{1+e^{l_k(X)}}, \tag{3.77}$$

which follows directly from the definition of the log odds ratio.

Now that the relevant Bayesian filter algorithms are described, the filter applications *target tracking* and *occupancy grid mapping* are presented in the following, which both play a major role in the generation of the proposed metric environment representation.

3.2 Target Tracking

Target tracking is an application of Bayesian filtering, which aims at the determination of tracks from a sequence of measurements. A track is a state trajectory estimated from a set of measurements that have been associated with the same target [17]. In the intelligent vehicles context, target tracking is most often about determining states, such as position, velocity, or acceleration, of other dynamic objects, e.g. vehicles. Target tracking is, however, not limited to these kinds of objects but may also be used to determine states of other environment entities such as lane markings or the shape of available free space.

The classical tracking architecture is illustrated in Fig. 3.7. Noisy sensor measurements are compared with predicted measurements of already established tracks in a data association step. If an association is possible, the corresponding measurement is used to update the state of the track under consideration. If a measurement cannot be associated, it is used to initialize a new track. The track management is responsible for maintaining all current tracks, discarding old or weak tracks, and handling new tracks. It holds the actual tracks with estimated states and error covariances readily available for subsequent algorithms. The tracks' state prediction is carried out with the help of a motion model described by a system dynamic equation (3.9a), whereas the measurement prediction is realized by a measurement model (sensor model) described by a measurement equation (3.9b) to enable the necessary comparison between predicted and real sensor measurements.

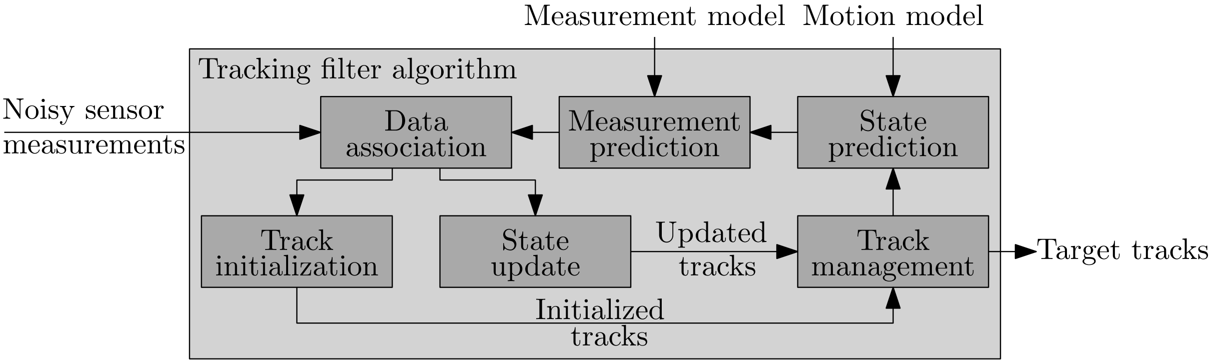


Figure 3.7: Classical tracking architecture.

Consequently, target tracking is wider in scope than pure Bayesian filtering, mainly because it additionally involves the problem of data association [17], which is the topic of the next subsection.

3.2.1 Tracking Filters and Data Association

Whereas pure filtering is only concerned with uncertainty in measured values and the estimation of hidden states, data association is concerned with uncertainty in measurement origin. In target tracking applications, it is not obvious which measurement belongs to which object. These ambiguities become even more pronounced in the presence of false alarms (clutter), which are generally random in number, location, and intensity.

Validation Gates

The common denominator in all data association methods is the exploitation of the concept of validation gates. A validation gate is a region $\mathcal{V}_k$ within the measurement space that likely includes the true measurement belonging to the track under consideration. The most common choice is an elliptical validation gate

$$\mathcal{V}_k(\gamma_\mathrm{G}) = \{\boldsymbol{z}_k \mid (\boldsymbol{z}_k - \hat{\boldsymbol{z}}_k^-)^T \boldsymbol{S}_k^{-1} \underbrace{(\boldsymbol{z}_k - \hat{\boldsymbol{z}}_k^-)}_{\boldsymbol{v}_k} \leq \gamma_\mathrm{G}\}, \tag{3.78}$$

centered at the predicted measurement $\hat{\boldsymbol{z}}_k^-$ with associated innovation covariance matrix $\boldsymbol{S}_k$ [17]. It includes all points that have a squared Mahalanobis distance, the normalized innovation squared, smaller or equal to a gate threshold γ_G. The threshold is obtained from the inverse chi-square cumulative distribution with a chosen gate probability $P_\mathrm{G} = P(\boldsymbol{z}_k \in \mathcal{V}_k(\gamma_\mathrm{G}))$ and $\dim(\boldsymbol{z}_k) = q$ degrees of freedom [17]. This follows from the assumption that the innovation $\boldsymbol{v}_k$ between the true and the predicted measurement is normally distributed around zero with covariance matrix $\boldsymbol{S}_k$, which is the case for a properly working KF. Therefore, this kind of gating exploits the full probabilistic knowledge available at that time as opposed to, for example, methods just relying on the Euclidean distance.[30] The volume of the validation region $\mathcal{V}_k$ is

$$V_k = V_\mathrm{U}(q)|\gamma_\mathrm{G}\boldsymbol{S}_k|^{\frac{1}{2}}, \tag{3.79}$$

with the volume of the q-dimensional unit hypersphere

$$V_\mathrm{U}(q) = \frac{\pi^{\frac{1}{2}q}}{\Gamma(\frac{1}{2}q+1)} = \begin{cases} \frac{\pi^{\frac{q}{2}}}{(\frac{q}{2})!}, & q \text{ even}, \\ \frac{2^{q+1}(\frac{q+1}{2})!\pi^{\frac{q-1}{2}}}{(q+1)!}, & q \text{ odd}, \end{cases} \tag{3.80}$$

and gamma function $\Gamma(\cdot)$ [17]. Only measurements within $\mathcal{V}_k$ are considered for a filter correction or track update step, respectively. They are said to be gated or validated and form the validated measurement set $\mathcal{Z}_{\mathrm{v},k} = \left\{\boldsymbol{z}_{m,k}\right\}_{m=1}^{N_\mathrm{v}}$ for a track. The remaining measurements can be used to initialize new tracks. Fig. 3.8 shows an exemplary, two-dimensional validation gate and an incoming measurement set at time step k with three validated and two non-validated measurements and associated innovations.

[30] The Mahalanobis distance reduces to the Euclidean distance if $\boldsymbol{S}_k = \mathbb{1}$.

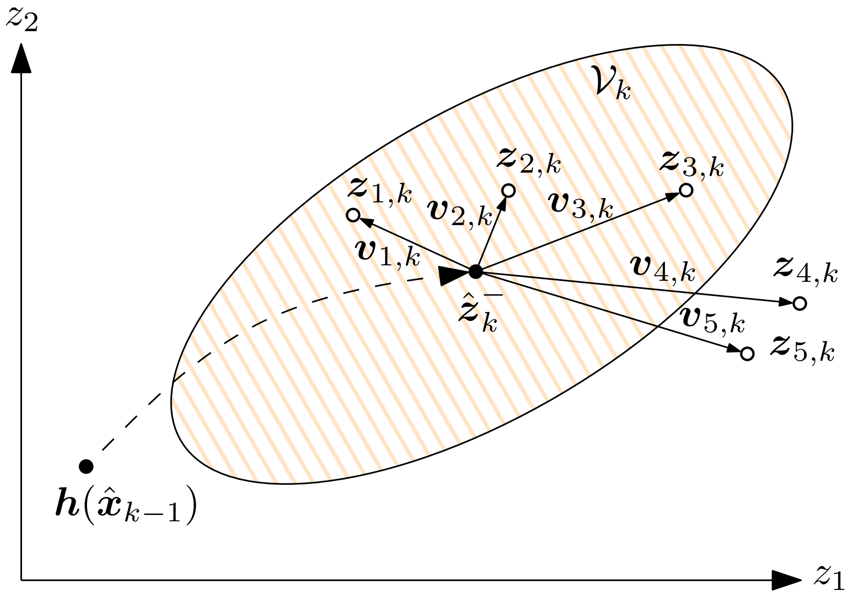


Figure 3.8: Two-dimensional, elliptical validation gate centered at the predicted measurement $\hat{\boldsymbol{z}}_k^-$ with incoming measurement set $\{\boldsymbol{z}_{m,k}\}_{m=1}^5$ and associated innovation vectors $\boldsymbol{v}_{m,k}$. Note that all points on the boundary of $\mathcal{V}_k$ have the same Mahalanobis distance to $\hat{\boldsymbol{z}}_k^-$, defined by the gate threshold. All measurements within $\mathcal{V}_k$ are validated, so that $\mathscr{Z}_{\mathrm{v},k} = \left\{\boldsymbol{z}_{m,k}\right\}_{m=1}^3$.

At this point, the importance of reasonable motion and measurement models becomes apparent. A deviation between reality and what the model expects may lead to situations in which true measurements are outside the gate and therefore discarded, which might entail track losses. Although this could be made less likely by increasing the gate size, e.g. by either increasing the gate threshold or the innovation covariance by raising the measurement noise, this approach would result in a larger number of clutter measurements falling inside the gate as a negative side effect. This would not only waste processing time as more measurements have to be considered but also complicate the separation of nearby objects due to greater gate overlap.

Tracking Filters

Different data association methods as well as their corresponding tracking filters differ in the way they extract information from the validated measurements. They can be classified into three categories.

First, methods differ in the way they assign measurements to tracks. A hard assignment is present if at most one measurement is used to update the target's state estimate at each time step. Consequently, these algorithms are either completely correct or completely wrong. In contrast, soft assignment algorithms can perform a weighted track update with more than one measurement. They are never completely correct or wrong but generally more robust.

Second, one can distinguish between algorithms using single-frame (also called single-scan) and multi-frame assignments. In single-frame algorithms, an assignment decision (be it soft or hard) has to be made in each time step immediately and cannot be withdrawn later, whereas multi-frame algorithms can delay the decision until more information is available and ambiguities are resolved.

Third, algorithms differ in whether they explicitly consider multi-target scenarios, thus perform data association jointly over all targets, or treat each target individually.[31]

Tracking filter algorithms are combined with or include the previously mentioned Bayesian filters of Table 3.1, p. 27, and are named after their data association scheme. The most prominent methods are the following:

- *Nearest Neighbor Standard Filters* (NNSF) [17] are single-target tracking filters that update the track with exactly the one measurement (hard assignment) with the smallest Mahalanobis distance to the predicted measurement even if this measurement is already used to update another track. NNSFs are well-suited in case of accurate prediction and measurement models for well-separated targets and low clutter scenarios.

- *Global Nearest Neighbor Filters* (GNNF) [53] are multi-target tracking filters that update the state with only one measurement (hard assignment) that is the result of solving a linear assignment problem globally, e.g. via the Munkres or, nowadays more common, the Auction algorithm [34]. GNNFs are well-suited in case of accurate prediction and measurement models and low clutter scenarios.

- *Probabilistic Data Association Filters* (PDAF) [15, 19] are single-target tracking filters that perform a weighted update with possibly several measurements (soft assignment) in the vicinity of a predicted measurement for each target individually. They are target-oriented

[31] Note that single-target filters can also be run in multi-target scenarios but with degraded performance. If targets are well-separated in the measurement space or otherwise distinguishable, a single-target filter is sufficient and explicit multi-target schemes just waste computational resources.

and evaluate the measurement-to-target association probabilities for the latest set of measurements. PDAFs are well-suited in case of noisy models, well-separated targets, and existing clutter.

- *Joint Probabilistic Data Association Filters* (JPDAF) [84] are multi-target extensions of the PDAF that perform a weighted update with possibly several measurements (soft assignment) simultaneously considering all targets. JPDAFs are well-suited in case of noisy models and existing clutter. Various extensions exist such as the JPDAF with merged measurements or the JPDA coupled filter. The former can cope with unresolved measurements in cases in which several close targets give rise to only one measurement. The latter is suitable for situations, in which targets share measurements within their validation gates for several time steps. This results in correlated target state estimation errors that are treated by estimating an augmented, stacked state vector of all targets [17].
- *Multi Hypothesis Tracking* (MHT) *Filters* [33, 34, 206] are multi-target tracking filters that evaluate the probabilities that there is a target from which a specific measurement sequence originated. They thus perform a target-to-measurement sequence association. As an optimal solution, the MHT evaluates every possible measurement sequence so that each hypothesis is considered, and subsequently performs a hard assignment with the most likely. This optimal solution is, however, computationally intractable due to the exponentially increasing computational burden. Therefore, MHT Filters are approximated in practice, e.g. via pruning strategies, but are still demanding from a computational point of view and with regard to implementation effort. They are well-suited for tracking closely spaced targets through clutter and additionally provide straightforward track initialization procedures.

In Table 3.2, these filters are classified into the mentioned categories and compared with respect to their clutter handling capabilities (CH) and processing speed (PS).

In the following, we have a closer look on the later employed *Probabilistic Data Association* (PDA) method as employed in the PDAF as well as on the combination of an IMM with underlying UKFs and PDAs, which provides a way to recursively estimate states and mode probabilities of targets, each of which is described by a jump Markov nonlinear system, in the presence of clutter.

Table 3.2: Tracking filter algorithms.
$--$ very low; $-$ low; $+$ high; $++$ very high

Filter	Soft Assignment	Multi-Frame	Multi-Target	CH	PS
NNSF	✗	✗	✗	✗	$++$
GNNF	✗	✗	✓	✗	$+$
PDAF	✓	✗	✗	✓	$+$
JPDAF	✓	✗	✓	✓	$-$
MHT	✗	✓	✓	✓	$--$

3.2.2 Probabilistic Data Association Filter

The PDAF [15,127], originally introduced in [19], differs from the standard KF equations only in the correction step to the effect that it performs a weighted update with all N_v validated measurements, summarized in the validated measurement set $\mathcal{Z}_{\mathrm{v},k} = \left\{\boldsymbol{z}_{m,k}\right\}_{m=1}^{N_\mathrm{v}}$. The weights are the measurement-to-target association probabilities $\beta_{m,k} = p(\theta_{m,k}|\mathcal{Z}_{\mathrm{v},k})$ of the mutually exclusive[32] and exhaustive[33] association events

- $\theta_{m,k}$: The measurement $\boldsymbol{z}_{m,k} \in \mathcal{Z}_{\mathrm{v},k}$ is the sole target-oriented measurement, the others are from clutter, and
- $\theta_{0,k}$: None of the measurements in $\mathcal{Z}_{\mathrm{v},k}$ is target-oriented.

For the derivation of the association probabilities, the algorithm implicitly assumes the following conditions to be fulfilled [17]:

- A target track has been initialized.
- Only a single target exists within the validation gate of the track under consideration. Consequently, there is no persistent interference, e.g. from neighboring targets.
- The target's predicted state and measurement densities are Gaussian. This is thus only fulfilled exactly for a KF prediction step under LGS assumptions.
- Target detections occur independently over time with known detection probability P_D.

[32] Mutually exclusive events cannot occur at the same time.

[33] Probabilities of exhaustive events sums up to one.

- At most one of the validated measurements can be originated from the target under consideration. The remaining ones are false alarms modeled as independent and identically distributed random variables with uniform spatial distribution in the measurement space. Therefore, a target is not allowed to give rise to several measurements.

For the number of false measurements, it is common to consider a diffuse prior clutter model, which uses the number of returns in the track gate to estimate the clutter density and which is suitable for heterogeneous clutter environments as any number of false measurements is considered equiprobable [34]. This leads to the non-parametric PDAF[34], in which the association probabilities $\beta_{m,k}$ are given by [17]

$$\beta_{m,k} = \begin{cases} \frac{e_m}{b+\sum_{i=1}^{N_\mathrm{v}} e_i}, & m \in \{1,\dots,N_\mathrm{v}\}, \\ \frac{b}{b+\sum_{i=1}^{N_\mathrm{v}} e_i}, & m = 0, \end{cases} \tag{3.81a}$$

$$e_m = e^{-\frac{1}{2}\boldsymbol{v}_{m,k}^T \boldsymbol{S}_k^{-1} \boldsymbol{v}_{m,k}}, \tag{3.81b}$$

$$b = \left(\frac{2\pi}{\gamma_\mathrm{G}}\right)^{\frac{q}{2}} \frac{N_\mathrm{v}(1-P_\mathrm{D}P_\mathrm{G})}{V_\mathrm{U}(q)P_\mathrm{D}}. \tag{3.81c}$$

The individual innovations $\boldsymbol{v}_{m,k}$ and the innovation covariance matrix corresponding to the true measurement $\boldsymbol{S}_k$ originate from a KF measurement prediction step (3.28), p. 29, whereas $\gamma_\mathrm{G}, V_\mathrm{U}(q)$, and P_G are given by the validation gate calculation as already explained in Section 3.2.1.

Once the data association probabilities are available[35], the PDAF performs a weighted state update according to

$$\hat{\boldsymbol{x}}_k = \hat{\boldsymbol{x}}_k^- + \boldsymbol{K}_k \boldsymbol{v}_k = \hat{\boldsymbol{x}}_k^- + \boldsymbol{K}_k \underbrace{\sum_{m=1}^{N_\mathrm{v}} \beta_{m,k} \underbrace{(\boldsymbol{z}_{m,k} - \hat{\boldsymbol{z}}_k^-)}_{\boldsymbol{v}_{m,k}}}_{\boldsymbol{v}_k}, \tag{3.82}$$

with the combined innovation $\boldsymbol{v}_k$ [17]. Therefore, the less likely a measurement belongs to the track under consideration the lesser it is weighted

[34] The parametric PDAF, in contrast, assumes a Poisson clutter model for the number of false measurements.

[35] The multi-target extension JPDAF only differs in the way the data association probabilities are calculated. The correction step remains the same.

in the correction step. The covariance update [17]

$$\boldsymbol{P}_k = \beta_{0,k}\boldsymbol{P}_k^- + (1-\beta_{0,k})(\boldsymbol{P}_k^- - \boldsymbol{K}_k\boldsymbol{S}_k\boldsymbol{K}_k^T) \\ + \boldsymbol{K}_k\left(\sum_{m=1}^{N_\mathrm{v}} \beta_{m,k}\boldsymbol{v}_{m,k}\boldsymbol{v}_{m,k}^T - \boldsymbol{v}_k\boldsymbol{v}_k^T\right)\boldsymbol{K}_k^T \tag{3.83a}$$

can likewise be intuitively explained. As with probability $\beta_{0,k}$, none of the measurements are target-oriented, just the predicted covariance $\boldsymbol{P}_k^-$ appears in the first summand as if no correction had been performed. With converse probability $(1-\beta_{0,k})$, one of the measurements is target-oriented, so that the standard KF covariance correction (3.32b), p. 29, is applied in the second summand. The third summand is called the spread of the innovations, a positive semi-definite term, which increases the covariance and accounts for data association uncertainty [17]. The PDAF is a nonlinear filter even under LGS assumptions due to the multiplications of the innovations with the association probabilities in (3.82), which in turn depend on the innovations in (3.81). From a computational point of view, the PDAF is only slightly more demanding than the standard KF [17]. The next section presents how the PDAF can advantageously be combined with IMMs and UKFs.

3.2.3 IMM-UK-PDAF

The IMM-PDAF, a combination of an IMM Filter with PDA[36], has originally been introduced in [110]. Numerous articles, e.g. [53, 127], point out its ability to track maneuvering targets in clutter and even attest the IMM-PDAF comparable tracking performance to the computationally orders of magnitude costlier MHT Filters [127]. In this section, an extended form is presented in which this filter is accompanied by underlying UKFs to handle nonlinearities. The filter is termed *Interacting Multiple Model Unscented Kalman Probabilistic Data Association Filter* (IMM-UK-PDAF) in the following and is mainly based upon combining information scattered through [16–18, 34, 118, 153, 167]. The IMM-UK-PDAF provides an approximate solution for recursively estimating states and mode probabilities of well-separated, maneuvering targets, each of which is described by a jump Markov *nonlinear* system, in the presence of clutter.

[36] Combinations of IMM and JPDAF [38] as well as IMM and JPDA coupled filters [251] exist as well but both are computationally more demanding and not necessary for the task at hand.

We consider the j-th model of a model set $\mathcal{M} = \left\{M_j\right\}_{j=1}^{r}$ to be given by

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}_j(\boldsymbol{x}_k, \boldsymbol{u}_k) + \boldsymbol{w}_{j,k}, \tag{3.84a}$$

$$\boldsymbol{z}_k = \boldsymbol{h}_j(\boldsymbol{x}_k) + \boldsymbol{v}_{j,k}, \tag{3.84b}$$

with input vector $\boldsymbol{u}_k \in \mathbb{R}^p$, state vector $\boldsymbol{x}_k \in \mathbb{R}^n$, measurement vector $\boldsymbol{z}_k \in \mathbb{R}^q$, process noise vector $\boldsymbol{w}_{j,k} \in \mathbb{R}^n$, measurement noise vector $\boldsymbol{v}_{j,k} \in \mathbb{R}^q$, system function $\boldsymbol{f}_j(\cdot)$, and measurement function $\boldsymbol{h}_j(\cdot)$. The noise sequences are mutually independent, zero-mean, white, and Gaussian with process noise covariance matrix $\boldsymbol{Q}_{j,k} \in \mathbb{R}^{n\times n}$ and measurement noise covariance matrix $\boldsymbol{R}_{j,k} \in \mathbb{R}^{q\times q}$. The initial state density for each model is considered Gaussian, and the evolution of the hybrid system among the r models is supposed to follow a first-order Markov chain with time-invariant Markovian model transition probability matrix $\boldsymbol{\Pi} \in \mathbb{R}^{r\times r}$.

The explanation of the tracking filter is subdivided into six main steps: interaction, model-specific filter prediction, measurement validation, model-specific filter update, mode probability update, and combination. A visual overview of these steps is given in Fig. 3.9 as an extension of Fig. 3.5, p. 42.

Interaction

In the interaction step, the individual state and covariance estimates of the underlying filters of the last time step $\hat{\boldsymbol{x}}_{j,k-1}, \boldsymbol{P}_{j,k-1}$ are probabilistically mixed to form a single initial state $\hat{\boldsymbol{x}}^{\star}_{j,k-1}$ and covariance $\boldsymbol{P}^{\star}_{j,k-1}$ for each filter j according to

$$\hat{\boldsymbol{x}}^{\star}_{j,k-1} = \sum_{i=1}^{r} \mu_{(i|j),k-1} \hat{\boldsymbol{x}}_{i,k-1}, \tag{3.85a}$$

$$\boldsymbol{P}^{\star}_{j,k-1} = \sum_{i=1}^{r} \mu_{(i|j),k-1} \left(\boldsymbol{P}_{i,k-1} + (\hat{\boldsymbol{x}}_{i,k-1} - \hat{\boldsymbol{x}}^{\star}_{j,k-1})(\hat{\boldsymbol{x}}_{i,k-1} - \hat{\boldsymbol{x}}^{\star}_{j,k-1})^T\right), \tag{3.85b}$$

with mixing probabilities

$$\mu_{(i|j),k-1} = \frac{p_{ij}\mu_{i,k-1}}{\mu^{-}_{j,k}} \tag{3.86}$$

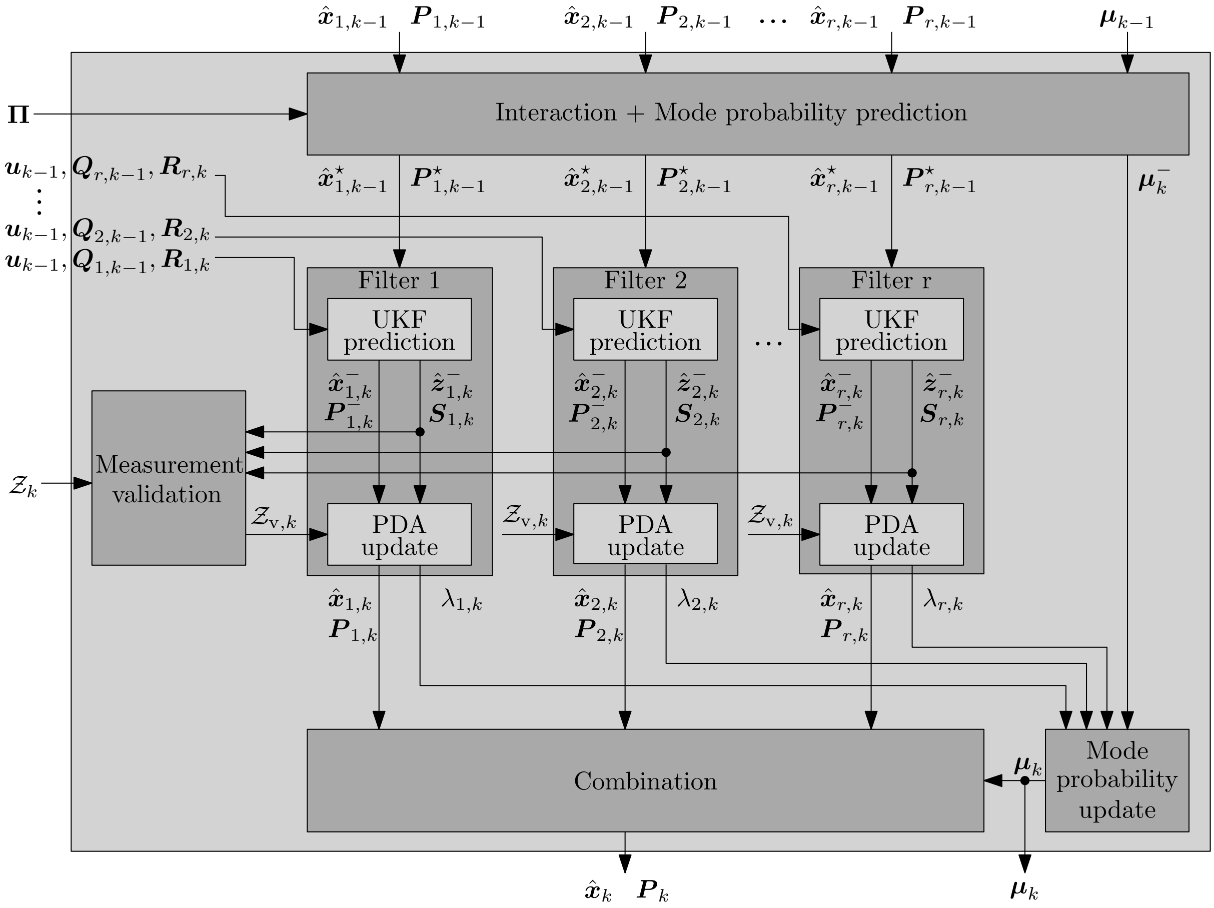

$\hat{\boldsymbol{x}}_{1,k-1}$ $\boldsymbol{P}_{1,k-1}$
$\hat{\boldsymbol{x}}_{2,k-1}$ $\boldsymbol{P}_{2,k-1}$
$\cdots$
$\hat{\boldsymbol{x}}_{r,k-1}$ $\boldsymbol{P}_{r,k-1}$
$\boldsymbol{\mu}_{k-1}$
$\boldsymbol{\Pi}$
Interaction + Mode probability prediction
$\boldsymbol{u}_{k-1}, \boldsymbol{Q}_{r,k-1}, \boldsymbol{R}_{r,k}$
$\vdots$
$\boldsymbol{u}_{k-1}, \boldsymbol{Q}_{2,k-1}, \boldsymbol{R}_{2,k}$
$\boldsymbol{u}_{k-1}, \boldsymbol{Q}_{1,k-1}, \boldsymbol{R}_{1,k}$
$\hat{\boldsymbol{x}}^{\star}_{1,k-1}$ $\boldsymbol{P}^{\star}_{1,k-1}$
$\hat{\boldsymbol{x}}^{\star}_{2,k-1}$ $\boldsymbol{P}^{\star}_{2,k-1}$
$\hat{\boldsymbol{x}}^{\star}_{r,k-1}$ $\boldsymbol{P}^{\star}_{r,k-1}$
$\boldsymbol{\mu}_k^-$
Filter 1
Filter 2
Filter r
UKF prediction
UKF prediction
$\cdots$
UKF prediction
$\hat{\boldsymbol{x}}^-_{1,k}$ $\boldsymbol{P}^-_{1,k}$
$\hat{\boldsymbol{z}}^-_{1,k}$ $\boldsymbol{S}_{1,k}$
$\hat{\boldsymbol{x}}^-_{2,k}$ $\boldsymbol{P}^-_{2,k}$
$\hat{\boldsymbol{z}}^-_{2,k}$ $\boldsymbol{S}_{2,k}$
$\hat{\boldsymbol{x}}^-_{r,k}$ $\boldsymbol{P}^-_{r,k}$
$\hat{\boldsymbol{z}}^-_{r,k}$ $\boldsymbol{S}_{r,k}$
$\mathcal{Z}_k$
Measurement validation
$\mathcal{Z}_{\mathrm{v},k}$
PDA update
$\mathcal{Z}_{\mathrm{v},k}$
PDA update
$\mathcal{Z}_{\mathrm{v},k}$
PDA update
$\hat{\boldsymbol{x}}_{1,k}$ $\boldsymbol{P}_{1,k}$
$\lambda_{1,k}$
$\hat{\boldsymbol{x}}_{2,k}$ $\boldsymbol{P}_{2,k}$
$\lambda_{2,k}$
$\hat{\boldsymbol{x}}_{r,k}$ $\boldsymbol{P}_{r,k}$
$\lambda_{r,k}$
Combination
$\boldsymbol{\mu}_k$
Mode probability update
$\hat{\boldsymbol{x}}_k$ $\boldsymbol{P}_k$
$\boldsymbol{\mu}_k$

Figure 3.9: Overview of one cycle of the IMM-UK-PDAF at time step k.

and predicted, a priori mode probabilities $\boldsymbol{\mu}_k^- = (\mu_{1,k}^- \dots \mu_{r,k}^-)^T$ given by

$$\mu_{j,k}^- = \sum_{i=1}^{r} p_{ij}\mu_{i,k-1}. \tag{3.87}$$

This step is identical to the standard IMM interaction, see p. 42.

Model-Specific Filter Prediction

Subsequently, these initial states and covariances are predicted to the current time step k for each model j individually to obtain predicted states $\hat{\boldsymbol{x}}_{j,k}^-$ and error covariances $\boldsymbol{P}_{j,k}^-$ as well as predicted measurements $\hat{\boldsymbol{z}}_{j,k}^-$ and innovation covariances $\boldsymbol{S}_{j,k}$. Due to nonlinearities, this is realized by means of an UKF prediction in analogy to Section 3.1.5, p. 33:

$$\boldsymbol{\mathcal{X}}_{j,k-1} = \left(\hat{\boldsymbol{x}}^{\star}_{j,k-1} \quad \gamma_{\mathrm{U}} \sqrt{\boldsymbol{P}^{\star}_{j,k-1}} \oplus \hat{\boldsymbol{x}}^{\star}_{j,k-1} \quad -\gamma_{\mathrm{U}} \sqrt{\boldsymbol{P}^{\star}_{j,k-1}} \oplus \hat{\boldsymbol{x}}^{\star}_{j,k-1} \right), \tag{3.88a}$$

$$\boldsymbol{\mathcal{X}}^{*}_{j,i,k} = \boldsymbol{f}_j(\boldsymbol{\mathcal{X}}_{j,i,k-1}, \boldsymbol{u}_{k-1}), \qquad \forall i \in \{0,\ldots,2n\}, \tag{3.88b}$$

$$\hat{\boldsymbol{x}}^{-}_{j,k} = \sum_{i=0}^{2n} w_i^{(m)} \boldsymbol{\mathcal{X}}^{*}_{j,i,k}, \tag{3.88c}$$

$$\boldsymbol{P}^{-}_{j,k} = \sum_{i=0}^{2n} w_i^{(c)} \left(\boldsymbol{\mathcal{X}}^{*}_{j,i,k} - \hat{\boldsymbol{x}}^{-}_{j,k}\right) \left(\boldsymbol{\mathcal{X}}^{*}_{j,i,k} - \hat{\boldsymbol{x}}^{-}_{j,k}\right)^T + \boldsymbol{Q}_{j,k-1}, \tag{3.88d}$$

$$\boldsymbol{\mathcal{X}}_{j,k} = \left(\hat{\boldsymbol{x}}^{-}_{j,k} \quad \gamma_{\mathrm{U}} \sqrt{\boldsymbol{P}^{-}_{j,k}} \oplus \hat{\boldsymbol{x}}^{-}_{j,k} \quad -\gamma_{\mathrm{U}} \sqrt{\boldsymbol{P}^{-}_{j,k}} \oplus \hat{\boldsymbol{x}}^{-}_{j,k} \right), \tag{3.88e}$$

$$\boldsymbol{\mathcal{Z}}_{j,i,k} = \boldsymbol{h}_j(\boldsymbol{\mathcal{X}}_{j,i,k}), \qquad \forall i \in \{0,\ldots,2n\}, \tag{3.88f}$$

$$\hat{\boldsymbol{z}}^{-}_{j,k} = \sum_{i=0}^{2n} w_i^{(m)} \boldsymbol{\mathcal{Z}}_{j,i,k}, \tag{3.88g}$$

$$\boldsymbol{S}_{j,k} = \sum_{i=0}^{2n} w_i^{(c)} \left(\boldsymbol{\mathcal{Z}}_{j,i,k} - \hat{\boldsymbol{z}}^{-}_{j,k}\right) \left(\boldsymbol{\mathcal{Z}}_{j,i,k} - \hat{\boldsymbol{z}}^{-}_{j,k}\right)^T + \boldsymbol{R}_{j,k}, \tag{3.88h}$$

with

$$w_0^{(m)} = \frac{\lambda_{\mathrm{U}}}{n+\lambda_{\mathrm{U}}}, \quad w_0^{(c)} = \frac{\lambda_{\mathrm{U}}}{n+\lambda_{\mathrm{U}}} + (1-\alpha_{\mathrm{U}}^2+\beta_{\mathrm{U}}), \tag{3.89a}$$

$$w_i^{(m)} = w_i^{(c)} = \frac{1}{2(n+\lambda_{\mathrm{U}})}, \qquad \forall i \in \{1,\ldots,2n\}, \tag{3.89b}$$

$$\lambda_{\mathrm{U}} = \alpha_{\mathrm{U}}^2(n+\kappa_{\mathrm{U}}) - n, \tag{3.89c}$$

$$\gamma_{\mathrm{U}} = \sqrt{n+\lambda_{\mathrm{U}}}. \tag{3.89d}$$

Measurement Validation

The prediction step is followed by the measurement validation step. For validating a measurement for all models of a track, it must lie within an elliptical validation gate

$$\mathcal{V}_k(\gamma_{\mathrm{G}}) = \{\boldsymbol{z}_k \mid (\boldsymbol{z}_k - \hat{\boldsymbol{z}}^{-}_{j_r,k})^T \boldsymbol{S}^{-1}_{j_r,k} (\boldsymbol{z}_k - \hat{\boldsymbol{z}}^{-}_{j_r,k}) \leq \gamma_{\mathrm{G}}\}, \tag{3.90}$$

centered at $\hat{\boldsymbol{z}}_{j_r,k}^-$, with gate threshold γ_G [17] and j_r normally calculated as [34]

$$j_r = \arg\max_{j \in \mathcal{M}} |\boldsymbol{S}_{j,k}|. \tag{3.91}$$

The gate is thus set up around the predicted measurement of the model with the largest, in terms of determinant, innovation covariance matrix, which it supposed to include the other gates to a large amount, see Fig. 3.10. It is important to employ a common validation region for all models of a track so that the model likelihoods are all conditioned on the same set of measurements [17]. The threshold γ_G is obtained from the inverse chi-square cumulative distribution with a chosen gate probability $P_\mathrm{G} = P(\boldsymbol{z}_k \in \mathcal{V}_k(\gamma_\mathrm{G}))$ and $\dim(\boldsymbol{z}_k) = q$ degrees of freedom according to [17]. The volume of the common validation region $\mathcal{V}_k$ is $V_k = V_\mathrm{U}(q)|\gamma_\mathrm{G}\boldsymbol{S}_{j_r,k}|^{\frac{1}{2}}$ with the volume of the q-dimensional unit hypersphere $V_\mathrm{U}(q) = \frac{\pi^{\frac{1}{2}q}}{\Gamma(\frac{1}{2}q+1)}$ and gamma function $\Gamma(\cdot)$ [17]. Consequently, we obtain one validated measurement set $\mathcal{Z}_{\mathrm{v},k} = \left\{\boldsymbol{z}_{m,k}\right\}_{m=1}^{N_\mathrm{v}}$ for all models of a track. Non-validated measurements can be used to initialize new tracks.

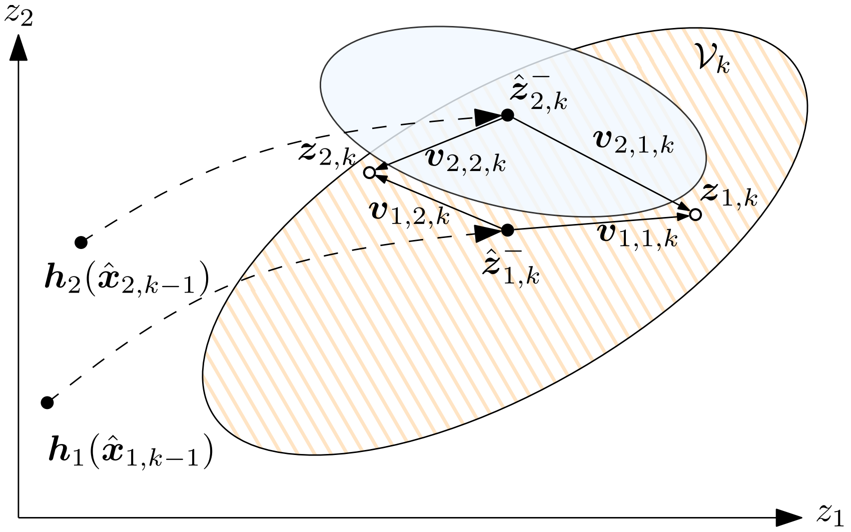


Figure 3.10: Measurement validation in a two-model IMM-UK-PDAF. Both measurements, $\boldsymbol{z}_{1,k}$ and $\boldsymbol{z}_{2,k}$, are validated for both models of this track because they lie in the validation region $\mathcal{V}_k$ of the model with the larger innovation covariance matrix visualized in orange. Note that still the individual innovations $\boldsymbol{v}_{j,m,k} = \boldsymbol{z}_{m,k} - \hat{\boldsymbol{z}}_{j,k}^-$ are used in the model-specific filter update step.

Model-Specific Filter Update

The measurement validation step is followed by a model-specific filtering. For this purpose, each track is updated with r PDAFs, which yield a posteriori state and covariance estimates, $\hat{\boldsymbol{x}}_{j,k}$ and $\boldsymbol{P}_{j,k}$, for each model by incorporating the current validated measurement set according to

$$\boldsymbol{C}_{j,k} = \sum_{i=0}^{2n} w_i^{(c)} \left(\boldsymbol{\mathcal{X}}_{j,i,k} - \hat{\boldsymbol{x}}_{j,k}^{-}\right) \left(\boldsymbol{\mathcal{Z}}_{j,i,k} - \hat{\boldsymbol{z}}_{j,k}^{-}\right)^T , \tag{3.92a}$$

$$\boldsymbol{K}_{j,k} = \boldsymbol{C}_{j,k}\boldsymbol{S}_{j,k}^{-1}, \tag{3.92b}$$

$$\hat{\boldsymbol{x}}_{j,k} = \hat{\boldsymbol{x}}_{j,k}^{-} + \boldsymbol{K}_{j,k}\boldsymbol{v}_{j,k} = \hat{\boldsymbol{x}}_{j,k}^{-} + \boldsymbol{K}_{j,k} \underbrace{\sum_{m=1}^{N_\text{v}} \beta_{j,m,k} \underbrace{(\boldsymbol{z}_{m,k} - \hat{\boldsymbol{z}}_{j,k}^{-})}_{\boldsymbol{v}_{j,m,k}}}_{\boldsymbol{v}_{j,k}}, \tag{3.92c}$$

$$\begin{aligned}\boldsymbol{P}_{j,k} &= \beta_{j,0,k}\boldsymbol{P}_{j,k}^{-} + (1 - \beta_{j,0,k})(\boldsymbol{P}_{j,k}^{-} - \boldsymbol{K}_{j,k}\boldsymbol{S}_{j,k}\boldsymbol{K}_{j,k}^T) \\ &\quad + \boldsymbol{K}_{j,k} \left(\sum_{m=1}^{N_\text{v}} \beta_{j,m,k}\boldsymbol{v}_{j,m,k}\boldsymbol{v}_{j,m,k}^T - \boldsymbol{v}_{j,k}\boldsymbol{v}_{j,k}^T \right) \boldsymbol{K}_{j,k}^T .\end{aligned} \tag{3.92d}$$

First, the cross covariance matrix $\boldsymbol{C}_{j,k}$ between predicted measurements and states is used together with the innovation covariance $\boldsymbol{S}_{j,k}$ to calculate the Kalman gain $\boldsymbol{K}_{j,k}$ for each model j such as in the normal UKF. The state and covariance update step then equals the conventional PDAF update step, see Section 3.2.2, which weights the influence of a measurement $\boldsymbol{z}_{m,k}$, with associated innovation $\boldsymbol{v}_{j,m,k}$, with its association probability $\beta_{j,m,k}$. For a considered nonparametric, uniform, diffuse prior clutter model, the association probability $\beta_{j,m,k}$ for $\boldsymbol{z}_{m,k}$ being the correct measurement given a target detection probability P_D is calculated as

$$\beta_{j,m,k} = \begin{cases} \frac{e_{j,m}}{b+\sum_{i=1}^{N_\text{v}} e_{j,i}}, & m \in \{1, \ldots, N_\text{v}\}, \\ \frac{b}{b+\sum_{i=1}^{N_\text{v}} e_{j,i}}, & m = 0, \end{cases} \tag{3.93a}$$

$$e_{j,m} = e^{-\frac{1}{2}\boldsymbol{v}_{j,m,k}^T \boldsymbol{S}_{j,k}^{-1} \boldsymbol{v}_{j,m,k}}, \tag{3.93b}$$

$$b = \left(\frac{2\pi}{\gamma_\text{G}}\right)^{\frac{q}{2}} \frac{N_\text{v}(1 - P_\text{D}P_\text{G})}{V_\text{U}(q)P_\text{D}}, \tag{3.93c}$$

in which $\beta_{j,0,k}$ is the probability that no measurements within the gate are correctly belonging to the track and model under consideration.

Mode Probability Update

The model likelihoods $\lambda_{j,k}$ are the Gaussian-uniform mixture

$$\lambda_{j,k} = p(\mathcal{Z}_{\mathrm{v},k}|M_{j,k}, \mathcal{Z}_{\mathrm{v},1:k-1}) = \frac{1-(P_{\mathrm{D}}P_{\mathrm{G}})}{(V_k)^{N_\mathrm{v}}} + \frac{P_{\mathrm{D}}V_k^{(1-N_\mathrm{v})}}{N_\mathrm{v}\sqrt{|2\pi \boldsymbol{S}_{j,k}|}} \sum_{m=1}^{N_\mathrm{v}} e_{j,m}, \tag{3.94}$$

equivalent to the standard IMM-PDAF for linear jump Markov systems, for a derivation see [17]. This is because the individual UKFs equivalently represent all individual measurement predictions as Gaussians just as the standard KF, and because the clutter is likewise assumed uniformly distributed in the measurement space. Thereupon, the mode probabilities can be updated based on how well the measurements fit to the model according to

$$\mu_{j,k} = \frac{\mu_{j,k}^{-}\lambda_{j,k}}{\sum\limits_{i=1}^{r} \mu_{i,k}^{-}\lambda_{i,k}}, \tag{3.95}$$

just as in the standard IMM Filter [17].

Combination

For final output purpose, the individual updated filter states are again combined to a final state and covariance estimate via [17]

$$\hat{\boldsymbol{x}}_k = \sum_{j=1}^{r} \mu_{j,k}\hat{\boldsymbol{x}}_{j,k}, \tag{3.96a}$$

$$\boldsymbol{P}_k = \sum_{j=1}^{r} \mu_{j,k}\left(\boldsymbol{P}_{j,k} + (\hat{\boldsymbol{x}}_{j,k} - \hat{\boldsymbol{x}}_k)(\hat{\boldsymbol{x}}_{j,k} - \hat{\boldsymbol{x}}_k)^T\right). \tag{3.96b}$$

So far, an overview of Bayesian filters and the filtering application *target-tracking* has been given, with focus on the later-employed algorithms. The next section deals with another Bayesian filter application, namely *occupancy grid mapping.*

3.3 Occupancy Grid Mapping

Occupancy grid mapping, originally introduced in [76,77], is a technique to generate consistent maps from noisy or uncertain measurement data under the assumption that the world is *static* and the vehicle *pose is known* [247]. It is therefore also termed *mapping with known poses.* The resulting grid map $\boldsymbol{m}$ tessellates the environment into finitely many grid cells m_i, each of which encode the probability of being occupied $p(m_i)$ depending on the sensor readings. Although the term occupancy may encode different facets, it most often relates to whether a mobile robot, such as an intelligent vehicle, can or cannot pass a specific area of the environment physically.

Occupancy grid mapping is a special case of the *Simultaneous Localization and Mapping* (SLAM) problem that requires a simultaneous pose estimation [13, 72, 247]. The DBN visualization of the different inference tasks of occupancy grid mapping and SLAM are contrasted in Figure 3.11.

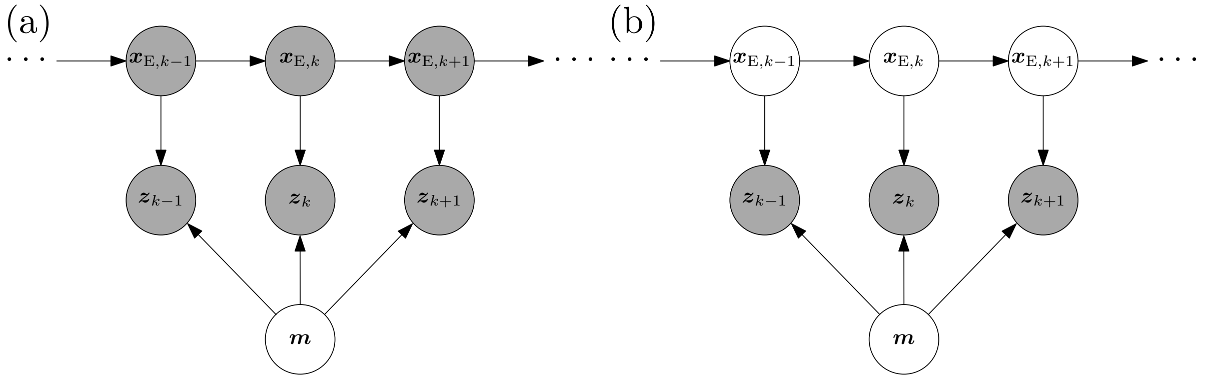

(a)
$\boldsymbol{x}_{\mathrm{E},k-1}$
$\boldsymbol{x}_{\mathrm{E},k}$
$\boldsymbol{x}_{\mathrm{E},k+1}$
$\boldsymbol{z}_{k-1}$
$\boldsymbol{z}_{k}$
$\boldsymbol{z}_{k+1}$
$\boldsymbol{m}$
(b)
$\boldsymbol{x}_{\mathrm{E},k-1}$
$\boldsymbol{x}_{\mathrm{E},k}$
$\boldsymbol{x}_{\mathrm{E},k+1}$
$\boldsymbol{z}_{k-1}$
$\boldsymbol{z}_{k}$
$\boldsymbol{z}_{k+1}$
$\boldsymbol{m}$

Figure 3.11: DBNs of the probabilistic mapping problem with observable nodes shaded in gray. In occupancy grid mapping (a), the sole hidden state is the map $\boldsymbol{m}$. The ego vehicle poses $\boldsymbol{x}_{\mathrm{E}}$ and environment sensor measurements $\boldsymbol{z}$ are assumed known. In SLAM (b), poses are to be estimated simultaneously with the map.

From a Bayesian point of view, the problem addressed by occupancy grid mapping is thus to determine the posterior distribution over maps $\boldsymbol{m}$ given the set of environment measurements $\boldsymbol{z}_{1:k}$ and the sequence of known poses of the ego vehicle $\boldsymbol{x}_{\mathrm{E},1:k}$ up to time step k as[37]

$$p(\boldsymbol{m}|\boldsymbol{z}_{1:k}, \boldsymbol{x}_{\mathrm{E},1:k}). \tag{3.97}$$

Due to the enormous complexity of calculating (3.97), the problem needs to be simplified to be computationally tractable.[38]

[37] Although the state of the map is discrete, maps are written in lower case in this thesis to distinguish them from the maneuver random variables.

[38] Suppose a very rough map of only 500 cells with binary state space, then the dimension of the map's joint state space is already $2^{500} \approx 3 \cdot 10^{150}$.

3.3.1 Traditional Mapping Solution

The traditional mapping solution breaks down the problem into a collection of problems with *binary*, *static* state, i.e. the estimation of $p(m_i|\boldsymbol{z}_{1:k}, \boldsymbol{x}_{\mathrm{E},1:k})$ for all grid cells m_i [77, 247]. Consequently, each cell is considered stochastically independent of all other cells and the posterior distribution over maps is approximated as the product of marginal distributions according to

$$p(\boldsymbol{m}|\boldsymbol{z}_{1:k}, \boldsymbol{x}_{\mathrm{E},1:k}) = \prod_i p(m_i|\boldsymbol{z}_{1:k}, \boldsymbol{x}_{\mathrm{E},1:k}). \tag{3.98}$$

Because of this factorization, the estimation of the occupancy probability of each grid cell can be realized by individual, separate BBFs as explained in Section 3.1.8. By applying the log odds representation of the posterior occupancy probability of each grid cell

$$l_k(m_i) = \log \frac{p(m_i|\boldsymbol{z}_{1:k}, \boldsymbol{x}_{\mathrm{E},1:k})}{1 - p(m_i|\boldsymbol{z}_{1:k}, \boldsymbol{x}_{\mathrm{E},1:k})}, \tag{3.99}$$

the recursive filter solution for solving (3.99) is consequently given by

$$l_k(m_i) = l_{k-1}(m_i) + \log \frac{\overbrace{p(m_i|\boldsymbol{z}_k, \boldsymbol{x}_{\mathrm{E},k})}^{\text{Inverse sensor model}}}{1 - p(m_i|\boldsymbol{z}_k, \boldsymbol{x}_{\mathrm{E},k})} - \log \frac{\overbrace{p(m_i)}^{\text{Prior}}}{1 - p(m_i)}, \tag{3.100}$$

in which the inverse sensor model specifies the distribution of the binary cell state variable as a function of the measurement and pose [247].[39]

3.3.2 Inverse Sensor Models

The inverse sensor model, see Section 3.1.8, necessary for calculating (3.100) can be learned from the forward sensor model via supervised

[39] There also exist grid mapping algorithms with *forward sensor models* of the form $p(\boldsymbol{z}_k|\boldsymbol{m}, \boldsymbol{x}_{\mathrm{E},k})$, which do not have to apply the simplifying factorization involved in (3.98) [246]. The forward mapping approach introduced in [246] is, however, not incremental but needs to pass through the data multiple times. Moreover, it is computationally orders of magnitude more demanding than mapping with inverse models and the result is likewise no full posterior distribution over maps but a maximum likelihood map estimated via expectation maximization. By conditioning on this map estimate, an approximate marginal posterior distribution of each grid cell's occupancy can, however, be recovered in a subsequent processing step. In the intelligent vehicles and ADAS domain, mapping with forward sensor models does not (yet) play any role.

learning [247], derived analytically from the forward sensor model in special cases [77, 145], or specified manually directly [86, 269]. For 2D range sensors, it is commonly assumed that the sensor outputs a noisy range and angle measurement of a detected obstacle.[40] The area in front of the detection is treated as (rather) free in the inverse sensor model, the area in the vicinity of the obstacle detection as (rather) occupied, and the area behind the obstacle as unknown. The latter is based on the assumption that a range sensor is incapable of providing information behind an obstacle due to occlusions.

The three regions are clearly visible in Fig 3.12, which shows an exemplary inverse sensor model of a 2D range sensor based on [77,86,269]. In front of the obstacle, the inverse sensor model is described by a linear, rising function, which stays below an occupancy value of 0.5. In this region, the corresponding cell occupancy probabilities are thus lowered in a BBF update as the corresponding log odds ratios of the inverse sensor model are negative. The positive gradient models the increased probability of missing a true target farther away [269] and consequently ensures that the corresponding occupancy probabilities are decreased less strongly in larger distances to the sensor. In the vicinity of the obstacle, the detection is blurred due to the Gaussian distributed angle and range measurements, whereas the inverse sensor model is equal to 0.5 behind the obstacle as no environment knowledge is provided there. Considering a non-informative map prior, i.e. $p(m_i) = 0.5 \Rightarrow \log \frac{p(m_i)}{1-p(m_i)} = 0$, occupancy grid mapping via (3.100) is thus performed by adding the log odds form of such an inverse sensor model to the log odds ratio of the posterior cell occupancy of the grid map of the last time step $l_{k-1}(m_i)$ for each grid cell m_i.

Inverse sensor models exist for sonars [77], radars [91], lidars [107, 131, 269], and stereo cameras [140]. Different sensors can be fused by mapping a separate grid with each sensor and subsequently combining the resulting maps, e.g. via the independent opinion pool method [27, 77] under a stochastic independence assumption between measurements of the different sensors. For the commonly assumed non-informative map prior, this is equivalent to adding the posterior log odds of all separately mapped grids [175].[41]

[40] Note that this type of output can be generated for all range sensors including sonars, lidars, radars, and stereo cameras after appropriate preprocessing.

[41] An alternative approach consist in updating a single map with different sensors. In this case, however, the sensor with the faster sampling rate dominates the mapping result [73], which is a disadvantage in the case of asynchronous sensors.

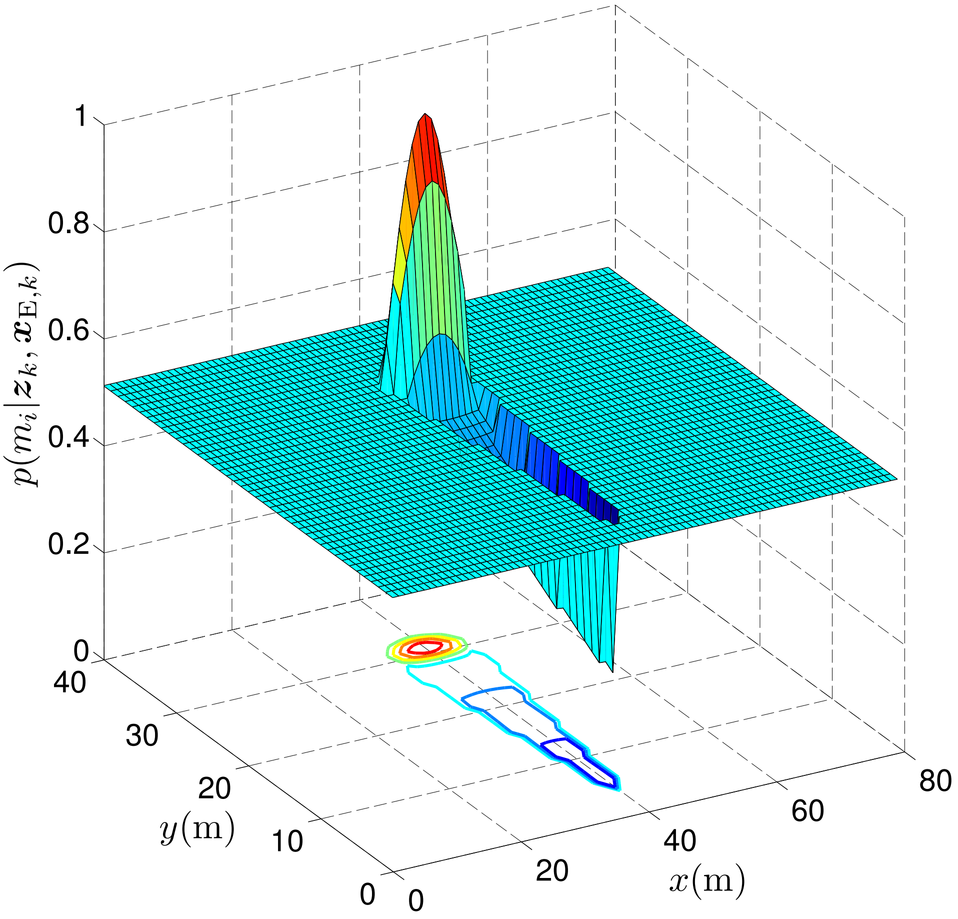


Figure 3.12: Exemplary inverse sensor model with corresponding probability contour plot for a 2D range sensor with Gaussian range and angle measurement uncertainties. The sensor is positioned at $\boldsymbol{x}_{\mathrm{E},k} = (40\,\mathrm{m}\ 0\,\mathrm{m}\ 90\,^\circ)^T$ and points in y-direction. A fictive obstacle detection is received 30 m ahead, thus $\boldsymbol{z}_k = (30\,\mathrm{m}\ 0\,^\circ)^T$. The free space area in front is modeled as a linear rising function with values smaller than 0.5, cells in the vicinity of the detection receive values greater than 0.5 under consideration of the uncertainties, whereas the area behind the obstacle detection is treated as unknown, which results in values for the inverse sensor model equal to 0.5.

Finally, it should be mentioned that occupancy grids can not just be mapped via Bayesian inference but also via Fuzzy or Dempster-Shafer theory, for a comparison see [207]. In contrast to Bayesian approaches, the latter can distinguish between areas of conflicting information and non-observed areas [73], and be generalized to more than two cell states [92]. On the downside, the storage requirements double as separate maps have to be maintained for occupied and free evidences. Nevertheless, Dempster-Shafer-based mapping is becoming increasingly popular in the intelligent vehicles domain [73, 91, 137]. Independent of the underlying mapping theory, the assumption that cells are *static* is taken in the majority of such traditional mapping algorithms. This causes problems if dynamic objects are present. The next chapter addresses this topic in detail and provides a novel method for grid mapping (and tracking) in *dynamic* road environments.

4 Grid Mapping in Dynamic Environments

The present chapter is devoted to a novel method for grid mapping in dynamic road environments. Starting with the weaknesses of traditional mapping filters, related work on this topic is presented. Subsequently, the proposed dynamic mapping approach is introduced and implementation as well as performance aspects are highlighted. Finally, evaluation results from real dynamic traffic scenarios are shown to illustrate the capabilities of the developed system. The chapter closes with a summary of the method's key aspects.

4.1 Introduction and Motivation

Mapping is one of the fundamental problems in mobile robotics. Especially occupancy grid mapping is a widespread method for building dense representations of the environment in a probabilistic way and found entrance into the intelligent vehicles domain as already explained in the overview in Section 2.2.

One main challenge in state-of-the-art mapping is to deal with dynamic environments [245]. In modern ADAS and intelligent vehicles, grid maps are most often constructed by mapping algorithms designed only for a static world even in dynamic environments, e.g. via BBFs as explained in Section 3.1.8. Hence, the so-constructed maps suffer from corruptions caused by dynamic objects in form of trails of slowly declining occupancy probabilities. This undesirable effect is illustrated in Fig. 4.1 on an exemplary driving scene with a dynamic vehicle in front. The corresponding BBF-based grid map is shown on the right. The reason for the occurring corruptions is that to unlearn that a cell is free takes as many observations as to learn its occupancy in BBF-based grids [49].

The aim of this chapter is to present a method that overcomes these limitations and minimizes mapping errors *online* in a *computationally ef-*

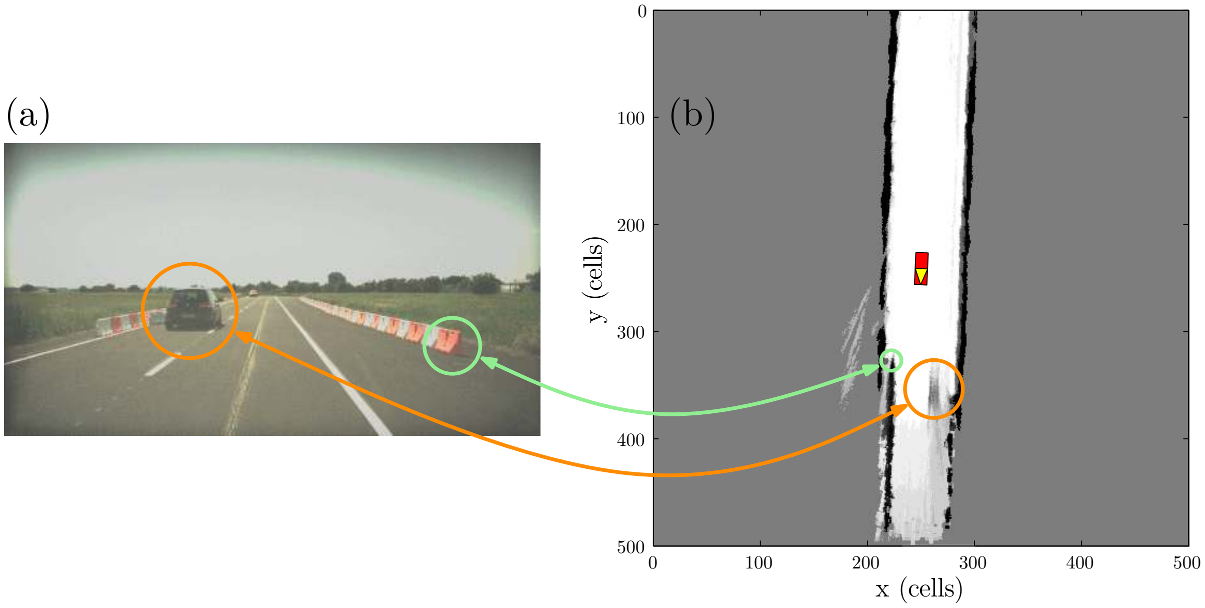


Figure 4.1: Effects of mapping dynamic environments with algorithms designed for a static world. The exemplary driving scenario (a) contains a dynamic object in front of the ego vehicle and construction walls at the side. Each cell of the corresponding occupancy grid map (b) is filtered with an individual BBF – the red ego vehicle is shown in the middle of the map. The darker a grid cell, the higher its occupancy probability. Undesirable mapping errors stemming from the vehicle in front are clearly visible in the grid.

ficient way to generate a *local*[1], optimized grid map that only contains *static* environment structures while simultaneously detecting and *tracking* the corruption-causing *dynamic* objects. This facilitates higher-level environment recognition tasks, which do not have to deal with moving object corruptions any more, extend available static free space for trajectory planners, and provide an object-based, compact description of dynamic objects. The method should ideally

- cope with different sensors employed in the automotive domain including radars, lidars, and stereo cameras,
- be computationally efficient to keep up with the sensor update time,
- be easily combinable with existing systems relying on a traditional BBF-based mapping,

[1] Problems only involved in global mapping, e.g. loop closing, are not treated in this thesis.

- work even if dynamic objects pass static structures in close vicinity, which causes merged static and dynamic environment grid cells, and

- cope with ego motion and noise.

Before the proposed approach to this problem is presented, an overview of related work on occupancy grid mapping in dynamic environments is first given in the following.

4.2 Related Work

The following discussion about related work on dynamic mapping focuses on works addressing the problem of *online* map building in *highly dynamic* environments. Therefore, neither offline approaches such as [31], which use the expectation maximization algorithm to differentiate between dynamic and static parts of the environment, are considered further nor methods trying to learn a limited number of configurations of dynamic objects as shown in [237] because objects in road environments move continuously and cannot be restricted to certain configurations.[2]

Works that address the problem of *online* map building in *highly dynamic* environments can be subdivided into two categories:

- Methods that incorporate the scene dynamics directly into the mapping process [54, 57, 62, 244, 277].

- Methods that filter out measurements supposed to belong to dynamic objects and subsequently use a traditional mapping algorithm suitable for a static world [42, 100, 141, 183, 184, 193, 229, 259–261, 263].

A representative of the first category is the so-called *Bayesian Occupancy Filter* (BOF) [57], which generates a four-dimensional grid representation of the environment – two dimensions for the Cartesian coordinates and two for the velocity components of each cell. Each cell's motion is predicted with a constant velocity model. A computationally less demanding modification [54] produces a two-dimensional occupancy grid representation with a distribution of possible velocities for each cell. Another approach is described in [62, 63] (and similarly in [244]), in which a particle filter-based tracking is used to infer the velocity of the grid cells. The raw occupancy grid serves as the measurement vector and the particles

[2] A survey covering these approaches can be found in [173].

additionally provide the building blocks of the modeled world. Two grids are used in [277], one for the occupancy probability of moving objects, and one of static objects. These grids are mapped with different inverse sensor models. However, no physically interpretable prediction models are applied. Hence, if a moving object cannot be detected, the area is inevitably mapped onto the static grid because detection failures cannot be bridged.

Representatives of the second category are the works of [261,263], which bring up the general problem of *Simultaneous Localization, Mapping and Moving Object Tracking* (SLAMMOT), and provide a computationally feasible solution by decomposing the problem into two separate estimators that solve *Simultaneous Localization and Mapping* (SLAM) and *Detection And Tracking of Moving Objects* (DATMO) concurrently.[3] This results in a separate moving object grid for each object as well as a grid for the static world. Either a consistency-based or a moving-object map-based detector is used to classify measurements as belonging to moving objects directly. These are filtered from the static map to get a better pose estimation within SLAM. In [100], moving people are tracked by a sample-based JPDAF and measurement beams resulting from moving objects are likewise excluded to improve the map quality. In [260], new measurements are compared with the local map to classify them as dynamic. However, ambiguities occur if objects appear in unexplored regions because it is not possible to distinguish between moving and newly appearing static environments. Therefore, a tracking step is used in [259] as an improvement after the initial dynamic cell hypothesis generation to separate dynamic and static objects more robustly. Laser scanner measurements are subsequently connected with the help of box models of predefined sizes for different classes such as pedestrians or vehicles. A motion evidence score is calculated in [193] to help filtering out false moving object hypothesis. This score is based on newly occupied and free regions, but, in opposition to the proposed approach, the information is not used within the measurement vector generation for an employed RBPF-based tracking. A combination of object-based multi-target tracking and grid mapping can further be found in [42, 229], where grid cells are associated with object states and predicted within the map. A subsequent feedback from the map is used to preserve consistency between the map content and the object list. With this laser scanner-based system, trails of moving objects

[3] The topics of mapping and tracking have long been studied in isolation – probably because the tracking community's primary topic of radar-based aircraft tracking does not require mapping due to the large open space without static obstacles.

can be suppressed and the tracking itself benefits from the grid mapping. In [141], the disparity image is classified into regions of stationary and moving parts by generating object hypotheses by segmenting object contours on a v-disparity image, which are tracked over time with additional exploitation of optical flow within the measurement equations. A classifier based on a sequential probability ratio test on the residuals of a stationary and a moving process model is used to distinguish moving from stationary parts of the environment. Subsequently, only non-moving parts of the disparity image are integrated into the grid. In [183], fuzzy rules are used for the distinction between static and dynamic cells. These rules take criteria such as cluster size, correlation, and grid resolution into account. Once a cell is classified as dynamic, it is, however, not updated in the conventional way but by taking the current sensor measurement as the new occupancy probability. Due to unconvincing results, the same authors propose a different approach in [184]. In the improved algorithm, the grid itself undergoes a hierarchical segmentation based on a distance criterion. Subsequently, each segment is associated with already tracked objects if possible. Segments associated with moving objects are used to update an IMM tracking filter, whereas only the ones associated with stationary objects update an occupancy grid. The problem of direct grid segmentation, as done in [184], is that adjacent static and moving segments can merge. This might limit its usefulness for unstructured inner city environments with many static segments in the vicinity of moving objects.

Conclusion

In conclusion, methods of the first category are computationally demanding as high-dimensional state spaces have to be estimated. Moreover, they cannot easily be combined with existing systems relying on a conventional grid representation, contrary to the works of the second category. The main limitation of the methods of this second category is that they can only be applied if measurements can robustly be separated into measurements of stationary and moving objects.[4] This is a challenging task even for costly laser scanner sensors such as utilized in [193, 259, 263] and all the more for sensors like stereo cameras and radars (which are used in this

[4] Some vision-based works, e.g. [79], circumvent this separation problem by using trained image feature-based vehicle/people detectors and exclude all emerging tracks from a simultaneously built static occupancy grid. This is, however, neither possible with other environment sensors nor for non-trained dynamic objects and would also exclude standing vehicles/people.

work) as already stated in [184]. Moreover, works of the second category are often tailored towards a specific sensor modality, e.g. using optical flow information only available in vision-based systems, and are therefore not sensor-independent. Also, the challenge of correct static environment mapping in cases in which dynamic objects move in close vicinity to static structures, is not treated explicitly in the mentioned approaches.

The proposed method belongs to the second category and therefore keeps the traditional, BBF-based grid mapping scheme, but differs from the mentioned works in a couple of ways:

- Not raw sensor measurements are classified as static or dynamic but cells in temporally consecutive maps after a traditional grid map update step. Spurious measurements are thus filtered out initially by the inverse sensor model and the algorithm can directly be coupled with existing systems.

- Newly available free space is used explicitly for the generation of dynamic object hypothesis even in the vicinity of static environment structures. With this novel approach, it is not only possible to receive a unique object heading angle measurement in each time step but also to reconstruct even only partially visible objects without the need of clustering the original grid or measurement.

- A nonlinear, adaptive IMM-UK-PDAF as introduced in Section 3.2.3 is used not merely to recursively estimate the states of underlying motion models of extended objects but rather for checking the consistency of cell movement with motion characteristics of real dynamic objects. This helps in the decision between real dynamic objects, newly mapped static environment, and sensor noise. It is further possible to filter out dynamic cells even if the motion detector fails, namely by using the prediction models.

4.3 Proposed Approach

In the following, the proposed approach to online mapping of highly dynamic driving environments is presented. Excerpts have already been published by the author in [225, 226].

4.3.1 System Overview

An overview of the proposed dynamic mapping approach is given in Fig. 4.2. It consists of three main parts: The generation of dynamic cell hypothesis (blue), the subsequent tracking (orange), and the classification and grid post processing (yellow). Inputs are the current, unoptimized grid map $\boldsymbol{m}_k$, the map of the last time step $\boldsymbol{m}_{k-1}$, and the ego motion vector $\boldsymbol{u}_{k-1}$. Outputs are an optimized grid map $\boldsymbol{m}_{\text{opt},k}$, in the best

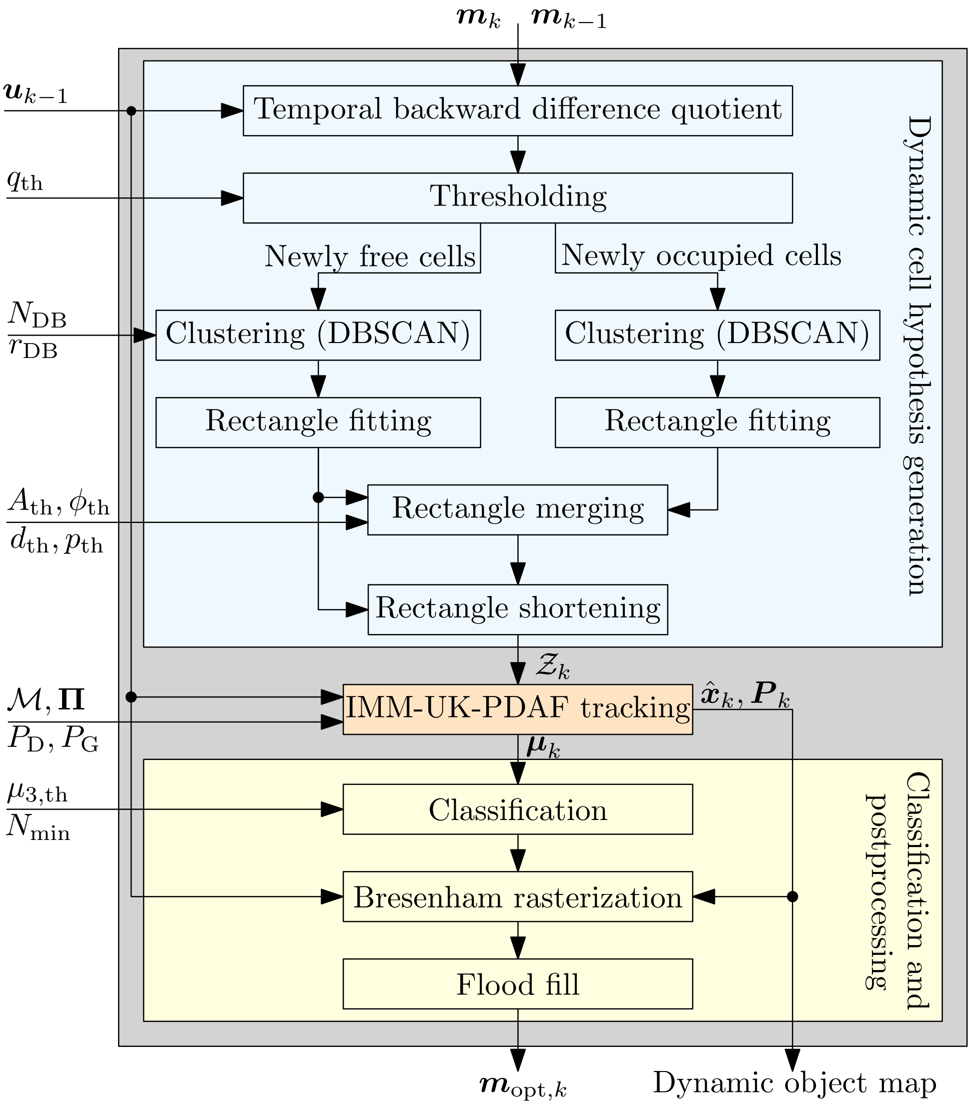


Figure 4.2: Overview of one time step of the proposed dynamic mapping approach. Inputs are maps with dynamic object corruptions, $\boldsymbol{m}_k$ and $\boldsymbol{m}_{k-1}$, as well as ego motion $\boldsymbol{u}_{k-1}$. Outputs are an optimized grid map $\boldsymbol{m}_{\text{opt},k}$ and a dynamic object map with estimated motion states. Parameters are explained in the main text.

case free of dynamic object corruptions, and a dynamic object map. The input grid maps themselves are supposed to be the result of individual BBF updates as explained in Section 3.3.1. The set of environment sensor measurements is generated by a stereo camera and an automotive long-range radar sensor with individual appropriate inverse sensor models in this work, see [91].[5] The set of ego vehicle poses is supposed to be known precisely due to the ego vehicle's proprioceptive sensors. Each of the three parts is described in detail in the following subsections. Relevant steps are further illustrated in Fig. 4.3 on the grid map of the exemplary driving scene, which serves as a running example in this chapter.

4.3.2 Generation of Dynamic Cell Hypothesis

The first step within the grid-based dynamic cell hypothesis generation is the calculation of the ego-motion compensated temporal backward difference quotient of the occupied probability $p(m_i)$ for each grid cell m_i to highlight changing regions. This is similar to frame differencing [278], which is a common approach to motion detection in the image analysis domain.[6] The result, visualized in Fig. 4.3b, illustrates the complexity of robustly extracting true dynamic object cells from such a difference grid as changing regions could not only stem from real dynamic objects but also from newly mapped static environments and noise. Moreover, main parts of moving objects are suppressed and invisible in this representation. This is because changes in occupancy probability are small in areas where dynamic objects are really located but large in areas that dynamic objects just entered or just passed. Consequently, dynamic objects have to be reconstructed.

For this purpose, two new binary grids are generated by segmenting the temporal difference grid with two thresholds $\pm q_{\text{th}}$, so that one contains only cells of increasing free probability (newly free cells) and the other only cells of increasing occupancy probability (newly occupied cells). These new grids are subsequently clustered separately by a *Density Based Spatial Clustering for Applications with Noise* (DBSCAN) with neighborhood radius r_{DB} and minimal point number N_{DB} [80]. The advantageous properties of DBSCAN in contrast to other methods like the well-known standard k-Means are the following:

[5] Stereo camera: Continental MFC 300; Radar sensor: Continental ARS 300.

[6] Motion estimation algorithms such as optical flow [159,271], which also take spatial differences and uncertainty into account, might lead to better motion detection but are computationally too expensive.

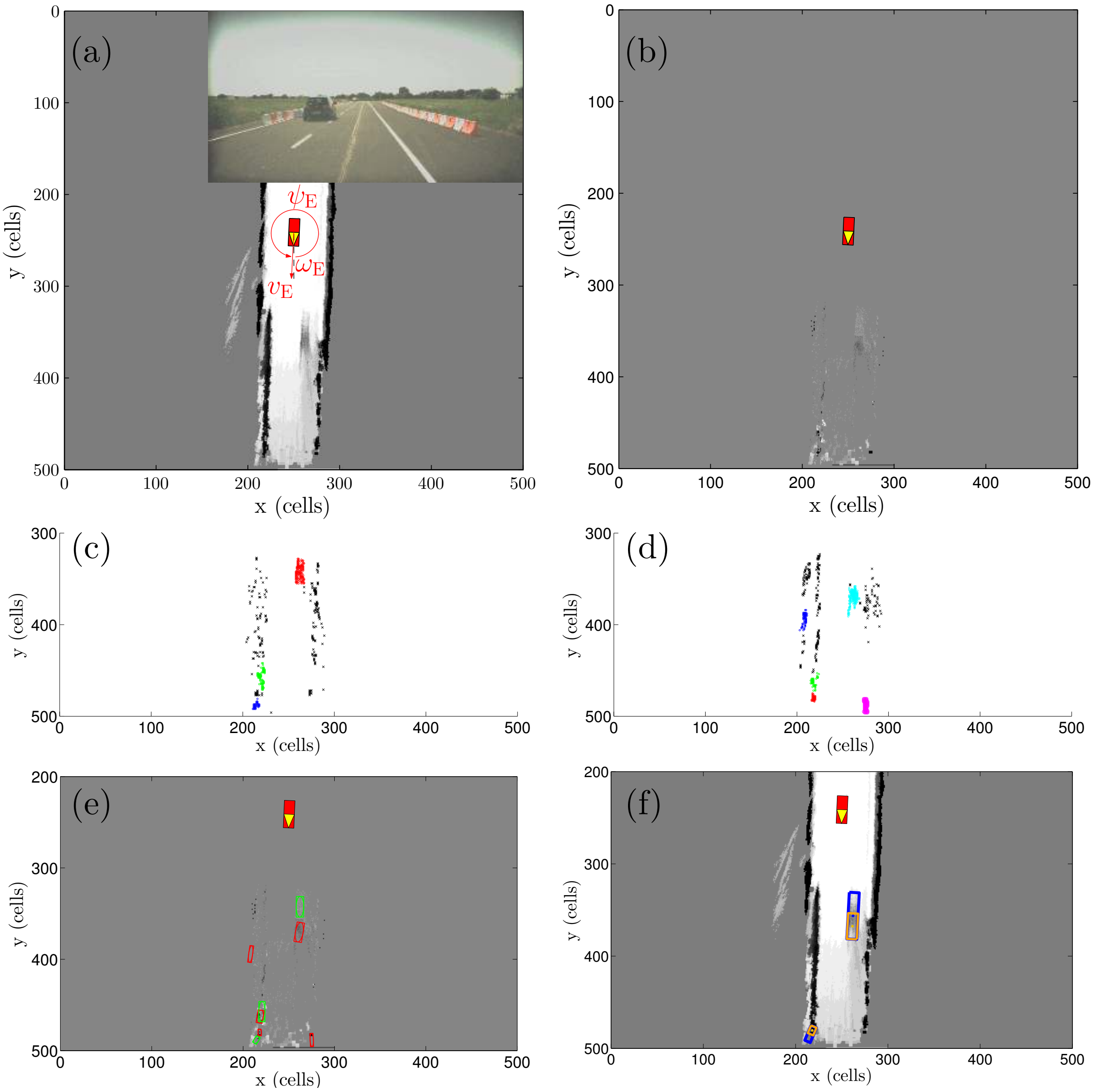


Figure 4.3: Generation of dynamic cell hypothesis in an exemplary driving scenario. The input grid map $\boldsymbol{m}_k$ (a) is of fixed size (80 m × 80 m; 500 cells × 500 cells) and shifted with the ego vehicle's motion (velocity v_{E}, yaw rate ω_{E}), whose orientation in the map is given by the yaw angle ψ_{E}. The cells of the temporal backward difference quotient grid (b) are subsequently clustered. The results of this clustering step in form of newly free and newly occupied areas are shown in (c) and (d), respectively. Rectangles are then fitted to each cluster (newly free: green; newly occupied: red), see (e), and the combined, merged rectangles (blue) as well as reconstructed dynamic object hypothesis (orange) are finally depicted in (f). These hypothesis form the measurements for the subsequent tracking step.

- The number of clusters must not be known or specified beforehand. This fact is important as the number of real dynamic objects can never be known in advance.
- Clusters can have arbitrary shapes and do not need to be similarly sized. This fact is important as the amount of newly available free or occupied space changes with the objects' dimensions and velocities, which are not known beforehand.
- Outliers are readily identified during the clustering process and consequently not all data points need to be assigned to clusters. This fact is important as especially newly mapped static environment structures produce false dynamic cell hypothesis.

The resulting clusters are shown for newly free areas in Fig. 4.3c and for newly occupied areas in Fig. 4.3d – outliers are depicted with black crosses. We realize that most of the newly mapped static environment areas and noise are already correctly identified as outliers because of their low density. Moreover, clusters resulting from the real dynamic object, i.e. the vehicle in front of the ego vehicle, are separated from the ones that stem from newly mapped construction walls. Any distance-based clustering, such as an almost connected components clustering, would lead to completely false results as a merging between static and dynamic environment clusters occurred.

After the density-based clustering is performed, oriented rectangles are calculated for each individual found cluster. Results are shown in Fig. 4.3e, in which rectangles corresponding to newly free clusters are visualized in green and ones corresponding to newly occupied clusters in red. The rectangle calculation is made more robust by first extracting the error ellipse of each cluster by scaling its covariance matrix and generating the bounding rectangle for each ellipse.

Thereafter, rectangles of newly free clusters are merged with rectangles of newly occupied clusters, if they belong to the same object, in order to bridge invisible object areas. This rectangle shape model assumption[7] is reasonable for vehicles, bicycles, or pedestrians, and enables a completion of only partially visible objects, e.g. if only two edges are visible. The merging criteria, deduced from considerations that hold in fictitious, perfectly mapped bird's-eye view grids with non-turning objects, are the following:

[7] Note that approaches exist that provide a more detailed shape information of dynamic objects [10].

- Newly occupied and newly free rectangles belonging to the same object must have the same area as dynamic objects are supposed to be non-deforming and thus need the same new occupied space that they free up elsewhere.

- Newly occupied and newly free rectangles must have the same orientation due to the rectangular object shape assumption.

- Cells that lie in between newly occupied and newly free clusters belonging to the same object must be occupied in the original grid.

- The distance between newly occupied and newly free areas must lie within certain bounds corresponding to reasonable dynamic object sizes.

In reality, these criteria can never be fulfilled exactly and are therefore relaxed. For a successful merging, areas $A_\mathrm{f}, A_\mathrm{o}$ and angles $\phi_\mathrm{f}, \phi_\mathrm{o}$ of newly free and newly occupied rectangles just have to be similar, only a certain percentage p_o of occupied cells has to exist in the original grid on the connection line of the centers of mass of a newly free and a newly occupied rectangle, and the distance d_fo between newly free and newly occupied areas must not be arbitrary large. The criteria are further illustrated in Fig. 4.4 on an ideal example. With the thresholds $A_\mathrm{th}, \phi_\mathrm{th}, d_\mathrm{th}, p_\mathrm{th}$, the following set of equations has to be fulfilled for a successful merging:

$$\frac{1}{A_\mathrm{th}} \overset{!}{<} \frac{A_\mathrm{f}}{A_\mathrm{o}} \overset{!}{<} A_\mathrm{th}, \tag{4.1a}$$

$$\min\big((\phi_\mathrm{f} - \phi_\mathrm{o})\mathrm{mod}(\pi/2); \pi/2 - (\phi_\mathrm{f} - \phi_\mathrm{o})\mathrm{mod}(\pi/2)\big) \overset{!}{<} \phi_\mathrm{th}, \tag{4.1b}$$

$$d_\mathrm{fo} \overset{!}{<} d_\mathrm{th}, \tag{4.1c}$$

$$p_\mathrm{o} \overset{!}{>} p_\mathrm{th}. \tag{4.1d}$$

Since angles are ambiguous with respect to the quadrant at this stage and since a merging can occur via the length or width of the rectangle, the minimal angle difference of all these variants is compared to the threshold ϕ_th in (4.1b). In the parametrization of these thresholds, a balance has to be found between two conflicting demands. If the criteria are relaxed too strongly, many false dynamic object reconstructions are generated, whereas strict thresholds pose a risk that real dynamic objects are missed. In practice, it has found to be beneficial to loosen the criteria quite a bit and to treat emerging false detections (clutter) in the subsequent filtering.

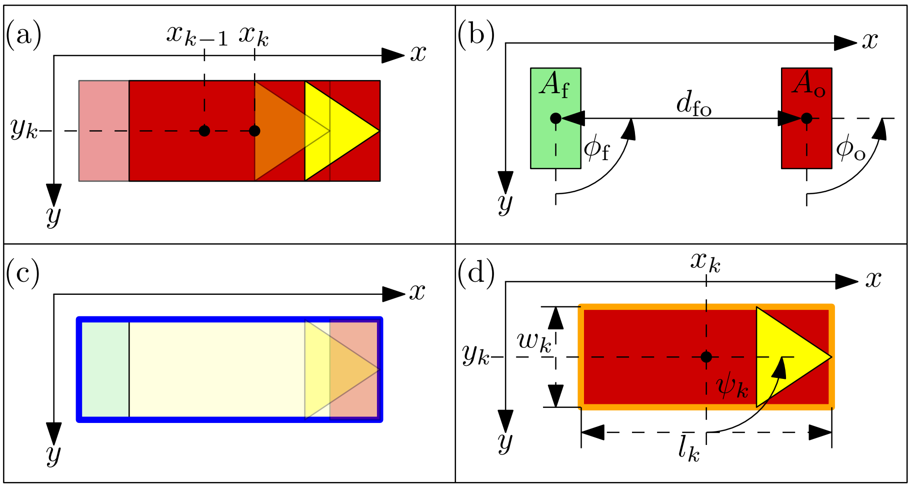


Figure 4.4: Dynamic object reconstruction example. The position of a dynamic object moving to the right (yellow arrow) is visualized in (a) for two consecutive time steps in a grid map-fixed coordinate system. The corresponding ideal temporal backward difference quotient grid is shown in (b). Newly free cells are depicted in green, newly occupied cells in red. The merging criteria are similar areas A_f, A_o, similar angles ϕ_f, ϕ_o, maximum distance d_fo, and minimal percentage of occupied cells on the connection line of the center of masses of the two rectangles. The result of the merging in form of the (blue-framed) combined rectangle is shown in (c), whereas the final (orange-framed) rectangle, which has been shortened by the newly free cells and corresponds to the reconstruction of the original object, is visualized in (d).

If several merging combinations between newly free and newly occupied rectangles are possible, the distance criteria is treated as most relevant to decide which merging is indeed performed. This rather simple approach has found to be sufficient.[8] The final parameter values are given in Table 4.2, p. 96.

The merging itself is performed by calculating the combined rectangle of newly free and newly occupied areas of an object. These are, however, too large as the areas of newly free cells do not correspond to the real dynamic object because the object already just left that area. Consequently, the merged rectangles are shortened by the newly free rectangles to generate a final dynamic object hypothesis. Fig. 4.3f shows the result in form of the merged rectangles that fulfilled the criteria in blue, and the final, reconstructed dynamic object hypothesis in orange, overlaid over the orig-

[8] The problem could also be treated as a generalized assignment problem for which a global optimum over all possible merging combinations with arbitrary weighted criteria is to be found. This is, however, computationally much more demanding.

inal grid map. Besides the correct reconstruction of the dynamic vehicle in front, also a false reconstruction took place at lower left as the newly mapped static environment coincidentally fulfilled the criteria as well. At this time step, it is impossible to decide whether a reconstruction stems from a real dynamic object or not.

To make this decision in a robust way, a novel IMM-UK-PDAF tracking-based approach is applied, which allows to reason about whether movements of temporally consecutive measurements are consistent with motion characteristics of real dynamic objects or not. Measurements for this tracking step are center coordinates x, y, heading ψ, length l, and width w of each final reconstructed dynamic object hypothesis. Note that the heading angle is, at this stage, not ambiguous with respect to the quadrant any more because we know where newly occupied cells are located in the reconstructed hypothesis. This is further illustrated in Fig. 4.5.

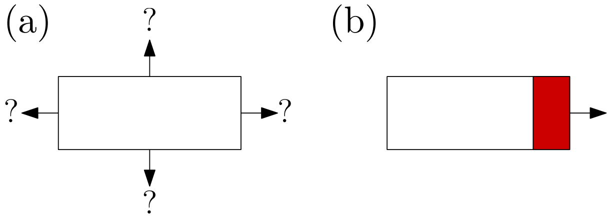


Figure 4.5: Explanation of the availability of a unique heading angle measurement. Rectangles from directly clustered measurements (without grid) contain angle ambiguities as the object could be moving in each direction (a). This ambiguity is resolved by measuring the angle in the direction of newly occupied cells shown in red (b).

All measurements are combined in the measurement vector $\boldsymbol{z} = (z_1 \; \dots \; z_5)^T = (x \; y \; \psi \; l \; w)^T$, and all N_z measurement vectors at time step k in turn constitute the measurement set $\mathcal{Z}_k = \left\{\boldsymbol{z}_{i,k}\right\}_{i=1}^{N_z}$, which is the input of the tracking step described in the following.

4.3.3 Tracking of Dynamic Cell Hypothesis

For tracking, we consider a nonlinear jump Markov system with a model set $\mathcal{M} = \left\{M_j\right\}_{j=1}^{r}$ of $r = 3$ models, which share a common state vector $\boldsymbol{x} = (x \; y \; \psi \; v \; \omega \; l \; w)^T$. The states consist of object center position x, y in the grid map-fixed coordinate system, heading angle[9] ψ, absolute veloc-

[9] Note that no distinction between heading (yaw) angle and course angle is made because the slip angle [204] is treated as zero during tracking.

ity magnitude in driving direction v, absolute turn rate ω as well as object length l and width w, see Fig. 4.6.

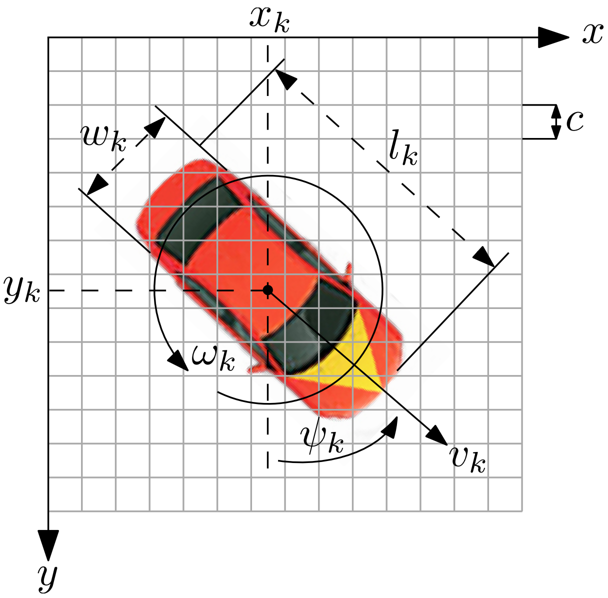


Figure 4.6: States of a tracked object at time step k with center position x_k, y_k in the grid map-fixed coordinate system, heading angle ψ_k, absolute velocity magnitude in driving direction v_k, absolute turn rate ω_k as well as object length l_k and width w_k. The quadratic grid cells have a fixed length c.

Motion and Measurement Models

Two of the motion models describe physically consistent motion behavior of real objects in form of a (nearly) *Constant Velocity* (CV) model M_1 given by the discrete-time, stochastic system dynamic equation

$$\boldsymbol{x}_{k+1} = \begin{pmatrix} x_k + \frac{v_k T \sin \psi_k}{c} \\ y_k + \frac{v_k T \cos \psi_k}{c} \\ \psi_k \\ v_k \\ 0 \\ l_k \\ w_k \end{pmatrix} + \boldsymbol{u}_k + \boldsymbol{w}_{1,k}, \tag{4.2}$$

and in form of a (nearly) *Constant Turn Rate and Velocity* (CTRV) model M_2 to cope with turning objects, described by

$$\boldsymbol{x}_{k+1} = \begin{pmatrix} x_k + \frac{v_k}{c\omega_k}(\cos\psi_k - \cos(\omega_k T + \psi_k)) \\ y_k + \frac{v_k}{c\omega_k}(\sin(\omega_k T + \psi_k) - \sin\psi_k) \\ \psi_k + \omega_k T \\ v_k \\ \omega_k \\ l_k \\ w_k \end{pmatrix} + \boldsymbol{u}_k + \boldsymbol{w}_{2,k}, \tag{4.3}$$

with the grid cell length c and the – not necessarily constant – sampling time T.[10] Both models share a common artificial input vector $\boldsymbol{u}_k$, which models the purely translational[11], deterministic, local grid map shift due to ego motion, and both include individual, zero-mean, white, Gaussian noise vectors $\boldsymbol{w}_{1,k}, \boldsymbol{w}_{2,k}$ with corresponding covariance matrices $\boldsymbol{Q}_{1,k}, \boldsymbol{Q}_{2,k}$, respectively. Such models are common in maneuvering multi-target tracking applications and are extensions of the ones found in [152], for example.[12]

Both consistent motion models are additionally supplemented with a third one, M_3, that corresponds to random motion and is in fact a stationary model described by

$$\boldsymbol{x}_{k+1} = (x_k \; y_k \; \psi_k \; 0 \; 0 \; l_k \; w_k)^T + \boldsymbol{u}_k + \boldsymbol{w}_{3,k}, \tag{4.4}$$

with much larger process noise covariance matrix $\boldsymbol{Q}_{3,k}$. The common artificial input vector including ego vehicle velocity magnitude v_{E}, heading ψ_{E},

[10] The alternative version of the CTRV model, in which the velocity vector is represented in Cartesian coordinates instead of the combination of velocity magnitude in driving direction (polar velocity) and heading angle has found to perform worse in a comparative study [93] and is therefore not employed. Note also that it is not necessarily better to use an even larger model set because this will not only increase computational load but also degrade achievable filtering results due to the excessive competition between unnecessary models [150]. If the yaw rate approaches zero, the CTRV model reduces to the CV model as can be seen via L'Hôpital's rule.

[11] The grid map is never rotated. Only the ego vehicle's orientation changes within the map, see Fig. 4.3a. The main advantage over using an ego vehicle-fixed grid is the avoidance of grid cell interpolation errors that would otherwise be induced in case of ego vehicle turns, see also [73, 91, 269].

[12] The models differ because of the different coordinate systems and the additional recursive estimation of length and width.

and yaw rate ω_{E}, is given by

$$\boldsymbol{u}_k = \begin{pmatrix} -\frac{v_{\mathrm{E},k}}{c\omega_{\mathrm{E},k}}\big(\cos\psi_{\mathrm{E},k} - \cos(\omega_{\mathrm{E},k}T + \psi_{\mathrm{E},k})\big) \\ -\frac{v_{\mathrm{E},k}}{c\omega_{\mathrm{E},k}}\big(\sin(\omega_{\mathrm{E},k}T + \psi_{\mathrm{E},k}) - \sin\psi_{\mathrm{E},k}\big) \\ \boldsymbol{0} \in \mathbb{R}^5 \end{pmatrix} \tag{4.5}$$

for stronger ego motion turns under CTRV assumptions, whereas a CV ego motion assumption is used in case of negligibly small yaw rates. The common measurement equation is

$$\boldsymbol{z}_k = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \boldsymbol{x}_k + \boldsymbol{v}_k, \tag{4.6}$$

with Gaussian measurement noise vector $\boldsymbol{v}_k$ and associated measurement noise covariance matrix $\boldsymbol{R}_k$. Therefore, velocity and yaw rate are hidden states as they are not measured directly.[13]

The main aim of the tracking step is to estimate whether temporally consecutive measurements correspond to real dynamic objects and thus behave according to a CV/CTRV model, or whether they are rather the result of random noise or static environment false positives and therefore better fit the random motion model. The reason why the chosen model set, along with a multiple model tracking filter, allows this classification is illustrated in Fig. 4.7 on a one-dimensional example.

IMM-UK-PDAF-Based Tracking

The tracking itself is performed with an IMM-UK-PDAF as introduced in Section 3.2.3 and sketched in Fig. 3.9, p. 57. States $\hat{\boldsymbol{x}}_k$, error covariances $\boldsymbol{P}_k$, and mode probabilities $\boldsymbol{\mu}_k$ are thus recursively estimated in a computationally efficient way. An important aspect in the measurement validation step is that instead of using the predicted measurement with the largest – in terms of determinant – innovation covariance $\boldsymbol{S}$ for setting up the track gate, only the model subset $\mathcal{M}_{\mathrm{s}} = \{M_1, M_2\}$ is considered, so that j_r in (3.90), p. 58, equals $j_r = \arg\max_{j\in\mathcal{M}_{\mathrm{s}}} |\boldsymbol{S}_{j,k}|$. Otherwise, the

[13] The radar, in principle, provides a direct measurement of the relative radial velocity, which could be embedded into the measurement equations. As the focus of this work is to realize a sensor-independent approach suitable also for lidar- and camera-based systems, this option has not been exploited.

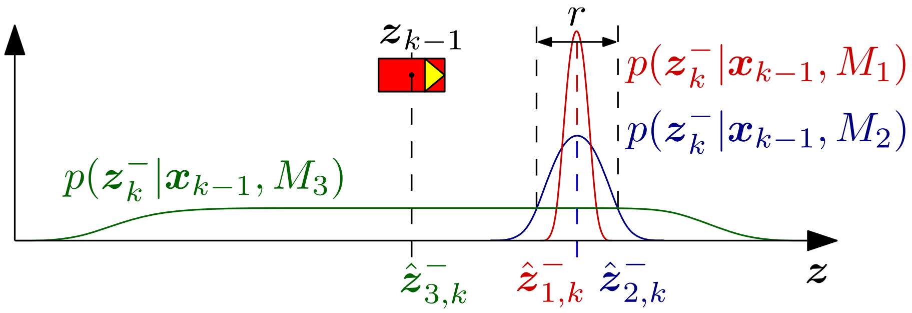

$\boldsymbol{z}_{k-1}$
r
$p(\boldsymbol{z}_k^-|\boldsymbol{x}_{k-1}, M_1)$
$p(\boldsymbol{z}_k^-|\boldsymbol{x}_{k-1}, M_2)$
$p(\boldsymbol{z}_k^-|\boldsymbol{x}_{k-1}, M_3)$
$\hat{\boldsymbol{z}}_{3,k}^-$
$\hat{\boldsymbol{z}}_{1,k}^-$
$\hat{\boldsymbol{z}}_{2,k}^-$
$\boldsymbol{z}$

Figure 4.7: Explanation of the motion behavior classification. Let the dynamic object move to the right. Its last position is given by $\boldsymbol{z}_{k-1}$ and the corresponding states $\boldsymbol{x}_{k-1}$ include a positive velocity. The measurement prediction densities for this scenario $p(\boldsymbol{z}_k^-|\boldsymbol{x}_{k-1}, M_j)$ are qualitatively visualized. All of the three models can theoretically describe any kind of motion. The difference is that models M_1 and M_2 provide sharper predictions centered at the predicted position further to the right, whereas M_3 spreads its predicted probability over a much wider measurement space and remains centered at the old position. The overlap with the consistent models M_1 and M_2 is small. If a new measurement at time step k now falls in the measurement space r, the consistent models explain the measurement much better. This results in a higher mode probability for M_1 and M_2. The fact that the filter favors the model that fits better to the measurement sequence (higher mode probability) allows a classification of consistent and inconsistent motion behavior with negligible degradation of tracking performance. The more specific the prediction models, the better do the discrimination capabilities become.

large innovation covariance of the random motion model would lead to frequent gate overlap due to the associated large gates. Moreover, the random motion model M_3 is filtered by a conventional KF instead of an UKF due to its linearity. To prevent the problem of the discontinuity at the zero and 2π angle crossing within the tracking, the angle state is unbounded and measured angles with $\psi \in [0, 2\pi[$ are artificially increased by integer multiples of 2π to match the state prediction. This modification also solves the problem in the IMM interaction step (3.85), p. 56, in the case that one model has already crossed the discontinuity and the others did not.

The reason why exactly the filter combination in form of the introduced IMM-UK-PDAF has been chosen for the task at hand, are the following:

- The IMM Filter is an approved multiple model filter, which provides an excellent compromise between approximation accuracy and computational expense [167] and can cope with multiple motion modes – a prerequisite for the introduced classification between consistent and inconsistent motion behavior. GPB1 Filters are less accurate, whereas GPB2 Filters perform only slightly better at a much higher computational cost [167].

- Due to nonlinear motion models, underlying UKFs are used, which are more accurate and robust than EKFs [53]. PFs are computationally too demanding and not necessary due to the relatively mild nonlinearities of the motion models, so that each mode can reasonably be represented as a Gaussian. The same holds for HFs, which are too slow for any reasonably fine discretization of the seven-dimensional state space.

- PDAFs provide greater robustness than hard assignment algorithms such as NNSFs in clutter. As the thresholds in (4.1), p. 77, have to be relaxed in order not to miss real dynamic object reconstructions, clutter measurements exist and a PDAF is consequently advantageous. Explicit multi-target solutions (JPDAF, GNNF, MHT), which perform data association jointly over all tracks, are, however, not necessary as track gates in our case seldom overlap. One reason for this convenient fact lies in the choice of the IMM Filter, which adaptively changes the size of the validation gate. The gate is only raised in case of a target maneuver and narrowed during rather benign motion because a less noisy process model is in effect during these times.[14] The second reason is the proposed dynamic object reconstruction approach that provides measurements in a high, five-dimensional measurement space instead of the common, two-dimensional position measurement space. This aspect is further illustrated in Fig. 4.8.

  Both aspects in combination effectively attenuate gate overlap so that also explicit countermeasures against track coalescence effects [37] are not necessary. Another advantage of the reconstruction method is that every object creates at most one detection and not several as if raw measurements had been used. Thus, the PDAF assumption that at most one validated measurement originates from the target under consideration, see Section 3.2.2, is fulfilled and an application therefore justified.

[14] A less noisy process model leads to smaller predicted state error covariances, which in turn imply smaller innovation covariances after the measurement prediction and thus also a smaller gate size.

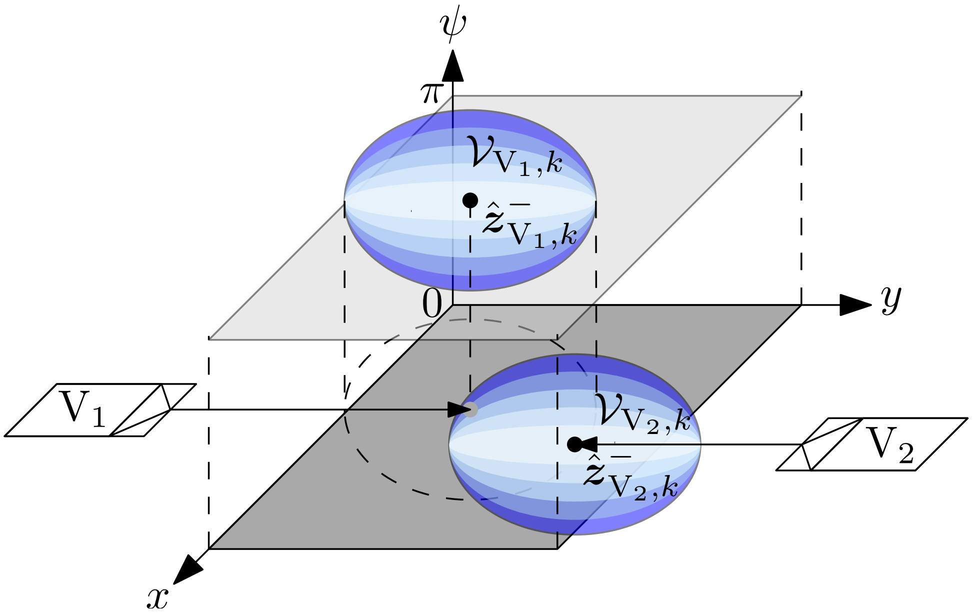

ψ
π
$\mathcal{V}_{\mathrm{V}_1,k}$
$\hat{\boldsymbol{z}}^-_{\mathrm{V}_1,k}$
0
y
V$_1$
$\mathcal{V}_{\mathrm{V}_2,k}$
$\hat{\boldsymbol{z}}^-_{\mathrm{V}_2,k}$
V$_2$
x

Figure 4.8: Validation gates $\mathcal{V}_{\mathrm{V}_1,k}, \mathcal{V}_{\mathrm{V}_2,k}$ of two oncoming vehicles V_1 and V_2 with predicted measurements $\hat{\boldsymbol{z}}^-_{\mathrm{V}_1,k}, \hat{\boldsymbol{z}}^-_{\mathrm{V}_2,k}$. The gates overlap in the two-dimensional x, y-measurement space but not in a measurement space that additionally includes the heading angle ψ. This mitigates data association problems.

Filter Parametrization

To make such an adaptive, nonlinear, time-variant, multiple model filter work as intended, a reasonable parametrization is crucial. The following parameters are to be specified:

1. Initial, common state estimates $\hat{\boldsymbol{x}}_0 = \hat{\boldsymbol{x}}_{1,0} = \hat{\boldsymbol{x}}_{2,0} = \hat{\boldsymbol{x}}_{3,0}$, initial, common estimation error covariance matrices $\boldsymbol{P}_0 = \boldsymbol{P}_{1,0} = \boldsymbol{P}_{2,0} = \boldsymbol{P}_{3,0}$, and initial mode probabilities $\boldsymbol{\mu}_0$.
2. Process noise covariance matrices $\boldsymbol{Q}_1$,$\boldsymbol{Q}_2$,$\boldsymbol{Q}_3$.
3. Measurement noise covariance matrix $\boldsymbol{R}$.
4. Model transition probability matrix $\boldsymbol{\Pi}$.
5. Target detection probability P_{D}.
6. Gate probability P_{G}.
7. UKF parameters $\alpha_{\mathrm{U}}, \beta_{\mathrm{U}}, \kappa_{\mathrm{U}}$.

The rationale behind the specific choices of the parameters, whose final values are summarized in Table 4.2, p. 96, are given in the following.

1. Proper filter initialization, i.e. the specification of $\hat{\boldsymbol{x}}_0$, $\boldsymbol{P}_0, \boldsymbol{\mu}_0$, is vital not only for obtaining consistent[15] state estimates but also because it affects the filter's initial transient behavior. Therefore, the initial filter parameters influence how fast reasonable hidden state estimates are obtained and also how fast erroneous initial state estimates can be corrected. Sufficiently high values of the error covariances are a prerequisite for consistent estimates and result in fast filter transients as new measurements are given more weight. On the downside, high initial state uncertainties imply noisier state estimates during filter start-up, often accompanied by large overshoots, and complicate track formations due to larger validation gates. Too low initial state uncertainties, in contrast, can lead to inconsistent estimates or even filter instability.

   In the system, a one-point initialization procedure [18] is employed and the filter is initialized with the first measurement vector that started a new track. Therefore, $\hat{x}_0, \hat{y}_0, \hat{\psi}_0, \hat{l}_0, \hat{w}_0$ can be specified directly. The corresponding initial estimation error uncertainties are set equivalent to the elements of the measurement covariance matrix $\boldsymbol{R}$. The hidden velocity and yaw rate states are initialized with zero.[16] Their standard deviations should be of the order of approximately half of the assumed maximum state values according to [18]. This ensures that the validation gates are large enough to include the next measurement even for targets with high velocity and yaw rate. All three underlying filters within the IMM Filter are initialized identically so that only a common $\hat{\boldsymbol{x}}_0$ and $\boldsymbol{P}_0$ is to be specified. The initial mode probabilities are equally distributed according to the principle of indifference, which corresponds to a non-informative prior. Therefore, $\boldsymbol{\mu}_0 = (\frac{1}{3}\ \frac{1}{3}\ \frac{1}{3})^T$ is chosen.

2. For process noise design, several dependencies have to be kept in mind. The larger the process noise, the noisier the state estimates because measurements are weighted more strongly than the uncertain, model-based predictions. Larger process noise, however, also brings the benefit of better track following capabilities in the case of

[15] A state estimator is consistent if the estimation errors are zero-mean and have covariances smaller or equal than the filter-calculated error covariances. [18]

[16] A velocity measurement is implicitly available because of the temporal difference in form of the length of newly available free/occupied space and could be used similar to a two-point differencing track initialization [18]. However, this implicit velocity measurement has found to be too noisy and an inclusion did not bring any benefit.

maneuvering targets, whose motion is different to the designed motion models. A smaller process noise, in contrast, results in smoother state estimates – a desirable effect – but also in a larger filter phase lag as predictions are weighted more strongly and the filter resists to follow the measurements. This could lead to track losses, inconsistent estimates, or even instabilities, which could have severe consequences. Therefore, the process noise should be set rather too high than too low. The modeling of the different process noises within the three IMM Filter models has to consider these conflicting demands. In principle, there exist two main variants of process noise modeling. First, direct noise modeling in discrete-time and second, modeling of the noise intensity matrices $\boldsymbol{Q}_{\mathrm{n}}$ of the underlying continuous-time models and subsequent discretization [18,152]. The assumption of the first variant is that the system is triggered by a discrete, white noise sequence, whose value is constant over a sampling period, e.g. by a piecewise constant acceleration. The second variant, in contrast, captures the integrated effect of an assumed, underlying, continuous-time, white noise process and results in a so-called discrete-time equivalent model. The latter is more convenient in the case of variable sampling times [18] and therefore employed.

The noise of the CTRV model M_2 is designed based on maximally physically possible changes in velocity and yaw rate according to noisy longitudinal and yaw target accelerations. Following [18], these changes are of the order $\sqrt{Q_{\mathrm{n}}T}$ during a sampling time T with Q_{n} being the corresponding scalar noise intensity. To maintain tracks even during strong acceleration and yawing maneuvers, the noise intensity matrix has therefore[17] to be of the order $\boldsymbol{Q}_{\mathrm{n},2} \approx T\mathrm{diag}(0,0,0,\dot{v}^2_{\max},\dot{\omega}^2_{\max},\dot{l}^2_{\max},\dot{w}^2_{\max})$ with assumed maximum target longitudinal acceleration $\dot{v}_{\max}$ and yaw acceleration $\dot{\omega}_{\max}$. Additional uncertainties in the dimension parameters based on maximum changes in length $\dot{l}_{\max}$ and width $\dot{w}_{\max}$ are introduced. The smaller, the less noisy the dimension estimates. They should, however, not be set to zero to provide some forgetting capabilities with respect to old measurements and to prevent that the corresponding state covariances converge to zero.

Similarly, the noise of the CV model M_1 is based on changes in veloc-

[17] For example, for the velocity process triggered by white noise, $\sqrt{Q_{\mathrm{n}}T} \approx \dot{v}_{\max}T$ $\Rightarrow Q_{\mathrm{n}} \approx \dot{v}^2_{\max}T$, with $[Q_{\mathrm{n}}] = [\mathrm{m}^2/\mathrm{s}^3]$, holds. As Q_{n} is time-invariant, it is the power spectral density of the white noise acceleration.

ity and yaw angle according to noisy vehicle accelerations and yaw rates and could be set analogously, i.e. based on maximum acceleration and yaw rate. This would, however, undermine the advantages of a multiple model filtering. As turning targets are already handled by the CTRV model, smoother state estimates are obtained during motion periods with nearly constant heading if the model is designed with smaller yaw rate uncertainty.

In contrast to the models M_1 and M_2 treated so far, M_3 models random, inconsistent motion that does not correspond to physically possible motion. For a good separation between consistent and inconsistent motion behavior, a large noise intensity corresponding to large position and yaw angle changes is induced. Additionally, the noise intensity for the dimension parameters l, w is raised within $\boldsymbol{Q}_{\mathrm{n},3}$. This facilitates the separation further as any strong change in object dimension over time is likewise an indicator of a false dynamic object reconstruction and – with this noise design – entails a rise of the mode probability of the random motion model M_3.

Finally, the continuous-time noise intensity matrices are discretized via the simplified discretization $\boldsymbol{Q}_{i,k} = \boldsymbol{Q}_{\mathrm{n},i}T$, $\forall i \in \{1, 2, 3\}$ according to (3.17), p. 24, which has found to be sufficient.

3. Also for measurement noise design, several dependencies must be considered. Similar to a too low process noise, also a too low assumed measurement noise results in inconsistent state estimates and thus in estimated error covariances smaller than the true covariances. Estimated states are therefore falsely attested a too high certainty. As gating is performed based on the statistical distance between the prediction and the measurement, the consequence of such an unjustifiably certain measurement prediction is that measurements might get discarded although they would have been accepted as valid measurements if the uncertainty had been estimated correctly. Track losses are the consequence. In contrast, larger than correct assumed measurement uncertainties lead to an overestimation of the estimation error and are therefore suboptimal but less severe.

   In the system, the final measurement noise covariance matrix $\boldsymbol{R}$ stems from real-world measurements in a large variety of traffic scenarios. Uncertainties of length and width measurements were estimated directly under the assumption that these states are constant. Position and heading uncertainties, in contrast, were estimated by treating an additionally available, accurate object-based tracking as a quasi-ground-truth.[18]

4. The model transition probability matrix $\mathbf{\Pi}$ affects the mode transition dynamics. Large off-diagonal values favor frequent, agile mode changes, whereas large diagonal values lead to less dynamic, but also less noisy, motion mode estimates. It is designed as a constant, symmetric, diagonally dominant matrix and provides a compromise between these two opposing demands.

5. The target detection probability P_{D} is estimated directly by labeling how often a real dynamic object has successfully been reconstructed, i.e. detected, by the grid-based dynamic object detection approach described in Section 4.3.2 and how often the detection failed.

6. The gate probability P_{G} directly affects the size of the validation gate. The gate has to be sufficiently large to include the true object measurement with a high certainty. Too high values for the gate probability, however, raise the filter processing time considerably as many measurements fall in the validation gate, which are all used within the PDA update step. Moreover, if gates of several dynamic objects overlap, then also the PDA assumption that only a single target exists within the validation gate is violated, which leads to biased estimates. The choice of the gate probability should also be guided by the employed motion models. If only a single motion model for benign motion with minor process noise is used, then the gate probability needs to be higher than if a noisier model is in effect, which induces a larger track gate already.

   With these considerations in mind, the final value of $P_{\mathrm{G}} = 0.9$ has found to be a favorable compromise.

7. The UKF parameters $\alpha_{\mathrm{U}}, \beta_{\mathrm{U}}, \kappa_{\mathrm{U}}$ are chosen with respect to the considerations already formulated in Section 3.1.5.

The filtering results of the exemplary scene are shown in Fig. 4.9. It becomes obvious that even longer detection fails can be bridged by the tracker; velocity estimates settle with minor overshoot; the yaw rate correctly remains near-zero; noise in position, heading angle, and dimension measurements are attenuated; and high values are rapidly assigned to the mode probability corresponding to the CV model because the preceding

[18] A highly accurate reference system has not been available.

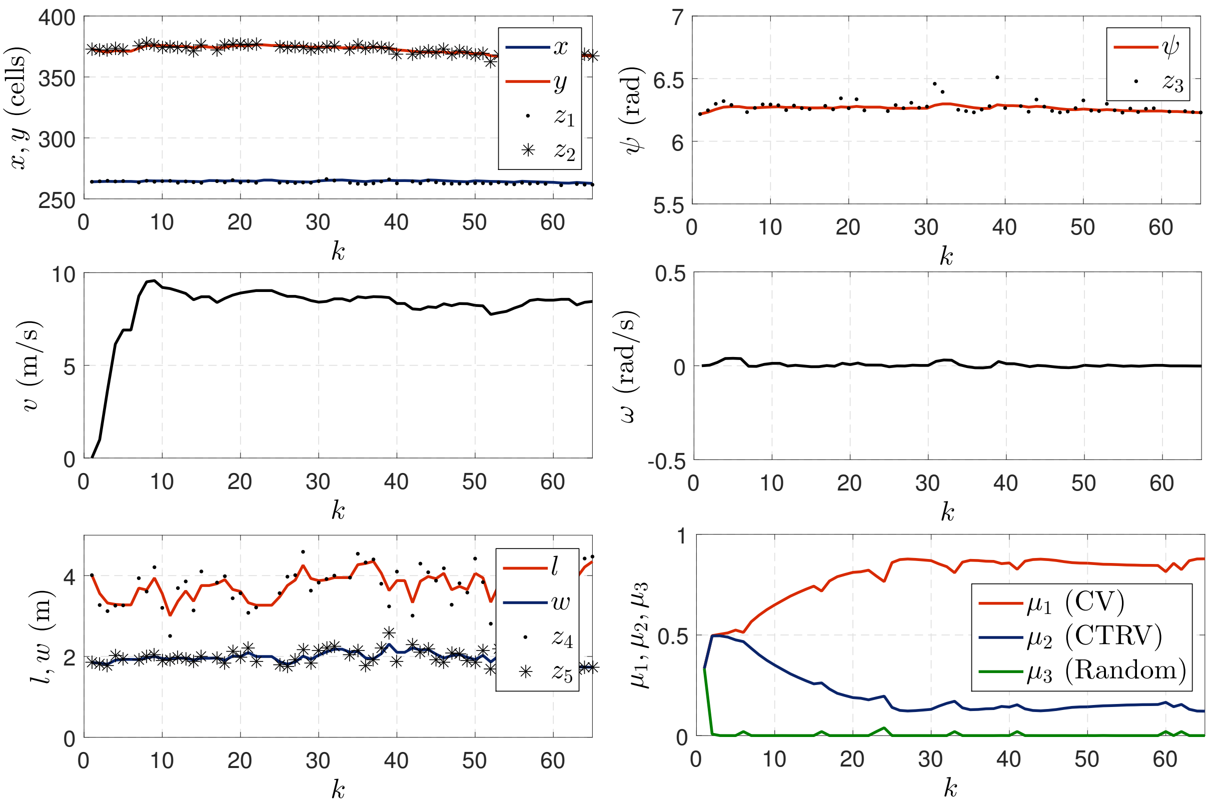


Figure 4.9: Estimated states and mode probabilities of the real dynamic object from the exemplary scene. One time step corresponds to approximately 65 ms, parameters as in Table 4.2, p. 96. Measurements stemming from the dynamic object reconstruction approach are additionally shown dotted.

vehicle moves through the construction site with nearly constant velocity. Fig. 4.10 additionally shows the emerging tracks and contrasts the estimated mode probabilities of the real dynamic object with those of the false reconstruction at lower left. The random motion mode probability of the consistently moving, real dynamic object rapidly converges towards zero (Fig. 4.10a), whereas the probability correctly remains high in the case of the false dynamic object reconstruction at lower left (Fig. 4.10b). This circumstance allows the classification of motion behavior, which is presented together with the grid post processing in the next section.

4.3.4 Classification and Grid Post Processing

For the decision whether a cell within a hypothesized dynamic, rectangular object can be classified as dynamic, the random motion mode probability μ_3 is compared to a threshold $\mu_{3,\text{th}}$ as soon as a track has received a minimal number of detections N_{min}. If the random motion mode probabil-

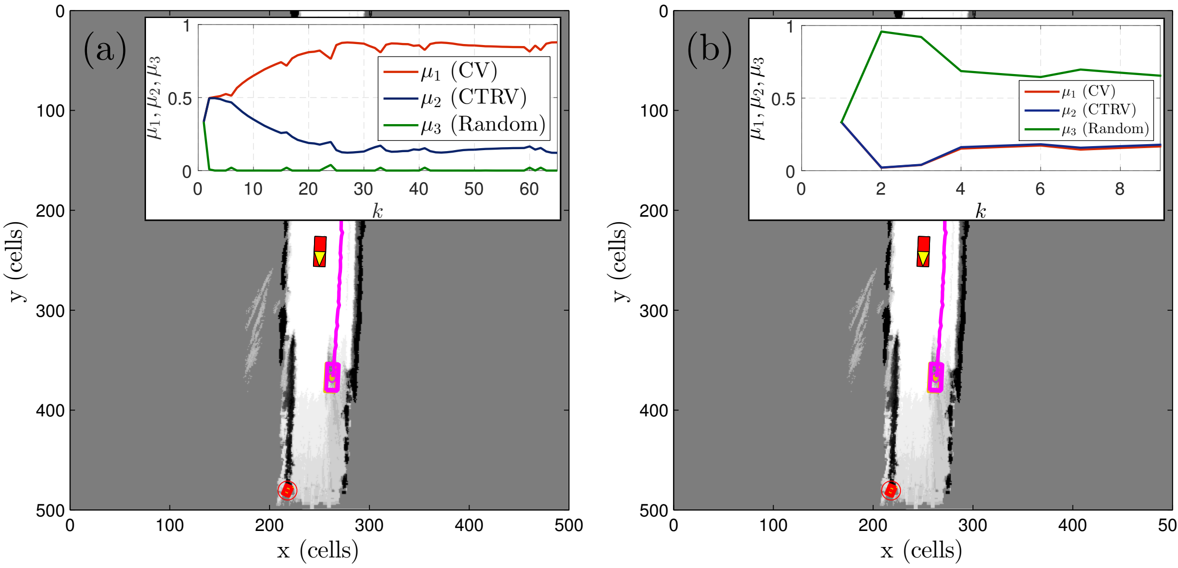


Figure 4.10: Unclassified dynamic object tracks with estimated mode probabilities for the real dynamic object shown in magenta (a) and for the false reconstruction shown in red (b). The random motion mode probability converges towards zero in the first case, whereas it remains high in the case of false reconstructions.

ity is sufficiently low, the object is moving in consistence with physically possible motion models and is therefore classified as a real dynamic object. In this case, it is integrated into the dynamic object map, see Fig. 4.11a.[19] In addition, all corresponding cells that belong to these tracks are artificially set to "free" in the current grid map since the map is supposed to contain only the static world for further processing. For this purpose, bounding rectangles are calculated for each track with their refined length and width estimations, and rasterized on the grid by means of the Bresenham algorithm [43] under consideration of past ego motion. A subsequent flood fill procedure is performed for clearing dynamic object mapping errors, and the final, optimized grid map $\boldsymbol{m}_{\text{opt},k}$ is generated (Fig. 4.11b). If the online classification of a cell as dynamic is not supported any more by a sufficiently low random motion mode probability, its actual occupancy probability is reinserted into the grid and the object is deleted from the dynamic object map. Wrong decisions can therefore be undone, which is

[19] The approach could therefore also be interpreted as a way of providing false track discrimination capabilities. Comparisons with alternatives such as track score-based [34], target existence-based [180], or "true target probability"-based [14] methods with an additional "unobservable target" model would be interesting but are out of scope of this work.

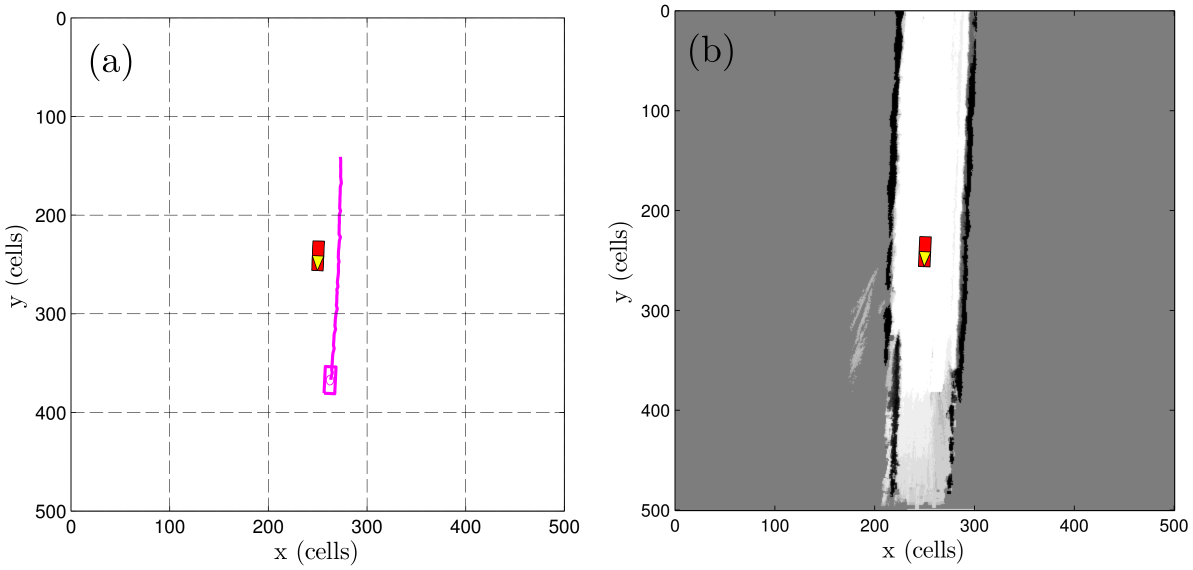


Figure 4.11: Result of the dynamic mapping approach for the exemplary scene. The dynamic object map with corresponding track is shown in (a), the final, optimized grid map $\boldsymbol{m}_{\mathrm{opt},k}$, which contains only the static environment, in (b). Dynamic object corruptions have been eliminated.

considered an additional advantage. The consistency between optimized occupancy grid map and dynamic object map is likewise always ensured with this approach.

In conclusion, the section presented a novel, computationally efficient method for grid mapping in dynamic road environments. It avoids dynamic object corruptions and simultaneously extracts a dynamic object map in a consistent way. The method has been implemented on an experimental vehicle and evaluated in a variety of real traffic scenarios, which is described in the following.

4.4 Implementation and Performance

The proposed dynamic mapping approach has been implemented both in MATLAB and in C++, and runs on an experimental vehicle shown in Fig. 4.12, which is equipped with a stereo camera and a radar sensor.[20] The optimized C++ version runs within (10.1 ± 4.7) ms, measured on a 3.3 GHz Intel Core i5-2500K quad-core CPU, which is well below the sensor update time of approximately 65 ms. Calculation times for the different

[20] Stereo camera: Continental MFC 300; Radar sensor: Continental ARS 300.

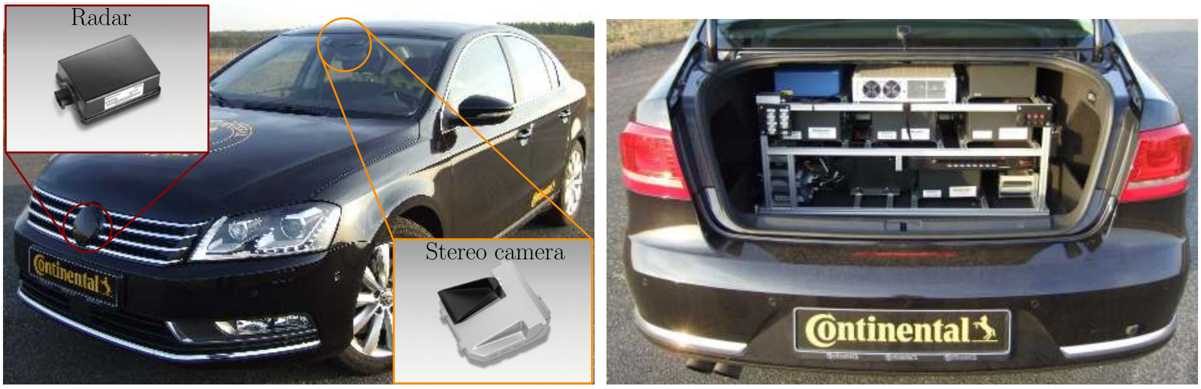


Figure 4.12: Experimental vehicle.

steps of the algorithm are further summarized in Table 4.1.[21] For achieving these low times, the algorithm has been implemented in a parallel fashion with the help of the OpenMP library [59] for multi-core processors. As can be seen in Fig. 4.2, p. 73, both the DBSCAN clustering and the rectangle fitting can be performed independently on newly occupied and newly free cells. This comes in handy because these two steps are computationally most demanding. However, even with parallel programming, a standard implementation of DBSCAN requires several dozens of milliseconds per grid. Only by exploiting the fact that data points for the clustering are all discrete, integer-valued coordinate points due to the underlying grid, it

Table 4.1: Calculation times of the dynamic mapping approach (mean, standard deviation, minimum, maximum) in ms. Temporal Difference (TD), Thresholding (TH), Clustering (CL), Rectangle Calculations (RC), Tracking (TR), Post Processing (PP).

Quantity	TD	TH	CL	RC	TR	PP	Total
Mean	2.7	0.5	3.3	3.1	0.2	0.3	10.1
Std	0.7	0.1	2.6	2.2	0.2	0.3	4.7
Min	2.5	0.5	0.7	0.0	0.0	0.0	3.7
Max	7.0	1.4	24.3	12.8	0.9	2.2	36.8

[21] The performance analysis is based on the same 17 real traffic scenarios (2766 grid maps) that are used for the system evaluation described in Section 4.5. Note that total min/max times are not necessarily the sum of min/max times of individual processing steps as these extreme individual values do normally not occur at the same time. Also the total standard deviation is not the root of the sum of squares of the individual standard deviations due to the different means.

becomes possible to restrict the required neighborhood searches to small, rectangular areas and only perform exact distance measurements between possible candidate points therein.[22] An additional visualization via the *Open Graphics Library* (OpenGL) has been implemented to illustrate the dynamic mapping results along with corresponding tracks on an on-board screen, see Fig. 4.13.[23]

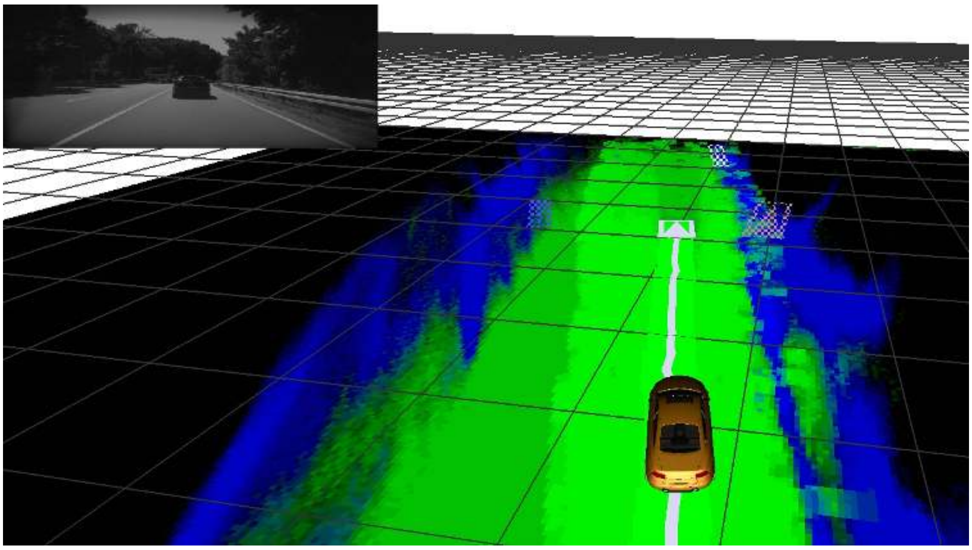


Figure 4.13: Online visualization of the dynamic mapping approach. Free cells are shown in green, occupied ones in blue. Dynamic object corruptions are eliminated and the corresponding true track is visualized in white. Tracks not classified as consistently moving, such as the dynamic object hypothesis on the right guardrail, are additionally shown hatched and are correctly not removed from the grid.

4.5 Evaluation

For the evaluation, 17 real traffic scenarios (2766 grid maps, 500000 cells per grid) were recorded, including 10 parallel traffic and 7 cross traffic scenarios on freeways and in inner cities. Figure 4.14 shows six exemplary scenes to give an impression of the scenarios and achievable results. Optimized grids with tracks are contrasted with unoptimized, BBF-based grid maps in the visualizations. All scenarios contain a large amount of

[22] Thanks to Zijad Maksuti, Christoph Münker, and Achim Stein for their help in implementing the optimized DBSCAN clustering.

[23] Thanks to Ralph Grewe for providing the basic grid map visualization code.

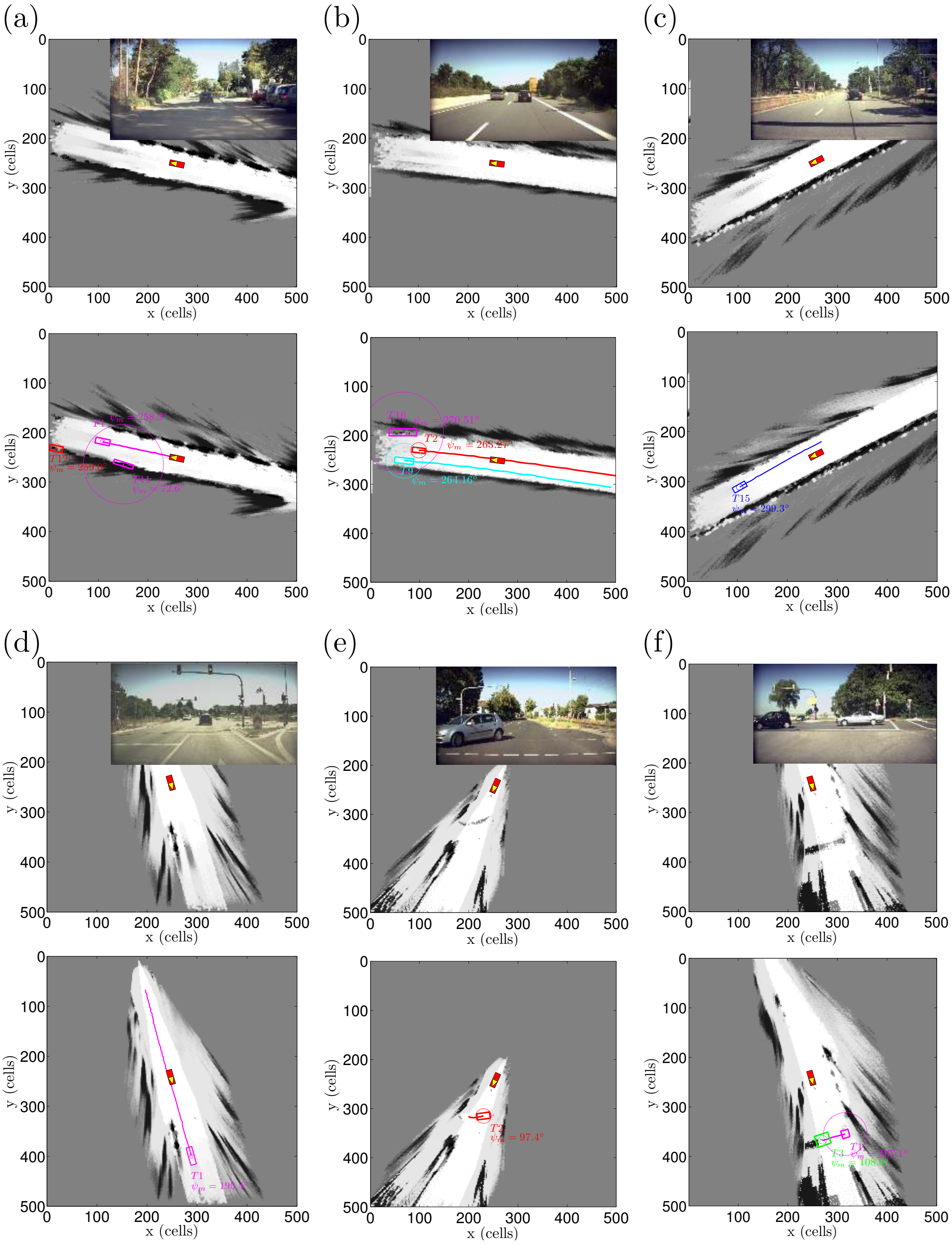


Figure 4.14: Excerpt of 6 out of 17 evaluation scenarios. Vehicle following scenario (a), high speed scenario (b), vehicle performing a lane change to the left (c), low speed scenario (d), cross traffic scenarios (e,f). Top: Original grid $\boldsymbol{m}_k$. Bottom: Optimized grid $\boldsymbol{m}_{\text{opt},k}$ with tracks. Only tracks classified as consistently moving are cleared from $\boldsymbol{m}_k$ and integrated into the dynamic object map.

static environment to be mapped simultaneously such as parking cars, guardrails, and trees, which are partly in direct vicinity of dynamic vehicles. This makes the separation between static and dynamic objects particularly challenging. The scenarios further involve a wide range of dynamic object velocities from walking speeds of vehicles just moving off such as in Fig. 4.14d through to velocities common in inner-cities as in Fig. 4.14a right up to velocities $> 120\,{}^{\mathrm{km}}/_{\mathrm{h}}$ as in the highway scenario in Fig. 4.14b. The parameters used in the evaluation are summarized in Table 4.2. The evaluation itself is performed on track level as well as on cell level and the results are shown in the following.

Table 4.2: Evaluation parameters of the dynamic mapping approach.

Parameter	Value
Grid size	80 m x 80 m
$c; T; q_{\mathrm{th}}$	$0.16\,\mathrm{m}$; $\approx 0.065\,\mathrm{s}$; $0.9\,{}^{1}/_{\mathrm{s}}$
$N_{\mathrm{DB}}; r_{\mathrm{DB}}$	12; 3 cells
$A_{\mathrm{th}}; \phi_{\mathrm{th}}$	3; $40\,^{\circ}$
$d_{\mathrm{th}}; p_{\mathrm{th}}$	12 m; 20%
$\boldsymbol{Q}_1$	$\mathrm{diag}\big(0, 0, (5\,{}^{\circ}/_{\mathrm{s}})^2, (2\,{}^{\mathrm{m}}/_{\mathrm{s}^2})^2, 0, (1\,{}^{\mathrm{m}}/_{\mathrm{s}})^2, (0.5\,{}^{\mathrm{m}}/_{\mathrm{s}})^2\big)T^2$
$\boldsymbol{Q}_2$	$\mathrm{diag}\big(0, 0, 0, (2\,{}^{\mathrm{m}}/_{\mathrm{s}^2})^2, (50\,{}^{\circ}/_{\mathrm{s}^2})^2, (1\,{}^{\mathrm{m}}/_{\mathrm{s}})^2, (0.5\,{}^{\mathrm{m}}/_{\mathrm{s}})^2\big)T^2$
$\boldsymbol{Q}_3$	$\mathrm{diag}\big((\frac{40}{c}\,{}^{\mathrm{m}}/_{\mathrm{s}})^2, (\frac{40}{c}\,{}^{\mathrm{m}}/_{\mathrm{s}})^2, (40\,{}^{\circ}/_{\mathrm{s}})^2, 0, 0, (2\,{}^{\mathrm{m}}/_{\mathrm{s}})^2, (1\,{}^{\mathrm{m}}/_{\mathrm{s}})^2\big)T^2$
$\boldsymbol{R}$	$\mathrm{diag}\big((\frac{1}{c}\,\mathrm{m})^2, (\frac{1}{c}\,\mathrm{m})^2, (10\,^{\circ})^2, (0.34\,\mathrm{m})^2, (0.2\,\mathrm{m})^2\big)$
$\boldsymbol{\Pi}$	$p_{ij} = 0.96\ \forall i = j; \quad p_{ij} = 0.02\ \forall i \neq j$
$\boldsymbol{P}_0$	$\mathrm{diag}\big((\frac{1}{c}\,\mathrm{m})^2, (\frac{1}{c}\,\mathrm{m})^2, (10\,^{\circ})^2, (10\,{}^{\mathrm{m}}/_{\mathrm{s}})^2, (25\,{}^{\circ}/_{\mathrm{s}})^2,$ $(0.34\,\mathrm{m})^2, (0.2\,\mathrm{m})^2\big)$
$\boldsymbol{\mu}_0; \hat{\boldsymbol{x}}_0$	$(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})^T$; $(z_{1,0}, z_{2,0}, z_{3,0}, 0, 0, z_{4,0}, z_{5,0})^T$
$\alpha_{\mathrm{U}}; \beta_{\mathrm{U}}; \kappa_{\mathrm{U}}$	0.001; 2; 0
$P_{\mathrm{D}}; P_{\mathrm{G}}$	0.7; 0.9
$\mu_{3,\mathrm{th}}; N_{\mathrm{min}}$	0.05; 3

4.5.1 Track Level Evaluation

On track level, the system is in fact a pattern recognition system that tries to distinguish whether emerging tracks originate from real dynamic objects or not. Defining a *True Positive* (TP) classification as "*Real dynamic object correctly classified as real dynamic object*" and a *False Positive* (FP) as "*Non-dynamic object misclassified as real dynamic object*", it becomes obvious that FPs can have severe consequences for collision avoidance systems as static environment cells would falsely be cleared from the grid. Therefore, a track already counts as a FP if it is misclassified as a real dynamic object at least once.[24] In contrast, a *False Negative* (FN) is not as severe because in this case, a real dynamic object and its cell corruptions will remain present in the grid as obstacles.

518 tracks were initialized in total. The classification results over all scenarios are 33 TP, 7 FP, 475 TN, and 3 FN, resulting in

- a precision of $\frac{\text{TP}}{\text{TP+FP}} = 0.83$,
- a recall of $\frac{\text{TP}}{\text{TP+FN}} = 0.92$, and
- a balanced accuracy[25] [46] of $\frac{1}{2}(\frac{\text{TP}}{\text{TP+FN}} + \frac{\text{TN}}{\text{TN+FP}}) = 0.95$.

The mean decision time from the initial detection to the classification of a TP is 3.3 time steps (standard deviation: 0.57 time steps). This is equal to the time span after which the grid cleaning procedure begins. The mean time span for which the rare FP classifications persist until falsely cleared cells are reinserted into the grid is only 3.5 time steps (standard deviation: 0.18 time steps). Consequently, FPs only have a very short-term negative effect, which is a particularly favorable result.

4.5.2 Cell Level Evaluation

For the cell level classification, it is examined how many grid cells are correctly cleared from dynamic object corruptions. For this purpose, all 2766 original grids were manually labeled by marking areas that are completely free in reality but not in the grid due to dynamic object corruptions. For all cells in these areas ($> 2 \cdot 10^6$ in total) it was checked whether they were successfully cleared by the dynamic mapping algorithm or not. Fig. 4.15 summarizes the final result for all scenarios, Fig. 4.16 only for parallel traffic scenarios[26], and Fig. 4.17 only for cross traffic scenarios. The

[24] The same holds for TPs. A track counts as a TP if it is correctly classified as dynamic at least once. If a TP result is falsely withdrawn over time, it will not alter the classification statistics but have negative effects on the cell level evaluation.

[25] Due to the class imbalance, the standard accuracy value of 0.98 would falsely indicate too positive results.

[26] Parallel traffic scenarios include all scenarios in which other dynamic objects mainly move in the same direction as the ego vehicle. They consist of vehicle following scenarios as well as vehicles moving on adjacent lanes or performing lane changes in front of the ego vehicle.

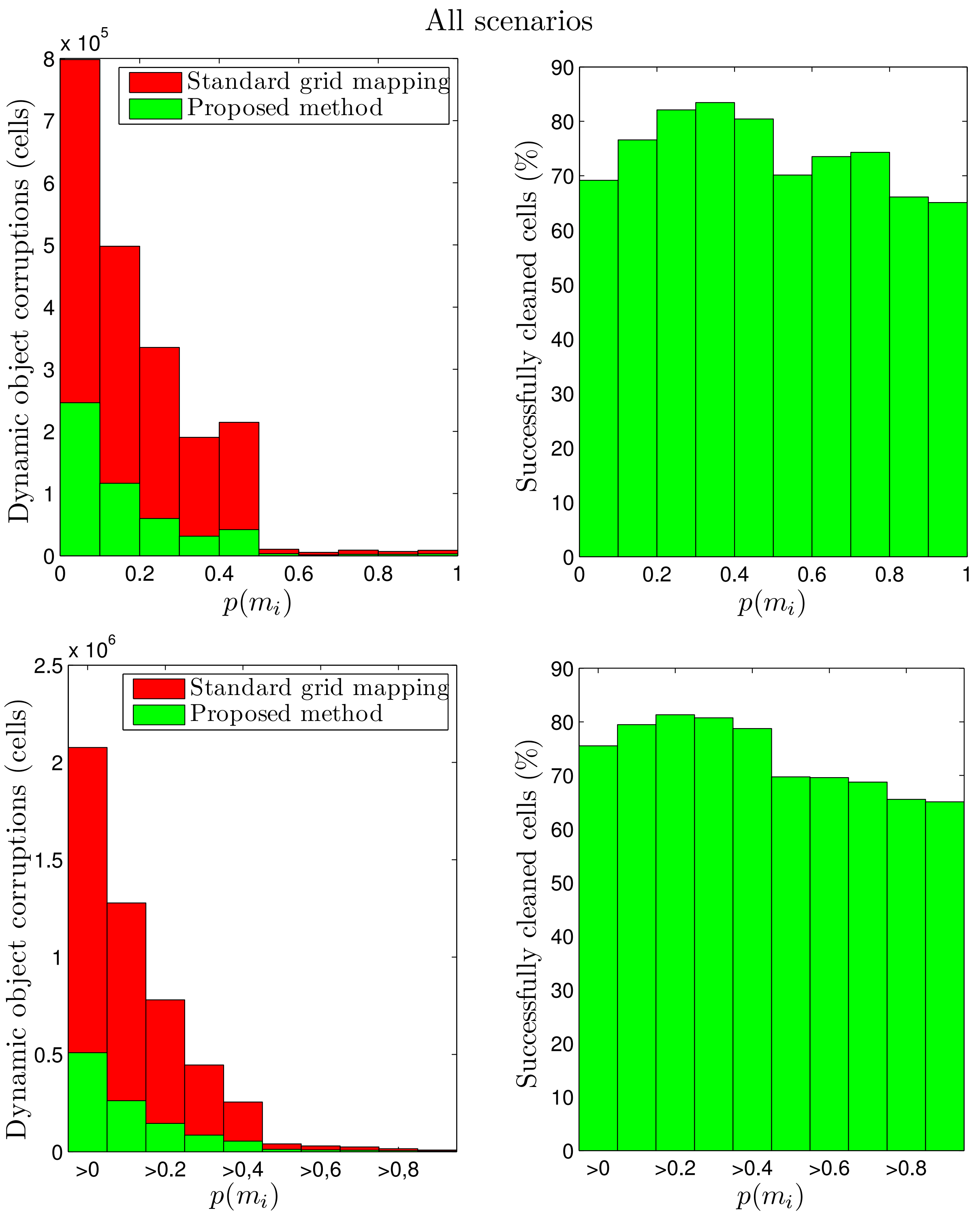


Figure 4.15: Evaluation results of the proposed approach to grid mapping in dynamic environments for all scenarios. Upper row: Number of dynamic object corruptions and percentage of successfully cleared cells in specific occupancy probability intervals. Bottom row: Analogue for open-ended occupancy probability intervals.

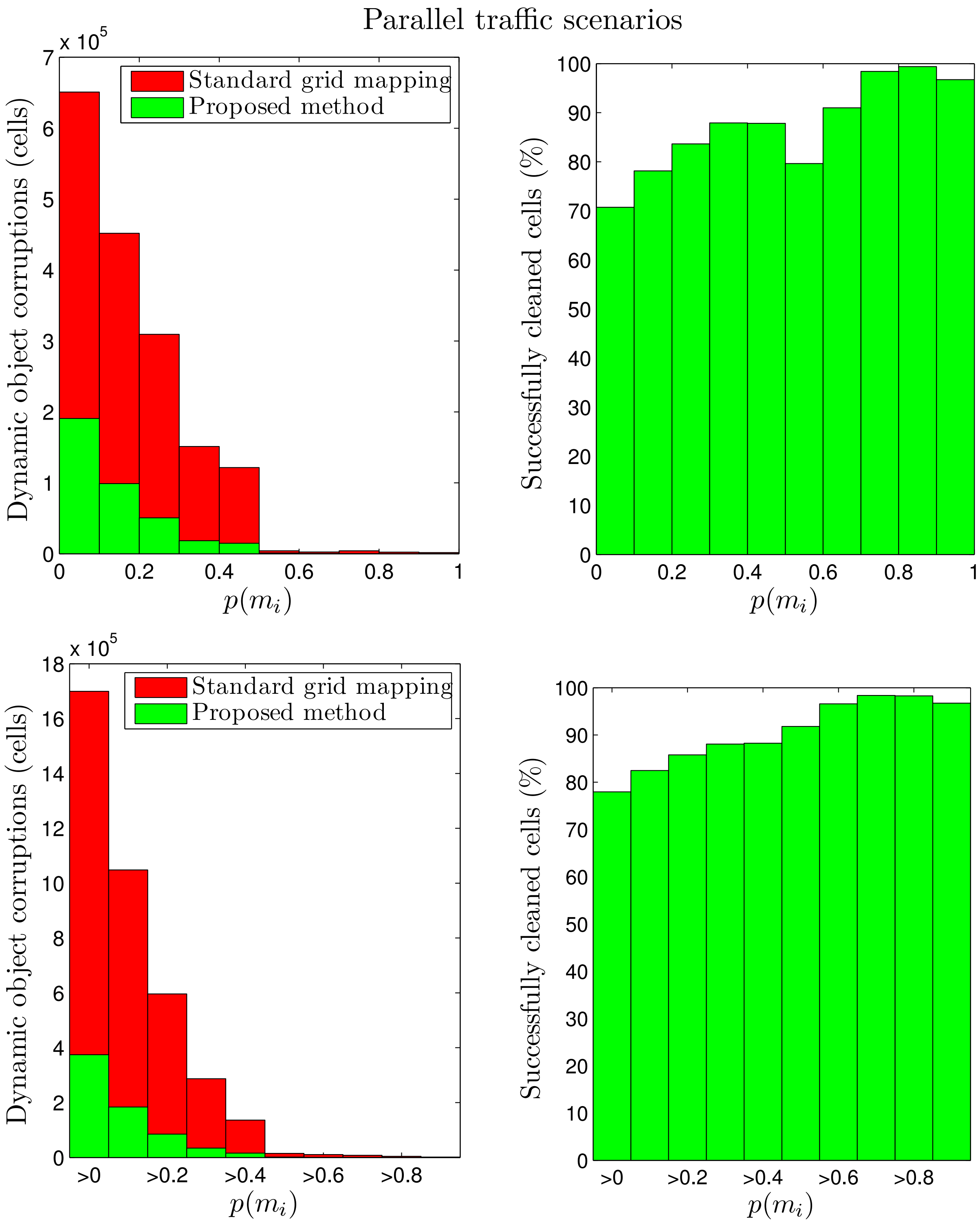


Figure 4.16: Evaluation results of the proposed approach to grid mapping in dynamic environments for parallel traffic scenarios. Upper row: Number of dynamic object corruptions and percentage of successfully cleared cells in specific occupancy probability intervals. Bottom row: Analogue for open-ended occupancy probability intervals.

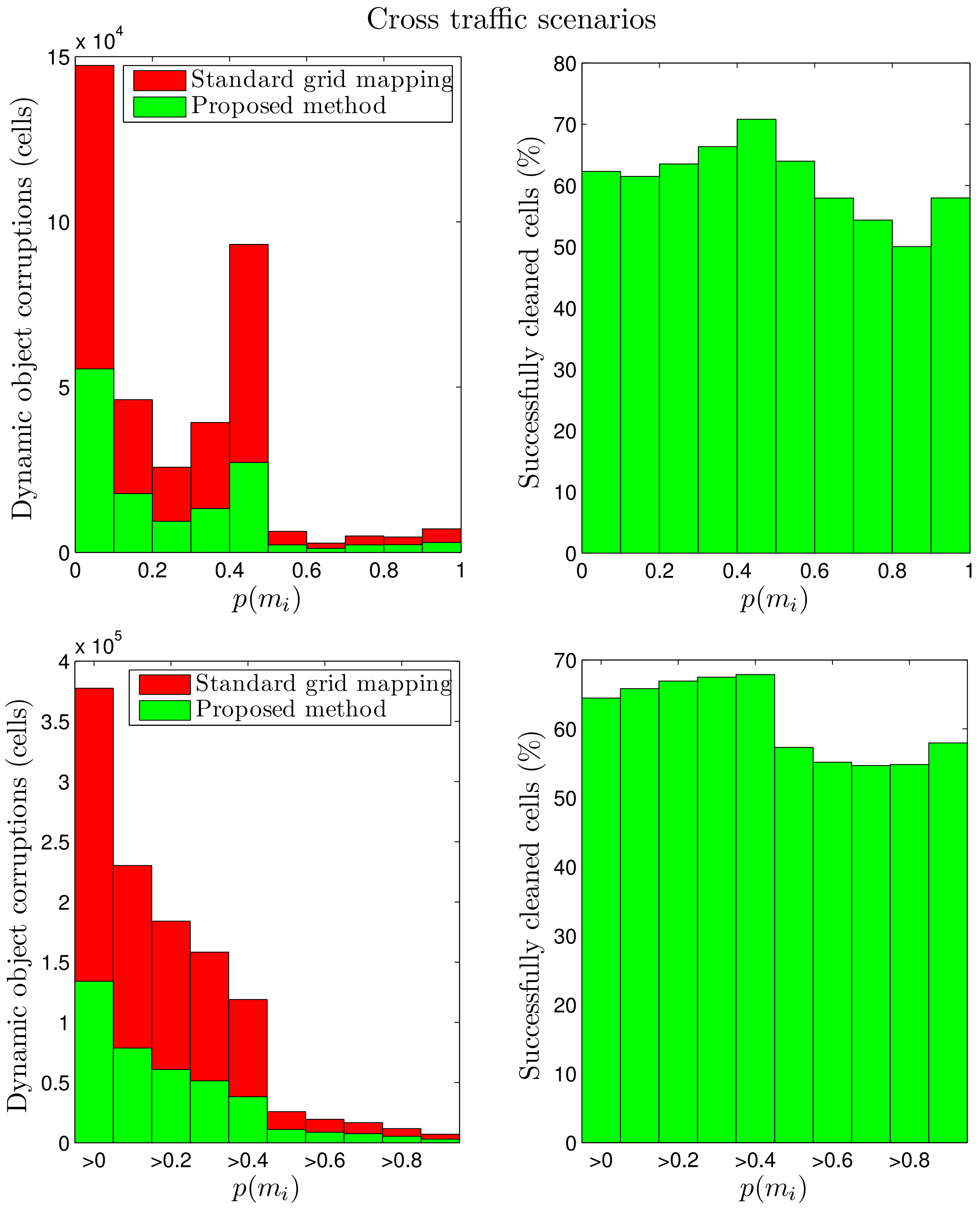


Figure 4.17: Evaluation results of the proposed approach to grid mapping in dynamic environments for cross traffic scenarios. Upper row: Number of dynamic object corruptions and percentage of successfully cleared cells in specific occupancy probability intervals. Bottom row: Analogue for open-ended occupancy probability intervals.

upper row in each figure shows the distribution of the number of remaining dynamic object corruptions in different occupancy probability intervals compared to the traditional, standard grid mapping algorithm via BBFs as well as the percentage of successfully cleaned cells within each interval. The proposed dynamic mapping algorithm shows considerably better results in all intervals. The advantages are specifically pronounced in parallel traffic scenarios. The bottom row additionally illustrates the analogue for different open-ended occupancy probability intervals. This representation is more informative for all subsequent algorithms that employ thresholding operations on the final, optimized grid map. The number and percentage of previously erroneously occupied cells (counted above a specific threshold), which are now correctly set to free by the dynamic mapping algorithm and therefore do not negatively impact subsequent algorithms any more, can directly be read off.[27] The proposed PFS map generation algorithm presented in the next chapter, for example, considers each cell with $p(m_i) \leq 0.3$ as free. Consequently, all falsely not cleaned dynamic cell corruptions with $p(m_i) \leq 0.3$ do not have a negative effect on this subsequent algorithm as they already count as free. The benefit of the proposed dynamic mapping algorithm can thus directly be read off by the evaluation results of the open-ended occupancy probability interval $p(m_i) > 0.3$. In parallel traffic scenarios, 88.1% of dynamic cell errors are filtered out. By additionally taking cross traffic scenarios into account, this fraction decreases to 80.8%. This is mainly due to the limited sensor opening angles (stereo camera: $50\,^\circ$, radar sensor: $17\,^\circ$) in combination with the fact that dynamic objects cannot be detected until they are completely within the field of view, thus inducing newly free cells behind. Considering all dynamic cell corruptions of all intervals, 75.5% of errors are successfully removed with the proposed algorithm.

4.6 Conclusion

A novel method for grid mapping in dynamic road environments with modest computational requirements has been presented. By a grid-based detection, tracking, and classification of dynamic cell hypothesis, most corruptions caused by other dynamic objects in the driving scene can be removed during the mapping process while simultaneously extracting a dy-

[27] Cells with occupancy probabilities below the threshold already count as free and it is not relevant any more whether they are treated by the dynamic mapping algorithm (and consequently set to free completely) or not.

namic object map. Static and dynamic environment entities are therefore well-separated in a consistent way, which facilitates higher-level situation interpretation tasks. The system has been implemented on an experimental vehicle and evaluated in real dynamic driving environments. The evaluation results showed superior performance compared to traditional occupancy grid mapping filters, which are in use in most state-of-the-art ADAS research projects. The main contributions of the proposed approach are

- the use of newly available free space to detect and reconstruct dynamic object hypothesis even in close vicinity to static environment structures and with the additional benefit of providing an unambiguous heading angle measurement,
- the robust classification by a novel IMM-UK-PDAF-based motion consistency check, which additionally provides state estimates of extended objects and allows to filter out dynamic mapping errors even during temporary motion detector fails via the prediction models, and
- the straightforward combination with existing mapping algorithms and different environment sensors as the method does not directly build upon raw sensor data but upon conventional grid maps, which are already in use in many current systems.

The result in form of an optimized, static grid map $\boldsymbol{m}_{\text{opt},k}$ provides the basis for the generation of compact PFS maps, which is the topic of the next chapter.

5 Parametric Free Space Maps

The present chapter is devoted to the generation process of PFS maps. Since this process can also be interpreted as a novel method of free space detection and estimation, related work on this topic is discussed first. The PFS mapping itself consists of a free space detection and a tracking part, which are both presented subsequently. Implementation and performance aspects are highlighted before experimental results within real driving environments are shown. The chapter concludes with the key aspects of the proposed method and a summary of the properties of the complete metric environment representation.

5.1 Introduction and Motivation

The PFS map as the proposed representation for arbitrary static environments for near-future ADAS has already been introduced in Section 2.3. It is a continuous, 2D bird's-eye view representation of the local static environment around the ego vehicle, which models relevant free space by a combination of a parametric curve and geometric primitives as shown in Fig. 5.1. PFS mapping is therefore also a method of free space detection. Knowledge about drivable free space is equally vital for autonomous vehicles and for next generation ADAS or *Collision Avoidance Systems* (CAS) that have to plan evasive maneuvers online because it makes a difference whether an area has just not yet been observed or whether real free space evidence is available.

Before explaining how PFS maps are robustly extracted from the optimized occupancy grid described in the previous chapter, an overview of related work on the topic free space detection and estimation is first given in the following.

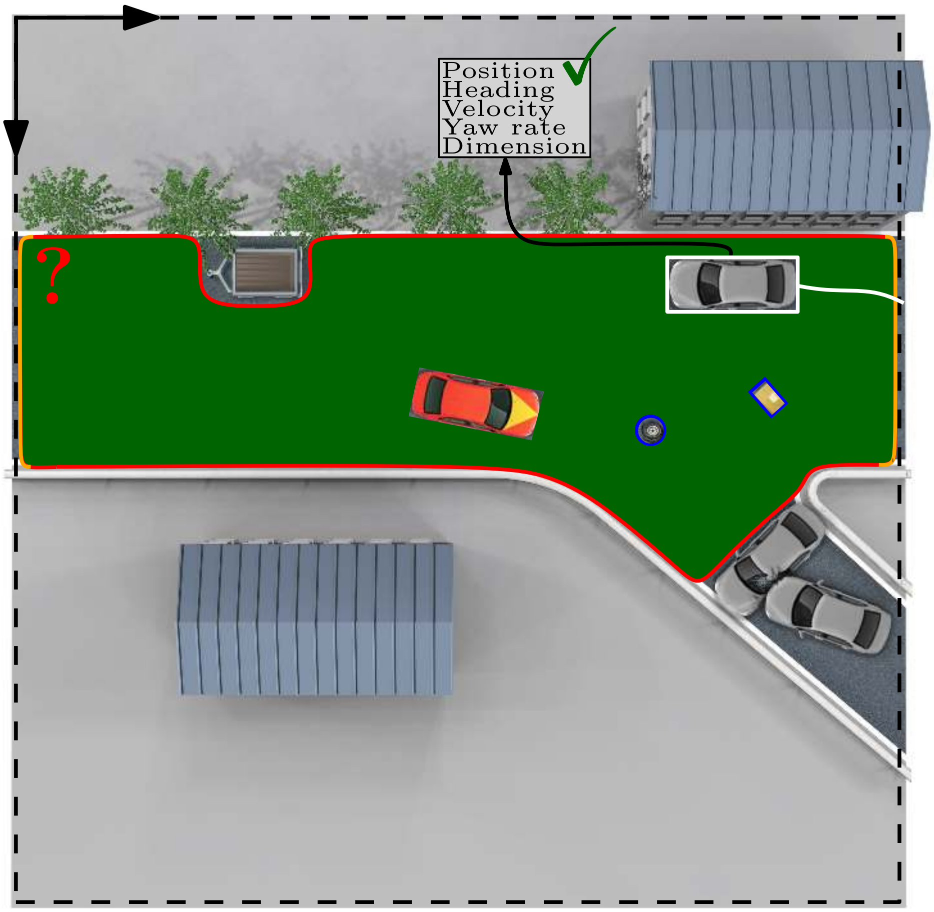


Figure 5.1: PFS map enclosing relevant static free space shown in green. The already treated dynamic object map is additionally overlaid in white.

5.2 Related Work

Methods of free space detection and estimation in the intelligent vehicles domain can be subdivided into the following categories:[1]

- Grid map-free methods [51, 135, 160, 195, 265].
- Grid map-based methods:
  - Grid segmentation by thresholding [41, 106, 121, 137, 174, 179].
  - Grid segmentation by dynamic programming [11, 12].

[1] Vision-based methods that focus on the detection and tracking of lane markings with specific road models [89] or perform road terrain detection based on visual appearance features, e.g. color or texture, are not included in the discussion as these are not in direct scope of this thesis and should ideally be combined with the proposed metric PFS map representation. For an overview of such methods, the reader is referred to [85, 89, 126, 171] and references therein.

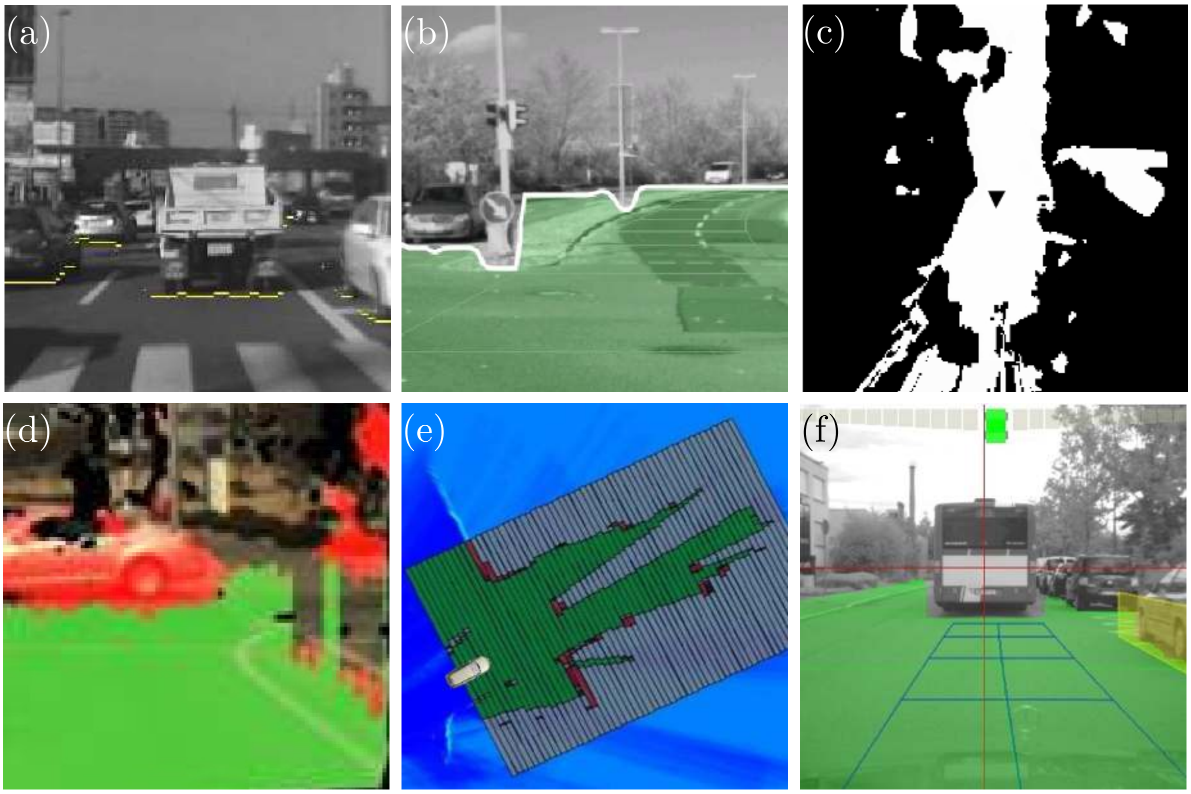


Figure 5.2: Examples of free space detection and estimation methods. By global optimization with respect to planar [135] (a) and non-planar roads [265] (b), via global grid thresholding [174] (c), by polar grid thresholding [121] (d), interval-based [41] (e), by grid segmentation via dynamic programming [11] (f). Images taken from the mentioned sources.

An example of a grid map-free approach to free space detection can be found in [51], in which inverse perspective mapping is applied to a camera image at two consecutive time steps. These are subtracted from each other after compensating for ego motion, which results in peaks in the difference image where obstacles protrude from the planar road surface. This difference image is then thresholded to obtain available free space. In [160], free space in front of the ego vehicle is detected with an automotive radar. The algorithm is, however, restricted to an estimation of the distance to the left and right road borders and cannot detect obstacles on the road. The algorithm described in [135] is based on gradient directions in the left and right images of a rectified stereo image pair. The matching gradient directions of the two images are used to verify disparities of obstacles that limit the free space. A consistent road-obstacle boundary is found by global optimization, efficiently realized by dynamic programming, see Fig. 5.2a. In [265], this algorithm is extended to handle non-planar roads, see Fig. 5.2b. Another grid map-free free space detection approach can be found in a current variant of the stixel world generation described in [195]. Here, free space and obstacles are similarly segmented in one single global optimization step directly without an intermediate grid representation.

The disadvantage of all approaches mentioned so far is that they only describe currently visible free space. In contrast, methods of the second category, which rely on an occupancy grid map-based representation, can take advantage of older free space information as once mapped static areas are still available in the grid. In the simplest case of threshold-based free space detection, the whole grid is thresholded to classify free and occupied regions [137, 174]. This approach, however, results in many unconnected free areas as can be seen in Fig. 5.2c, which makes such a free space representation not very beneficial for subsequent algorithms. Therefore, the majority of methods cast virtual rays from the ego vehicle in all directions of the grid and consider the space up to the point until these rays first cross an occupancy threshold as free space [106], see Fig. 5.2d. For this purpose, often a polar coordinate system is introduced so that rays are parallel to one coordinate axis and searching for obstacles in ray direction becomes straightforward [121,179]. The disadvantage of these methods is that they exclude mapped free space information which is not in direct line of sight but potentially still important and available – either due to the memory effect of the grid map environment representation or because sensors mapped areas behind obstacles.[2] Another threshold-based method results in an interval-based free space representation [41], see Fig. 5.2e.[3] Here, all cells in the occupancy grid that belong to the same potential interval are traversed in parallel to the ego vehicle's longitudinal axis with the help of the Bresenham algorithm. The dominant state of the classes *free*, *occupied*, and *unknown* is extracted for the cells of each traversed line via thresholding and cells of adjacent, parallel lines are matched to the same interval if they have the same dominant state. The result is a compact free space representation. Due to the rough discretization in longitudinal direction, potentially relevant details contained in the underlying grid are, however, suppressed. Moreover, the dependence of the data structure on the vehicle orientation may complicate subsequent environment recognition and interpretation algorithms such as junction or roundabout detection algorithms.

The disadvantage of all grid-based free space detection methods men-

[2] This can happen, for example, if a higher mounted camera overviews small obstacles as boxes or the radar maps areas behind other vehicles.

[3] Note that the related interval maps [41, 267] can also be mapped directly without an intermediate grid representation.

tioned so far is that no additional spatial and temporal smoothness constraints are introduced. Although the occupancy grid itself provides some temporal smoothness because of the recursive mapping, e.g. with BBFs, this kind of smoothness constraint is limited to individual cells due to the underlying stochastic independence assumption between grid cells. The grid mapping filters therefore contribute only partially to the stabilization of the subsequent free space segmentation. Consequently, the results are often susceptible to noise, which especially holds for approaches applied to online maps generated by off-the-shelf, standard automotive sensors. To overcome these disadvantages, the use of dynamic programming has been suggested in [11] to find an optimal path cutting a polar grid from left to right, see Fig. 5.2f.[4] Spatial smoothness is imposed by a cost that penalizes jumps in depth, whereas a cost that penalizes the deviation of the predicted segmentation result of the previous time step from the current solution enforces temporal smoothness. To cope with the problem of mistakenly finding the free space boundary on the background object and not on the first object in ray direction, background subtraction is applied in [12] before dynamic programming. This is implemented by marking occupied cells behind the first maximum above a threshold in ray direction as free. With this approach, free space behind the first obstacle in the current line of sight can, however, not be detected any more.

Conclusion

Free space detection and estimation methods that rely on occupancy grid maps have the advantage of sensor independence and can thus be applied for all sensor modalities in the same way. Additionally, these methods take previously observed free space into account, which is vital not only for evasive trajectory planning but also for environment interpretation. Existing grid-based free space detectors, however, either i) yield many disconnected free space areas difficult for interpretation, ii) employ only a rough spatial free space interval discretization without additional temporal stabilization, or iii) exclude mapped free space that is not directly in the line of sight but still available either due to the memory effect of the grid or because sensors mapped free areas behind obstacles. Moreover, the results still contain irrelevant free space information in form of unreachable areas, which complicates subsequent tasks such as trajectory planning. In addition, the common methods do not provide a parametric, continuous representation

[4] This approach also constitutes an intermediate step in the original, grid-based generation of the stixel world, see [12].

of arbitrarily shaped, drivable free space but still represent the segmentation results in form of discrete cells or intervals. Parametric descriptions are to date only common for road course models or road boundary estimators and not for general free space estimators.[5] Common road course models are, for example, circular models [107], cubic clothoid approximations [131], clothoids [65], or, recently, spline functions [9,156,243,264,265] for greater flexibility.

The proposed PFS map extends these approaches to model the complete free space around the ego vehicle by a combination of a dynamic, closed B-spline curve, whose attributed control points along with a geometric primitives representation of inner free space boundaries serve as a novel, compact representation of the static environment in form of available free space. Consequently, the resulting PFS map encodes relevant information contained in the grid in a parametric form, which greatly reduces the required storage and transmission bandwidth far beyond approaches that tackle the problem by pure grid compression techniques [91]. The continuous representation additionally avoids discretization effects and provides easy to parametrize spatial and temporal smoothness constraints. The additional suppression of irrelevant free space information simplifies situation analysis and trajectory planning and reduces the required transmission bandwidth even further.

5.3 Generation of Parametric Free Space Maps

In the following, the proposed approach for generating PFS maps is presented. Excerpts have already been published by the author in [221, 223, 226].

5.3.1 System Overview

Fig. 5.3 shows an overview of the proposed PFS map generation approach. It consists of two main parts: The free space detection by a grid map image analysis (blue) and the subsequent free space tracking (yellow). The input of the detection step is the optimized occupancy grid map $\boldsymbol{m}_{\mathrm{opt},k}$ such

[5] Although road models provides free space information to the left and right of the ego vehicle, they do not capture obstacles on the road, for example. Furthermore, methods of this kind do not work in completely unstructured environments such as parking lots or complex inner-city environments.

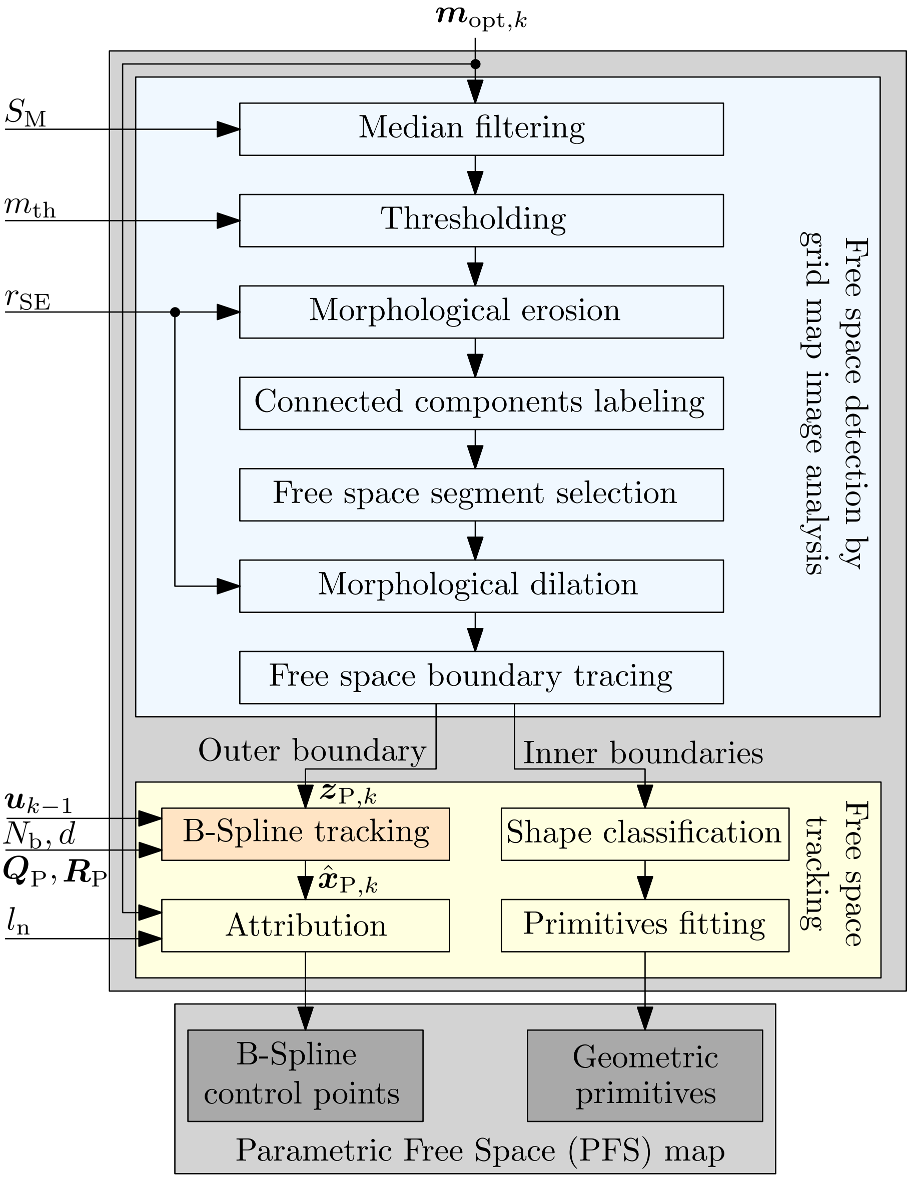


Figure 5.3: Overview of one time step of the proposed PFS map generation approach. Inputs are the optimized, static grid map $\boldsymbol{m}_{\mathrm{opt},k}$ and ego motion $\boldsymbol{u}_{k-1}$. The output is the PFS map. Parameters are explained in the text.

as generated by the dynamic mapping approach described in the previous Chapter 4. The detection step results in extracted free space boundaries, which constitute the measurements for the subsequent free space tracking. Both parts are explained in detail in the following subsections and further illustrated by the application to an exemplary scene as shown in Fig. 5.4a.

5.3.2 Free Space Detection by Grid Map Image Analysis

The main aspect of the grid-based free space detection step is to treat the grid map as an image and to employ methods well-known from the image analysis domain to extract relevant outer and inner free space boundaries.[6] Two variants of this grid map image analysis have been developed in the course of this thesis. In the following, we focus on the computationally less demanding version. Readers interested in the computationally more complex variant based on a full gray-level morphological image analysis are referred to [221].[7]

First, a median filter with neighborhood size S_{M} is applied to the grid map $\boldsymbol{m}_{\mathrm{opt},k}$ to get rid of noisy pixels (Fig. 5.4c). This nonlinear, rank-order filter is chosen because it preserves grid map edges and boundaries without blurring [116]. This aspect is important as the distances to the free space boundaries shall not be modified by the filtering operation. Afterwards, a pixel-based segmentation is performed by a simple thresholding operation with threshold m_{th} on the free probability (Fig. 5.4d). Subsequently, a morphological erosion [116] with a disc-shaped, flat structuring element of a radius r_{SE} equal to the vehicle's width is applied to the free space areas, which are treated as foreground structures for this operation.[8] The general result of an erosion of a set $\mathcal{F}$ with a structuring element $\mathcal{S}$ is the set of all points $\boldsymbol{p}$ such that $\mathcal{S}$ is included in $\mathcal{F}$ when the origin of $\mathcal{S}$ is placed at $\boldsymbol{p}$ [234]. The eroded set $\mathcal{E}_{\mathcal{S}}(\mathcal{F})$ is formally given by [234]

$$\mathcal{E}_{\mathcal{S}}(\mathcal{F}) = \{\boldsymbol{p} \mid \mathcal{S}_{\boldsymbol{p}} \subseteq \mathcal{F}\}. \tag{5.1}$$

This is further illustrated in Fig. 5.5a, in which $\mathcal{F}$ corresponds to the set of free cells and $\mathcal{S}$ to the disc-shaped, flat structuring element with origin in its center. The erosion thus artificially reduces the size of the free space, so that free space segments into which the vehicle does not fit are removed. Moreover, larger free space areas that are joined together only by narrow,

[6] Although the similarity between operations on occupancy grids and image processing has already been stated in one of the seminal articles about grid mapping [77], this connection is rather underexploited in literature – probably due to the rather distinct robotics and image processing communities.

[7] In short, the alternative approach uses the H-Minima transformed Beucher gradient image of the morphologically opened gray-level grid map as a segmentation function for a subsequent watershed transformation [168] to extract relevant free space areas.

[8] This is an important difference to most morphological image processing systems that treat objects (and not the absence of objects) as foreground structures.

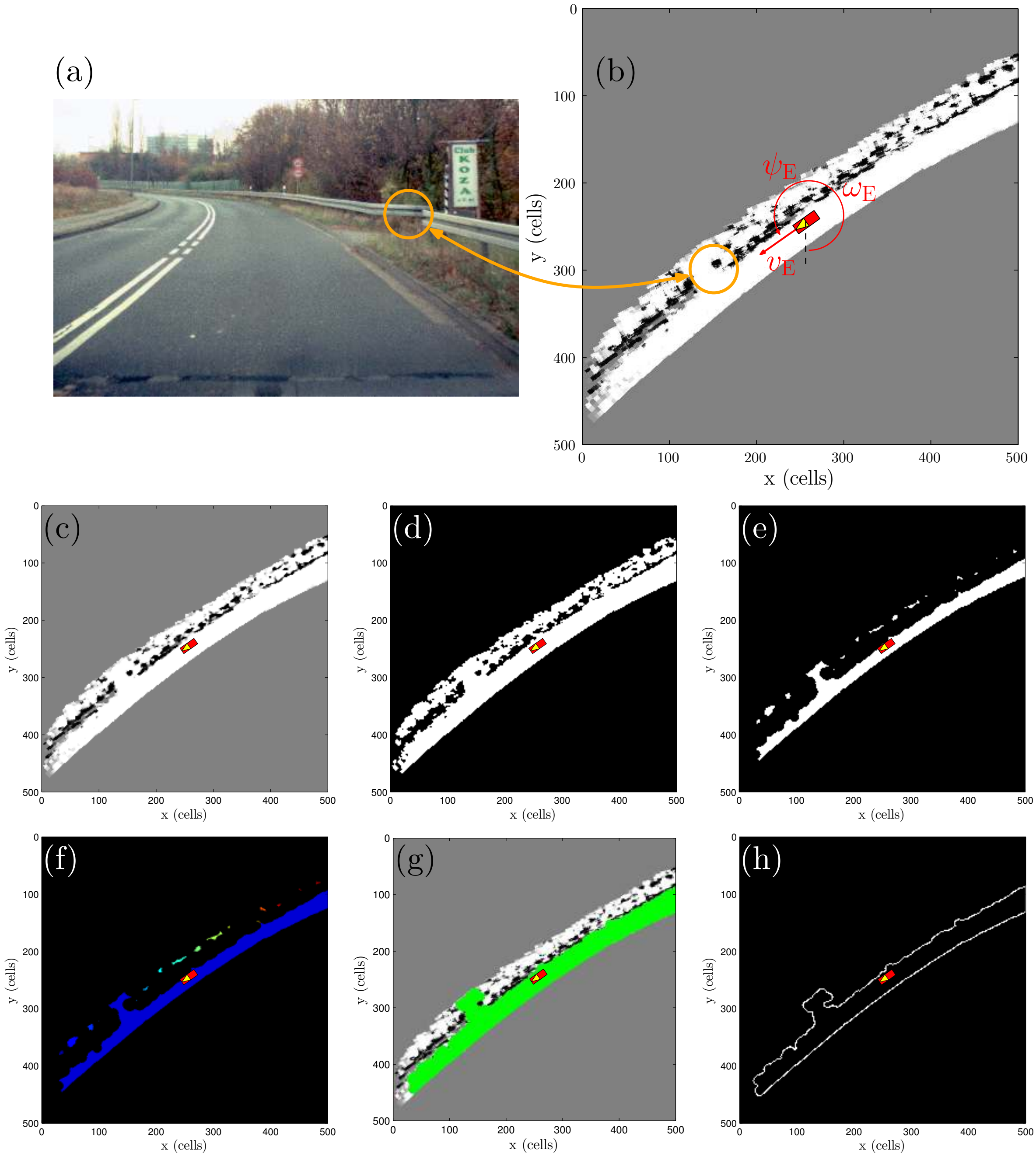


Figure 5.4: Free space detection by grid map image analysis in an exemplary driving scenario (a), which contains a rural road with trees behind the right guardrail and a turnoff to the right. The corresponding grid map $\boldsymbol{m}_{\mathrm{opt},k}$ is shown in (b). After a median filtering (c), the grid is segmented (d), followed by a morphological erosion (e), a connected components labeling (f), a morphological dilation of the selected, relevant free space segment (g), and a boundary tracing (h). Extracted boundary cells are treated as measurements for the subsequent free space tracking step.

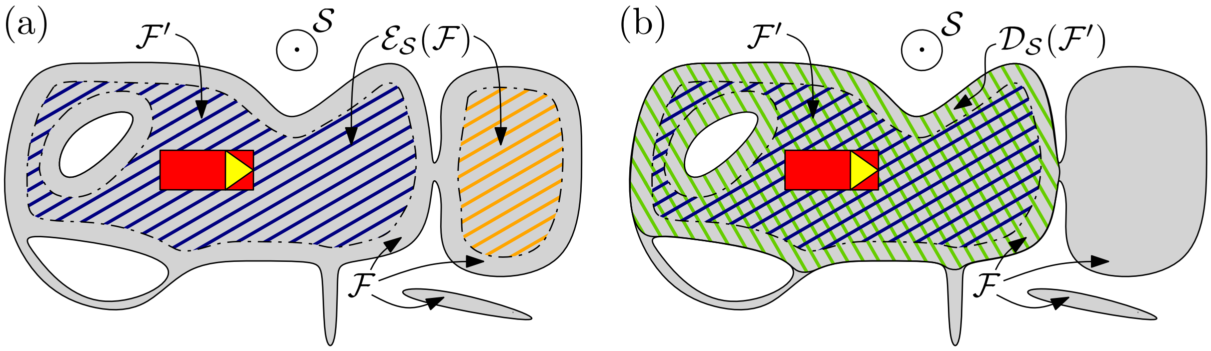


Figure 5.5: Morphological free space erosion and dilation. The erosion of a set of free space cells $\mathcal{F}$, shown in gray, with a structuring element $\mathcal{S}$ results in the eroded set $\mathcal{E}_\mathcal{S}(\mathcal{F})$ shown hatched in (a). Small free space areas such as the one at lower right are deleted, and impassable narrowings are separated from each other. The *relevant* eroded subset $\mathcal{F}'$, i.e. the one in which the ego vehicle (red) is located, is hatched in blue – the orange subset is erased. In (b), the dilation of $\mathcal{F}'$ is shown, which results in the set $\mathcal{D}_\mathcal{S}(\mathcal{F}')$ hatched in green. The reachable free space is thus brought back to its original size.

impassable connections are separated from each other by the erosion. In the exemplary scene, the result of the erosion is visualized in Fig. 5.4e. It is evident that all unreachable free space areas behind the guardrail are either removed or separated from each other. The large entry of the turnoff is, in contrast, still connected as the ego vehicle could perform a turn to the right. Note that objects are never erased or shrunk by the erosion, only free space is artificially reduced. In other words, the erosion can only extend objects and reduce free space and not vice versa. The processing step is thus conservative with regard to safety applications.

After the erosion, a fast connected components labeling[9] is performed. Differently labeled segments are visualized with different colors in Fig. 5.4f. This step is followed by the selection of the important free space segment, i.e. the segment in which the ego vehicle is currently located. An obvious way to realize this selection would be just to use the segment that contains the ego vehicle's front. This approach does, however, not work in all cases. As free space segments are eroded and therefore shrunk beforehand, it is theoretically possible that there is no eroded free space segment at this position but that the relevant segment begins several meters ahead of the vehicle. Therefore, it is necessary to search this segment in front of the

[9] A pure flood fill operation is not sufficient because it is not known beforehand which cell will be included in the correct free space segment in all cases. Consequently, no seed point can be specified.

ego vehicle. For this purpose, virtual rays of limited length are cast in driving direction of the vehicle by means of the Bresenham algorithm [43] and the first free space segment that is hit is considered the relevant one. In the example, this corresponds to the large blue segment in Fig. 5.4f. The remaining free space segments are erased as they are unreachable for the ego vehicle. In Fig. 5.5a, the orange hatched subset of $\mathcal{E}_{\mathcal{S}}(\mathcal{F})$ to the right is therefore deleted. The relevant free space segment $\mathcal{F}'$, in contrast, is subsequently dilated with the same structuring element to bring the *reachable* free space back to its original size. The general result of a dilation of a set $\mathcal{F}'$ with a structuring element $\mathcal{S}$ is the set of all points $\boldsymbol{p}$ such that $\mathcal{S}$ intersects $\mathcal{F}'$ when its origin is placed at $\boldsymbol{p}$ [234]. The dilated set $\mathcal{D}_{\mathcal{S}}(\mathcal{F}')$ is therefore formally given by [234]

$$\mathcal{D}_{\mathcal{S}}(\mathcal{F}') = \{\boldsymbol{p} \mid \mathcal{S}_{\boldsymbol{p}} \cap \mathcal{F}' \neq \emptyset\}. \tag{5.2}$$

This is further illustrated in Fig. 5.5b, in which $\mathcal{F}'$ corresponds to the set of free cells of the relevant, previously eroded free space segment. For the given exemplary driving scene, the result of the dilation is visualized in green in Fig. 5.4g overlaid over the original grid. In both cases, it becomes obvious that the original size of the relevant free space area is recovered by this processing step.

Readers familiar with morphological operations might wonder why no direct morphological opening is applied as the combination of erosion and dilation corresponds to an opening operation due to the same, symmetric structuring element. The general result of a morphological opening of a set $\mathcal{F}$ with a structuring element $\mathcal{S}$ is the union of all structuring elements that fit the set [234]. The opened set $\mathcal{O}_{\mathcal{S}}(\mathcal{F})$ is therefore formally given by [234]

$$\mathcal{O}_{\mathcal{S}}(\mathcal{F}) = \bigcup_{\boldsymbol{p}} \{\mathcal{S}_{\boldsymbol{p}} \mid \mathcal{S}_{\boldsymbol{p}} \subseteq \mathcal{F}\}. \tag{5.3}$$

A morphological opening applied directly to free space foreground structures would consequently directly remove all free space areas into which the structuring element does not fit and preserve the rest, exactly as desired. The reason why this approach is, however, not suitable for the task at hand is that free space areas in which the ego vehicle fits but which are connected by *short*, narrow, impassable passages do potentially not get disconnected. This undesired effect is visualized in Fig. 5.6, in which a morphological opening is applied to the original set of free space cells $\mathcal{F}$ directly. It is evident that the unreachable area to the right is still preserved after the opening because the structuring elements positioned to

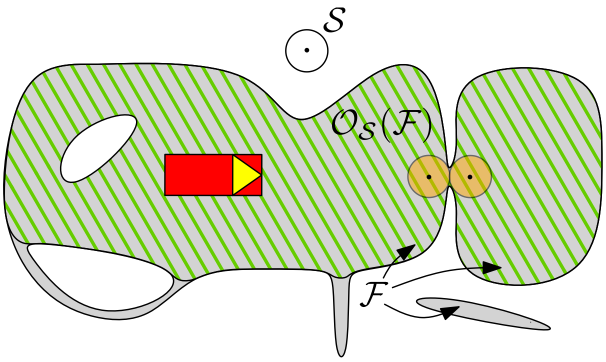


Figure 5.6: Morphological opening example. The opening applied to a set of free space cells $\mathcal{F}$ shown in gray with a structuring element $\mathcal{S}$ results in the opened set $\mathcal{O}_{\mathcal{S}}(\mathcal{F})$ hatched in green. The unreachable free space area to the right is still preserved because the exemplary structuring elements shown in orange overlap.

the left and right of the short narrowing overlap. The two step procedure of erosion and dilation, in contrast, permits to delete the separated right subset of the eroded set before performing the dilation.

These steps are followed by a boundary tracing with the algorithm described in [241] to extract outer and inner free space boundaries of the dilated, final free space segment. The result is shown in Fig. 5.4h. Outer boundary cells are used as measurements for a continuous, dynamic B-Spline free space contour tracking, whereas the shapes of the inner boundaries are subsequently classified and described by geometric primitives. Both steps are explained in the following subsections.

5.3.3 Dynamic B-Spline Free Space Contour Tracking

Before the free space tracking of the outer free space boundary is explained, the relevant basics of B-spline functions and curves are shortly presented in the first place.

B-Spline Functions and Curves

A *B-spline function* is a piecewise polynomial function that consists of $L \in \mathbb{N}$ concatenated polynomial segments of order[10] $d \in \mathbb{N}$, the so-called spans $S_0, \dots, S_{L-1}$, which are joined together at breakpoints [35].

[10] The order of a polynomial is the number of its coefficients and different from its degree, which is equal to its highest exponent [35].

B-splines are a computationally efficient representation of splines, where a spline function is represented as a weighted sum of N_b basis functions $B_n(s)$ with $n \in \{0, \ldots, N_\mathrm{b} - 1\}$.

A parametric *B-spline curve* consists of B-spline coordinate functions. With the coordinate functions $x(s)$ and $y(s)$, a parametric B-spline curve $\boldsymbol{r}(s) \in \mathbb{R}^2$, defined continuously over $s \in [0, L]$, can be compactly written as

$$\boldsymbol{r}(s) = \begin{pmatrix} x(s) \\ y(s) \end{pmatrix} = \underbrace{\begin{pmatrix} \boldsymbol{b}(s)^T & \mathbf{0} \\ \mathbf{0} & \boldsymbol{b}(s)^T \end{pmatrix}}_{\boldsymbol{U}(s)} \boldsymbol{x}_\mathrm{P}, \tag{5.4}$$

with the vector of B-spline basis functions

$$\boldsymbol{b}(s) = \big(B_0(s)\ B_1(s)\ \ldots\ B_{N_\mathrm{b}-1}(s)\big)^T \in \mathbb{R}^{N_\mathrm{b}}, \tag{5.5}$$

and the B-spline control vector

$$\boldsymbol{x}_\mathrm{P} = (q_{x,0}\ \cdots\ q_{x,N_\mathrm{b}-1}\ q_{y,0}\ \cdots\ q_{y,N_\mathrm{b}-1})^T \in \mathbb{R}^{2N_\mathrm{b}} \tag{5.6}$$

according to [35, 66]. The behavior of each coordinate function is influenced by weighting its basis functions by the components of the control vector – the control points or de Boor points $\boldsymbol{q}_i = (q_{x,i}, q_{y,i})$ [66]. Basis functions of order d with N_k knots at $k_0 \leq k_1 \leq \ldots \leq k_{N_{\mathrm{k}-1}}$ can be constructed by a recursion formula with the ground instance

$$B_{n,1}(s) = \begin{cases} 1, & \text{if } k_n \leq s < k_n + 1, \\ 0, & \text{otherwise,} \end{cases} \tag{5.7}$$

and the inductive step

$$B_{n,d}(s) = \frac{(s - k_n) B_{n,d-1}(s)}{k_{n+d-1} - k_n} + \frac{(k_{n+d} - s) B_{n+1,d-1}(s)}{k_{n+d} - k_{n+1}}, \tag{5.8}$$

in which the summands are treated as zero if the denominator vanishes [35]. If the knot spacing is uniform, the B-spline is called uniform as well, otherwise non-uniform. To obtain closed curves, the basis functions have to be taken as periodic over the interval $s \in [0, L]$. Fig. 5.7 exemplarily shows a uniform, quadratic, periodic B-spline function $x(s)$ with its corresponding control points, spans, knots, and basis functions to visualize the dependencies.

For spline tracking, the B-spline curve is not treated as deterministic but as probabilistic so that the B-spline control vector $\boldsymbol{x}_\mathrm{P}$ is a random

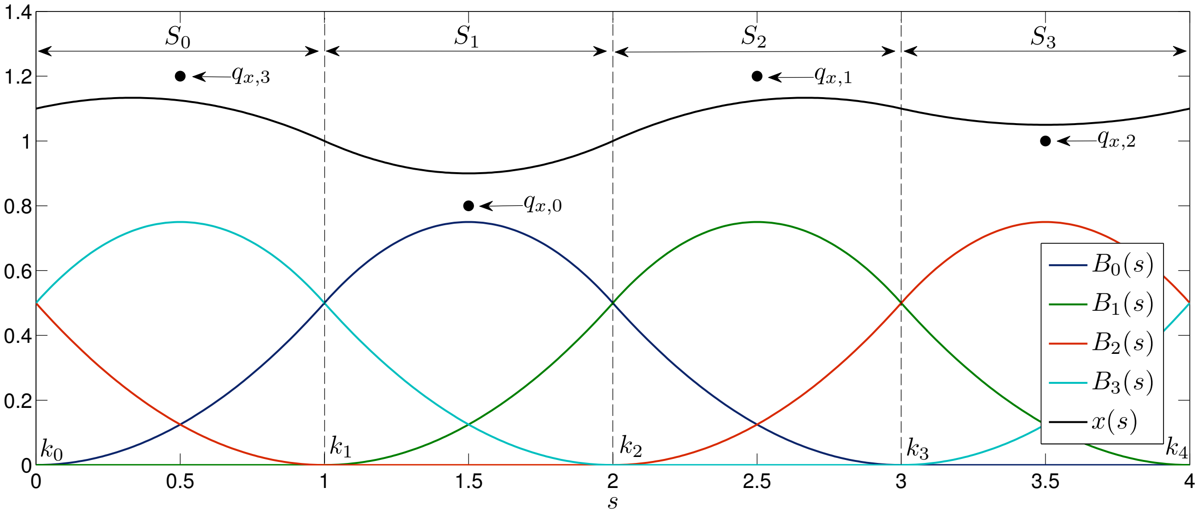


Figure 5.7: Uniform, quadratic ($d = 3$), periodic B-spline function $x(s)$ with $L = N_\mathrm{b} = 4$, $N_\mathrm{k} = 5$, its corresponding control points $q_{x,0}, \dots, q_{x,3}$, spans $S_0, \dots, S_3$, knots $k_0, \dots, k_4$, and (periodic) B-spline basis functions $B_0, \dots, B_3$.

vector. This circumstance is further illustrated in Fig. 5.8, which shows samples drawn from a distribution of spline curves with Gaussian distributed de Boor points.

The important advantages of B-splines for approximation, and especially for the online tracking application at hand, are the following:

- Each control point influences the curve's shape only locally, so that small changes in the control vector entail only small changes in the resulting curve. This compact support property of B-splines [35] permits to stabilize the free space curve estimation by constraining the motion of its control points, which would not be possible with Bézier splines.

- B-splines do not suffer from disturbing, oscillatory effects [35] in contrast to polynomial approximations.

- B-splines can be evaluated efficiently with precalculated span and placement matrices [35].

- B-spline curves are invariant under affine transformations [32]. On demand of subsequent situation interpretation algorithms, the resulting free space curve can therefore easily be transformed from a grid map-fixed to an ego vehicle-fixed coordinate system by simply transforming each control point individually.

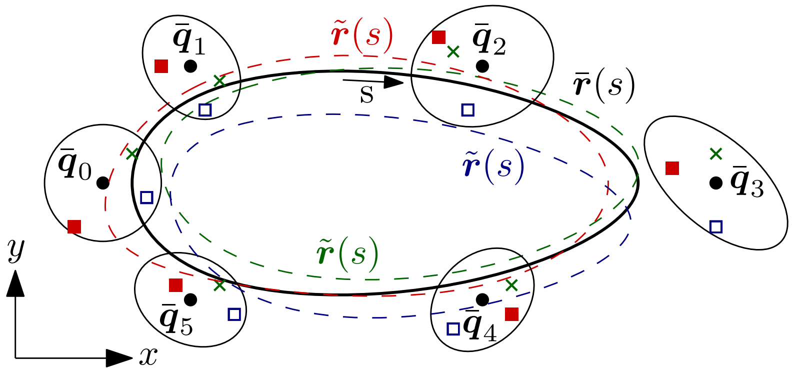


Figure 5.8: Probabilistic B-spline curve with Gaussian distributed de Boor points $\boldsymbol{q}_i$ centered around their means $\mathbb{E}(\boldsymbol{q}_i) = \bar{\boldsymbol{q}}_i$ with corresponding uncertainty ellipses. The curve's mean shape $\bar{\boldsymbol{r}}(s)$ is depicted in black; further exemplary samples $\tilde{\boldsymbol{r}}(s)$ with corresponding de Boor points are shown in color.

The following section clarifies how an IF is employed to recursively estimate and track the time-varying B-spline control vector of a deforming, moving, closed free space contour.

Motion and Measurement Models

The extracted cells of the larger outer boundary serve as a measurement vector for the recursive estimation and tracking of a closed B-spline curve. The curve is predicted from time step to time step by compensating ego motion via a CTRV model in analogy to the dynamic mapping and tracking approach already described in Section 4.3.3. Therefore, the system dynamic equation is given by

$$\boldsymbol{x}_{\mathrm{P},k+1} = \boldsymbol{x}_{\mathrm{P},k} + \boldsymbol{u}_{\mathrm{P},k} + \boldsymbol{w}_{\mathrm{P},k}, \tag{5.9}$$

with B-spline control vector $\boldsymbol{x}_{\mathrm{P},k} \in \mathbb{R}^{2N_{\mathrm{b}}}$, zero-mean, white, Gaussian process noise vector $\boldsymbol{w}_{\mathrm{P},k} \in \mathbb{R}^{2N_{\mathrm{b}}}$ with associated process noise covariance matrix $\boldsymbol{Q}_{\mathrm{P},k} \in \mathbb{R}^{2N_{\mathrm{b}} \times 2N_{\mathrm{b}}}$, and artificial input vector

$$\boldsymbol{u}_{\mathrm{P},k} = \begin{pmatrix} -\frac{v_{\mathrm{E},k}}{c\omega_{\mathrm{E},k}}\big(\cos\psi_{\mathrm{E},k} - \cos(\omega_{\mathrm{E},k}T + \psi_{\mathrm{E},k})\big) \cdot \mathbb{1} \in \mathbb{R}^{N_{\mathrm{b}}} \\ -\frac{v_{\mathrm{E},k}}{c\omega_{\mathrm{E},k}}\big(\sin(\omega_{\mathrm{E},k}T + \psi_{\mathrm{E},k}) - \sin\psi_{\mathrm{E},k}\big) \cdot \mathbb{1} \in \mathbb{R}^{N_{\mathrm{b}}} \end{pmatrix} \tag{5.10}$$

for ego motion compensation in spirit of (4.5), p. 82. The process noise has to compensate for errors in ego motion and, more important, for the free space contour deformation. As the ego motion compensated contour deformation is not predictable in a deterministic way, it is modeled as static with a stochastic diffusion large enough to permit adequate deformations but small enough to effectively prevent fast, unrealistic contour changes that cannot take place in one sampling time step. The prediction model can therefore also be seen as a soft constraint on the B-spline contour deformation over time.

The predicted contour is subsequently updated with new boundary measurements to estimate and keep track of the time-varying B-spline control vector $\boldsymbol{x}_{\mathrm{P},k}$ of the deforming, moving, closed free space contour. For this purpose, all extracted outer boundary cells are treated as equally distributed along s in order to obtain the appropriate s_i-locations of the N_p coordinate pairs of the measurement points along the curve, which form the measurement vector

$$\boldsymbol{z}_{\mathrm{P},k} = \big(z_x(s_1)\ z_y(s_1)\ \dots\ z_x(s_{N_\mathrm{p}})\ z_y(s_{N_\mathrm{p}})\big)^T \in \mathbb{R}^{2N_\mathrm{p}}. \tag{5.11}$$

This is a reasonable assumption because all neighboring boundary cells have the same distance from each other due to the preceding boundary tracing step. Care has, however, to be taken that the boundary search always starts in the same direction and near the predicted location of the boundary tracing starting point of the last time step to prevent wrong data associations during spline tracking and to minimize curve reparametrization errors.[11] Using (5.4), the measurement matrix $\boldsymbol{H}_{\mathrm{P},k}$ can be concatenated according to

$$\boldsymbol{H}_{\mathrm{P},k} = \big(\boldsymbol{U}(s_1)\ \dots\ \boldsymbol{U}(s_{N_\mathrm{p}})\big)^T \in \mathbb{R}^{2N_\mathrm{p}\times 2N_\mathrm{b}}, \tag{5.12}$$

so that the measurement equation is finally given by

$$\boldsymbol{z}_{\mathrm{P},k} = \boldsymbol{H}_{\mathrm{P},k}\boldsymbol{x}_{\mathrm{P},k} + \boldsymbol{v}_{\mathrm{P},k}, \tag{5.13}$$

with Gaussian measurement noise vector $\boldsymbol{v}_{\mathrm{P},k} \in \mathbb{R}^{2N_\mathrm{p}}$ and associated measurement covariance matrix $\boldsymbol{R}_{\mathrm{P},k} \in \mathbb{R}^{2N_\mathrm{p}\times 2N_\mathrm{p}}$. Note that the dimensions of $\boldsymbol{H}_{\mathrm{P},k}$, $\boldsymbol{R}_{\mathrm{P},k}$, and $\boldsymbol{v}_{\mathrm{P},k}$ change over time because the number of boundary measurements is not constant.

[11] As an extreme example, the exact same curve can be parametrized by a curve parameter s running clockwise or counterclockwise. If the parametrization of the predicted contour differs from the one employed in the measurement update, then the tracking fails.

Filtering

The filtering itself is realized with the help of an IF as described in Section 3.1.3. This filter choice turned out to be at least an order of magnitude faster than a standard KF formulation because of the large measurement and relatively small state vector dimension. The result of the filtering process under the taken LGS assumptions is the estimated information vector $\hat{\boldsymbol{\imath}}_{\mathrm{P},k}$ and information matrix $\boldsymbol{I}_{\mathrm{P},k}$ of the B-spline control vector. For output purpose, the information-based parametrization is transformed back to the equivalent moment parametrization of the Gaussian posterior pdf according to

$$\hat{\boldsymbol{x}}^{\mathrm{M}}_{\mathrm{P},k} = \boldsymbol{I}^{-1}_{\mathrm{P},k} \hat{\boldsymbol{\imath}}_{\mathrm{P},k}, \tag{5.14a}$$

$$\boldsymbol{P}_{\mathrm{P},k} = \boldsymbol{I}^{-1}_{\mathrm{P},k}, \tag{5.14b}$$

so that the filter finally provides the MMSE estimate of the B-spline control vector $\hat{\boldsymbol{x}}^{\mathrm{M}}_{\mathrm{P},k}$ and, upon request, its associated error covariance matrix $\boldsymbol{P}_{\mathrm{P},k}$, see Section 3.1.3. If no measurement is available, just the prediction step is carried out so that short detection fails can be bridged.

Comparing the proposed spline tracking to other approaches such as the active contour tracking described in [35], two important differences are evident. First, the velocity of the contour is not estimated because it is known from the ego vehicle's sensors. Second, measurements are not generated by searching for adequate image features along the normals of a predicted contour, but the boundary cells are extracted again (redetected) in each time step for greater stability.

Filter Parametrization

It remains to specify the B-spline and filter parameters. For the former, we use uniform, quadratic ($d = 3$), periodic B-spline basis functions, therefore $L = N_{\mathrm{b}} = N_{\mathrm{k}} - 1$ and $\{k_0, \dots, k_{N_{\mathrm{k}}-1}\} = \{0, \dots, L\}$, which are considered a good compromise between performance and approximation quality and result in continuously differentiable (C^1-continuous) free space contours. The *spatial* smoothness of the estimates $\hat{\boldsymbol{x}}^{\mathrm{M}}_{\mathrm{P},k}$ can easily be adjusted by changing N_{b}. The *temporal* smoothness, in contrast, can be altered by adapting the process noise covariance matrix $\boldsymbol{Q}_{\mathrm{P},k}$ for a given measurement noise covariance matrix $\boldsymbol{R}_{\mathrm{P},k}$. In the developed system, the number of de Boor points N_{b} is fixed to 70, which has proven to be sufficient to describe even very complex static environments around the vehicle. The initialization is realized by setting both the initial information vector $\hat{\boldsymbol{\imath}}_{\mathrm{P},0}$

and the information matrix $\boldsymbol{I}_{\mathrm{P},0}$ equal to the zero vector and matrix, respectively. This procedure reflects complete initial ignorance. Note that an optimal filter initialization is less crucial for the free space tracking than for the dynamic cell/object tracking described in the previous chapter. This is because the "free space object" is normally available and detected in every time step. A reinitialization is thus rarely necessary and the behavior of the filter's initial transient consequently less important. The final filter parameters are summarized in Table 5.2, p. 127.

Boundary Attribution

Now that the B-spline curve is estimated, it needs to be decided which parts of the curve constitute delimiters to obstacle boundaries and which are delimiters to unknown areas, e.g. stemming from the limited sensor field of view or occlusions. Such a boundary attribution can be realized by casting virtual rays of limited length l_n along the normals

$$\boldsymbol{n}_k(s) = \begin{pmatrix} n_{x,k}(s) \\ n_{y,k}(s) \end{pmatrix} = \begin{pmatrix} \frac{-\mathrm{d}\hat{y}_k(s)}{\mathrm{d}s} \\ \frac{\mathrm{d}\hat{x}_k(s)}{\mathrm{d}s} \end{pmatrix} = \begin{pmatrix} \mathbf{0} & -\frac{\mathrm{d}\boldsymbol{b}(s)}{\mathrm{d}s}^T \\ \frac{\mathrm{d}\boldsymbol{b}(s)}{\mathrm{d}s}^T & \mathbf{0} \end{pmatrix} \hat{\boldsymbol{x}}_{\mathrm{P},k}^{\mathrm{M}} \tag{5.15}$$

of the estimated B-spline free space curve

$$\hat{\boldsymbol{r}}_k(s) = \begin{pmatrix} \hat{x}_k(s) \\ \hat{y}_k(s) \end{pmatrix} = \boldsymbol{U}(s)\hat{\boldsymbol{x}}_{\mathrm{P},k}^{\mathrm{M}} = \begin{pmatrix} \boldsymbol{b}(s)^T & \mathbf{0} \\ \mathbf{0} & \boldsymbol{b}(s)^T \end{pmatrix} \hat{\boldsymbol{x}}_{\mathrm{P},k}^{\mathrm{M}} \tag{5.16}$$

into its exterior, rasterizing these rays on the grid with the help of the Bresenham algorithm [43], and checking the corresponding cells for objects. If none are found within a span of the B-spline curve, the attribute "unknown environment boundary" is attached to its associated spline control point. For the exemplary scene, the resulting, final PFS map is visualized overlaid over the extracted free space boundary cells in Fig. 5.9a and over the original grid map $\boldsymbol{m}_{\mathrm{opt},k}$ in Fig. 5.9b. The contour smoothly follows the traced outer boundary. Unreachable areas behind the guardrail are suppressed, whereas the potentially relevant turnoff to the right remains included. Existing road model estimators and free space estimators that are based on thresholding in ray direction would exclude areas such as the turnoff from their description. Additionally, the distinction between free space delimited by obstacle boundaries (red) and boundaries to unknown environment (orange) becomes visible, which is the result of the limited sensor field of view in this example.

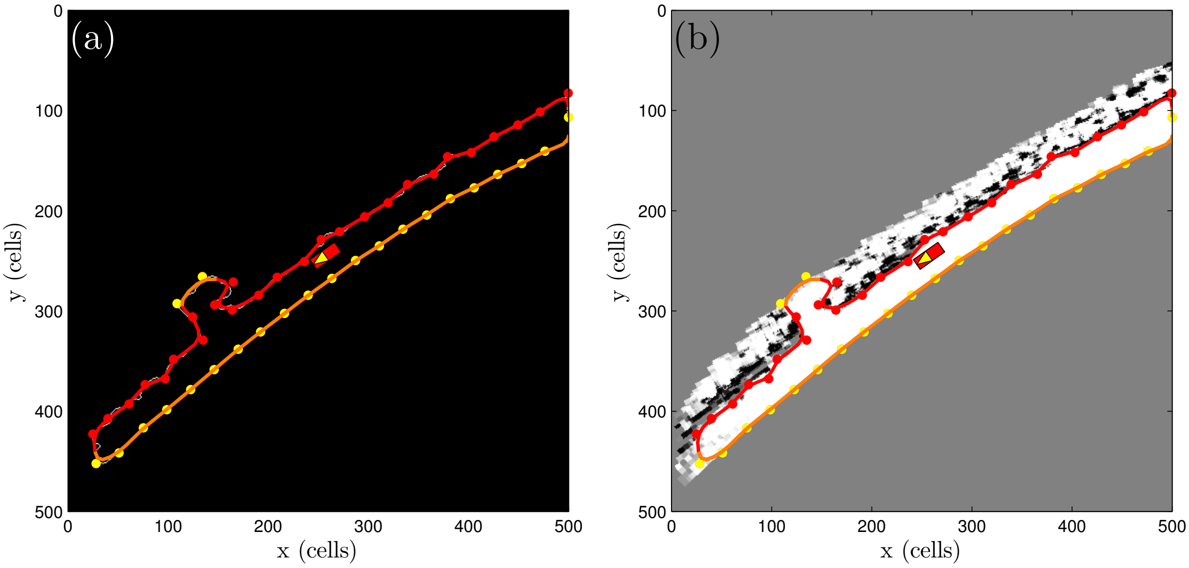


Figure 5.9: Free space tracking and boundary attribution result for the exemplary driving scene. The resulting PFS map with estimated B-spline curve and control points is shown overlaid over the extracted free space boundary cells in (a) and over the original grid in (b). Obstacle boundaries are highlighted in red, boundaries to unknown environment in orange.

5.3.4 Description of Inner Free Space Boundaries

The previous subsection explained the tracking of the outer free space boundary, which in most cases directly results in the final PFS map. In rare cases, however, the extracted relevant free space segment contains “free space holes”, which entail additional inner free space boundaries. These result if the sensors detect a significant amount of free space all-around an object. In principle, a B-spline-based description of such free space holes in analogy to the outer boundary description could be applied. However, objects that render such a mapping possible are normally small compared to the large outer free space boundary so that a representation in form of simple geometric primitives has proven to be sufficient. For this purpose, the shape of these free space holes is first classified based on roundness and rectangularity metrics [116].

The normalized roundness metric c_m is given by[12]

$$c_\mathrm{m} = \frac{4\pi A_\mathrm{h}}{(p_\mathrm{h}\kappa)^2}, \tag{5.17}$$

[12] There are various variants of this formula with or without the 4π factor or by changing nominator and denominator.

with area A_h, perimeter p_h, and Kulpa's perimeter correction factor κ [136], which approximately compensates discretization errors in digital images. This metric is equal (in digital images: approximately equal) to one for perfect circles and smaller for all other shapes. The correction factor is necessary to compensate for errors in the perimeter calculation in discretized images and is ideally set to $\kappa \approx 0.95$ [136]. The rectangularity metric r_m, in contrast, is calculated as the quotient of the area of the free space hole A_h and its minimal oriented bounding rectangle area $A_\mathrm{b,min}$ as

$$r_\mathrm{m} = \frac{A_\mathrm{h}}{A_\mathrm{b,min}} \tag{5.18}$$

according to [235].[13] This ratio is equal (in digital images: approximately equal) to one if the object is an ideal rectangle and smaller for all other shapes. The shape is classified as circular if $c_\mathrm{m} > r_\mathrm{m}$ holds, otherwise as rectangular. Depending on the classification result, bounding circles or oriented bounding rectangles are fitted subsequently, see Fig. 5.10 for an illustration.

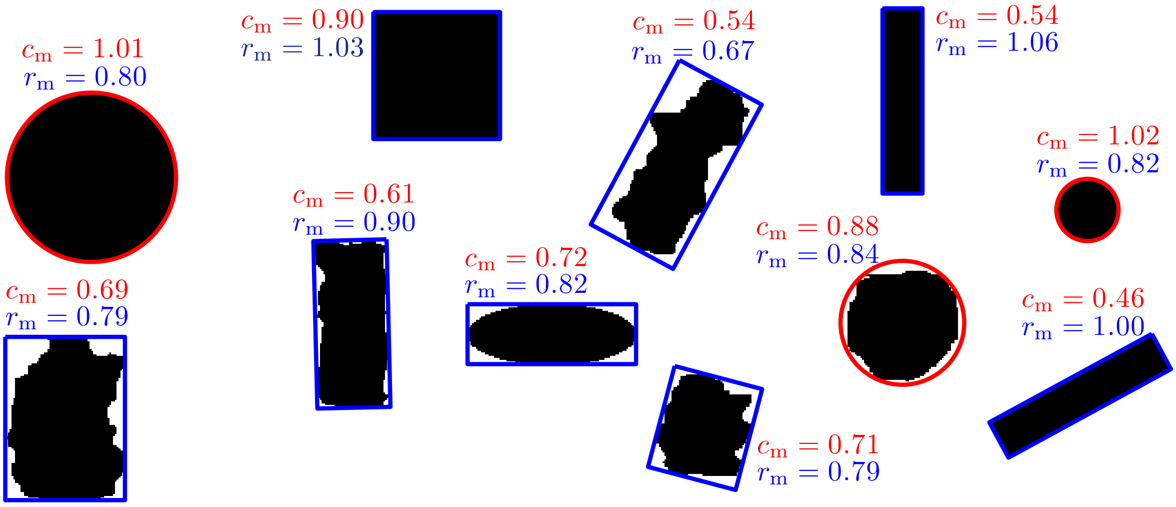


Figure 5.10: Shape classification based on roundness c_m and rectangularity r_m with corresponding geometric primitives in form of bounding circles (red) and oriented bounding rectangles (blue).

[13] There are other calculation methods for the rectangularity metric, see [209].

Considering N_c circles with state $\boldsymbol{x}_\mathrm{c} \in \mathbb{R}^{3N_\mathrm{c}}$ and N_r oriented rectangles with state $\boldsymbol{x}_\mathrm{r} \in \mathbb{R}^{5N_\mathrm{r}}$ as a result[14], then the state of the complete PFS map with B-spline control vector $\boldsymbol{x}_\mathrm{P} \in \mathbb{R}^{2N_\mathrm{b}}$ is finally given by

$$\boldsymbol{x}_{\mathrm{PFS},k} = \left(\boldsymbol{x}_\mathrm{P}^T \; \boldsymbol{x}_{\mathrm{c},1}^T \ldots \boldsymbol{x}_{\mathrm{c},N_\mathrm{c}}^T \; \boldsymbol{x}_{\mathrm{r},1}^T \ldots \boldsymbol{x}_{\mathrm{r},N_\mathrm{r}}^T\right)^T \in \mathbb{R}^{2N_\mathrm{b}+3N_\mathrm{c}+5N_\mathrm{r}}, \tag{5.19}$$

which constitutes the complete description of the static driving environment at time step k in form of relevant free space.

In conclusion, the section presented a computationally efficient method for the generation of PFS maps, which can be interpreted not only as a novel method of grid-based free space detection but also as a way of compactly representing and tracking arbitrary static driving environments. The method has been implemented on an experimental vehicle and tested in a variety of real traffic scenarios, which is described in the following.

5.4 Implementation and Performance

The proposed PFS map generation approach has been implemented both in MATLAB and in C++, and runs online on an experimental vehicle. The optimized C++ version has a combined calculation time of (13.7 ± 1.0) ms, measured on a 3.3 GHz Intel Core i5-2500K CPU, which is well below the sensor update time of approximately 65 ms. It can therefore run in combination with the previously described dynamic mapping approach without problems. Calculation times for the different steps of the algorithm are further summarized in Table 5.1.[15] The temporal stabilization via the B-spline tracking is, despite the advantageous IF-based estimation, computationally most demanding and consumes ≈ 53 % of the total PFS map processing time. This is unproblematic for the experimental vehicle equipped with mid-range desktop processors. If this turns out to be a limiting factor for embedded automotive hardware, then the free space

[14] The state of a circle is three-dimensional (center position coordinates and radius), the state of an oriented rectangle is five-dimensional (center position coordinates, orientation, length, and width).

[15] The performance analysis is based on the same 17 real traffic scenarios (2766 grid maps) that were used for the dynamic mapping evaluation described in Section 4.5. Note that total min/max times are not necessarily the sum of min/max times of individual processing steps as these extreme individual values do normally not occur at the same time. Also the total standard deviation is not the root of the sum of squares of the individual standard deviations due to the different means.

Table 5.1: Calculation times of the PFS map generation approach (mean, standard deviation, minimum, maximum) in ms. Median Filtering (M), Thresholding (TH), Erosion (E), Connected Components Labeling (C), Segment Selection (S), Dilation (D), Boundary Tracing (BT), B-Spline Tracking (TR), Attribution/Fitting (AF).

Quantity	M	TH	E	C	S	D	BT	TR	AF	Total
Mean	0.3	0.0	1.7	1.5	0.4	1.7	0.5	7.3	0.3	13.7
Std	0.0	0.0	0.2	0.2	0.1	0.2	0.1	0.4	0.0	1.0
Min	0.2	0.0	1.6	1.2	0.4	1.6	0.5	7.0	0.2	13.0
Max	0.6	0.0	4.3	3.8	1.3	4.3	1.2	13.7	0.7	25.6

tracking step can also be replaced by a least squares estimate of the B-spline control vector according to

$$\hat{\boldsymbol{x}}_{\mathrm{P},k} = (\boldsymbol{H}_{\mathrm{P},k}^T \boldsymbol{H}_{\mathrm{P},k})^{-1} \boldsymbol{H}_{\mathrm{P},k}^T \boldsymbol{z}_{\mathrm{P},k}, \tag{5.20}$$

with outer boundary measurements $\boldsymbol{z}_{\mathrm{P},k}$ and measurement matrix $\boldsymbol{H}_{\mathrm{P},k}$ given by (5.11) and (5.12), respectively. Temporal smoothness is, however, not ensured with this approach any more. A further run-time reduction can be achieved if only the free space in front of the ego vehicle, and not all-around, is calculated. This option has additionally been realized by rasterizing an artificial line of occupancy cells into the grid, which cuts the grid in half depending on the ego vehicle's orientation and performing all subsequent processing steps only for areas in front of the vehicle.

An implemented OpenGL visualization shows the PFS map overlaid over the corresponding occupancy grid map on an on-board screen, see Fig. 5.11a. The projection of the PFS map into the camera image, realized by a homography transformation, is additionally illustrated in Fig. 5.11b.

Bandwidth Requirements

As already stated in the introduction, compactness is an important property of any environment representation for near-future ADAS, in which the representation has to be transferred between many electronic control units with limited bandwidth. Since the number of spline control points is considered fixed, the required transmission bandwidth changes only marginally in the case of a varying number of inner free space boundaries. With 70 control points, which has proven to be sufficient for mod-

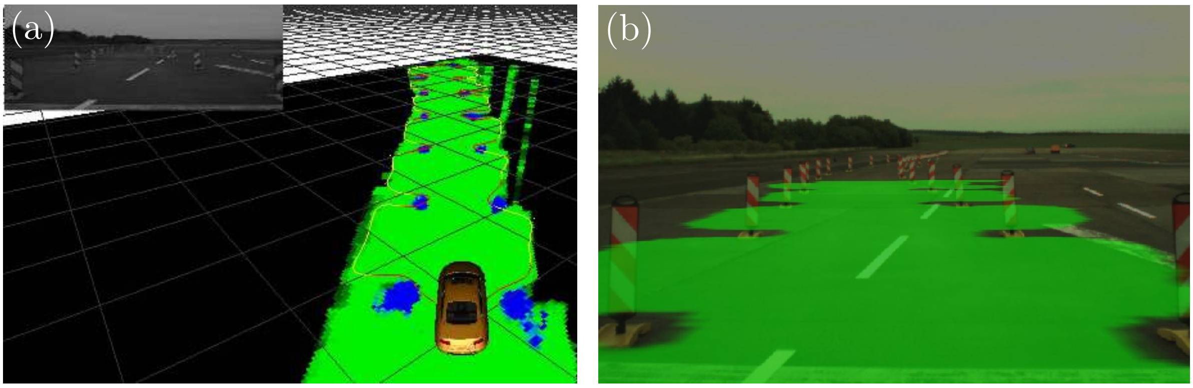


Figure 5.11: Online visualization of the PFS map overlaid over the occupancy grid map (a). Despite the complexity of the free space shape, the low-order B-Spline description with 70 de Boor points is sufficient to capture all the detail while smoothly bending around obstacles. The corresponding projection of the PFS map into the camera image is additionally shown in (b).

eling even very complex static environments around the vehicle, and a maximum of five geometric primitives as a worst case scenario, the required transmission bandwidth for the PFS map lies at

$$\frac{(70 \times 2 + 5 \times 4)16\text{bit} + 70 \times 1\text{bit}}{0.065\,\text{s}} \approx \frac{0.33\,\text{kB}}{0.065\,\text{s}} \approx 5.1\frac{\text{kB}}{\text{s}}, \tag{5.21}$$

considering 16 bit for each control point and geometric primitives coordinate, 1 bit for the boundary attribution, and a transmission once every 65 ms – the sensor update time. This is a factor of 12 less than the best (equally sized) lossless grid map compression results ($60\frac{\text{kB}}{\text{s}}$) and a factor of 2880 less than the best non-compressed results ($14686\frac{\text{kB}}{\text{s}}$) reported in [92]. In [91], the proposed PFS map representation has additionally been compared to the interval-based free space description and the stixel world and found to be by a factor of 23 smaller than the former and still by factor 2.5 smaller than the latter. The PFS map is therefore easily transferable even over automotive CAN. In the next section, PFS maps are shown for several more complex driving scenarios.

5.5 Experimental Results

To underline the generality of the PFS map representation, the results in different real-world driving scenarios of varying complexity are illustrated in Fig. 5.12. All maps have been generated with the same parameter set

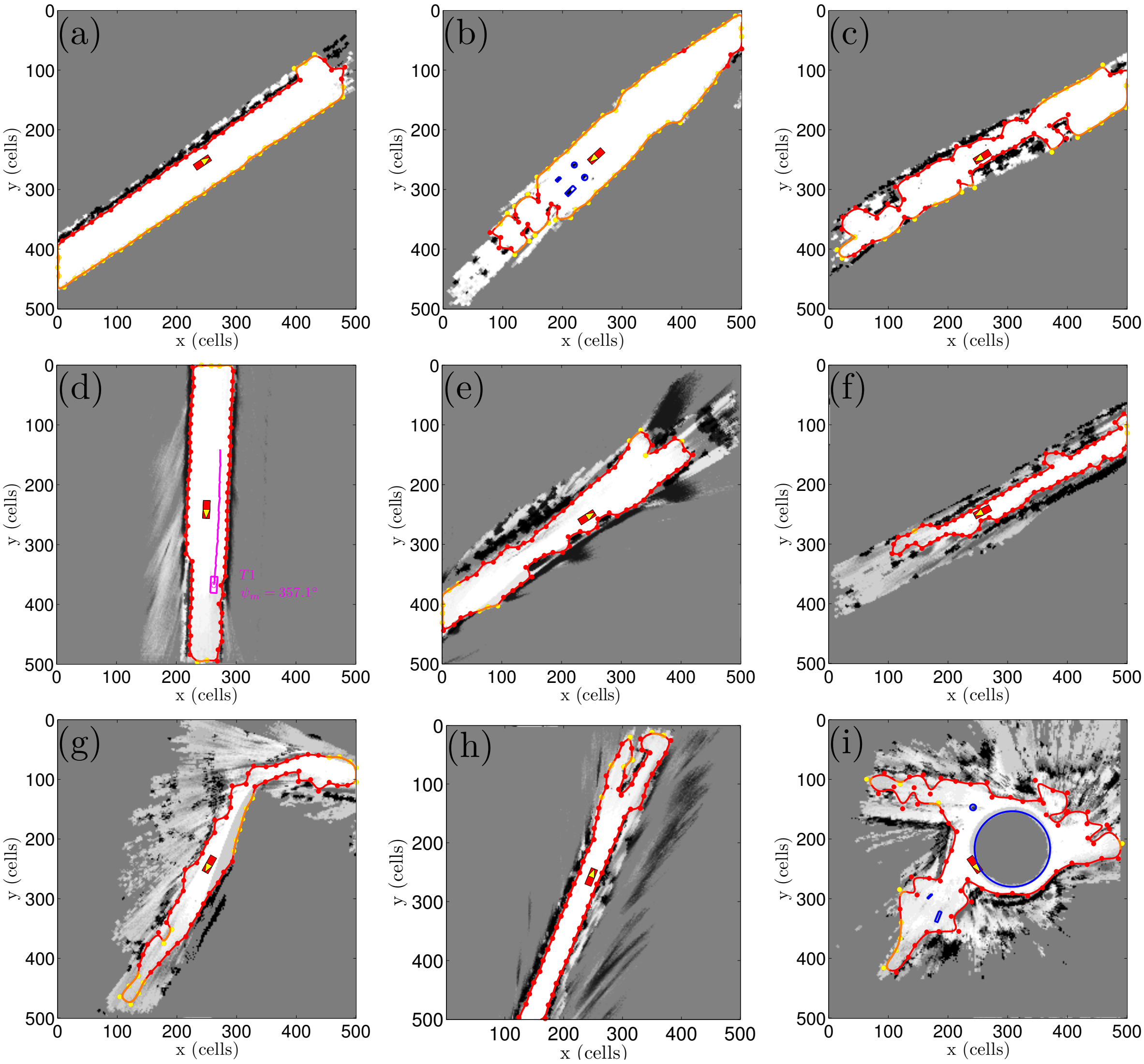


Figure 5.12: PFS maps (colored) overlayed over corresponding occupancy grid maps (grayscale) in different real-world driving scenarios.

summarized in Table 5.2. The PFS maps are overlaid over the corresponding occupancy grid maps that were used in the generation process. The first three grids were created solely by radar, whereas the others were mapped by a combination of radar and stereo camera to show the unproblematic extension to additional environment sensors.

Fig. 5.12a shows a country road with an obstacle boundary and a turnoff on the left side of the ego vehicle, which are both clearly visible in the PFS map representation. The free space boundary on the right side holds the attribute of an "unknown environment boundary" as visualized by yellow spline control points and corresponding orange spline curve segments. The reason for this is that no real free space evidence is available behind this

Table 5.2: PFS map generation parameters.

Parameter	Value
Grid size; Cell length c	80 m x 80 m; 0.16 m
$S_\mathrm{M}; m_\mathrm{th}; r_\mathrm{SE}; N_\mathrm{b}; l_\mathrm{n}$	3 cells x 3 cells; 0.7; 1.1 m; 70; 2 m
$\hat{\boldsymbol{i}}_{\mathrm{P},0}; \boldsymbol{I}_{\mathrm{P},0}$	$\mathbf{0} \in \mathbb{R}^{2N_\mathrm{b}}$; $\mathbf{0} \in \mathbb{R}^{2N_\mathrm{b} \times 2N_\mathrm{b}}$
$\boldsymbol{Q}_\mathrm{P}$	$(\frac{0.8}{c}\,\mathrm{m/s})^2 T^2 \cdot \mathbb{1} \in \mathbb{R}^{2N_\mathrm{b} \times 2N_\mathrm{b}}$
$\boldsymbol{R}_\mathrm{P}$	$(\frac{0.2}{c}\,\mathrm{m})^2 \cdot \mathbb{1} \in \mathbb{R}^{2N_\mathrm{p} \times 2N_\mathrm{p}}$
B-spline type	uniform, quadratic ($d = 3$), periodic

boundary because the area has not yet been observed by the radar sensor with its small opening angle of only 17°. In the construction site scenario shown in Fig. 5.12b, with delineator posts on the left and right, four geometric primitives are present in addition to the B-spline contour. Two of them are classified as round and two as rectangular. These objects are not included in the outer free space boundary because enough free space was detected, so that the ego vehicle could drive around them. This is in contrast to the obstacles further ahead. The PFS map in Fig. 5.12c shows an inner city scenario with parking cars as well as house fronts. Despite the complexity of the scene, the flexible spline curve can describe all relevant grid map information in a very compact way. Fig. 5.12d represents the PFS map for the exemplary scene of Chapter 4 with the dynamic object map additionally overlaid in magenta. It is obvious that static and dynamic entities are well-separated, and dynamic object corruptions do no longer influence the B-spline curve and PFS map, respectively. Without the previously described dynamic mapping approach, the PFS map's curve would bend around the dynamic object corruptions and might falsely trigger driver warnings or interventions as a consequence. Additionally, higher-level situation recognition algorithms that analyze the shape of the free space, e.g. for detecting junctions [189], would be negatively influenced, which highlights the importance of a dynamic object corruption-free grid. Fig. 5.12(e,f) show additional inner city scenes similar to Fig. 5.12c but mapped by a combination of radar and a noisier stereo camera. Here, the PFS map generation process depends more strongly on the threshold level, but reasonable results can nevertheless be obtained by using the same parameter set as previously on the cleaner radar grids. In Fig. 5.12e, for example, three parking cars are present on the right of the ego vehicle and the street turns to the left, as visible in the unknown environment boundary control points at the top. Fig. 5.12g shows the ego vehicle after a left turn, and after mapping many irrelevant regions on the upper left side of the grid that cannot be reached. This demonstrates that the morphological processing steps effectively remove these areas and the free space boundary only follows the relevant, reachable course of the road. Fig. 5.12h, in contrast, depicts a situation in which a traffic isle separates the road, and therefore also the free space in front of the ego vehicle. The PFS map can represent such situations in contrast to existing parametric, grid-based road course estimators. Lastly, a roundabout scenario is shown in Fig. 5.12i. The ego vehicle has already completed a full round. Again, all irrelevant grid map parts are suppressed in the PFS map representation, which is supposed to facilitate higher-level environment information recognition. Note, however, that the inner circle in this example does not represent the elevated middle of the roundabout but rather the limited sensor field of view.

In summary, the scenarios showed that the PFS map representation is general enough to represent a large variety of different driving environments and that it effectively suppresses most of the irrelevant information contained in common occupancy grids while keeping the relevant information in a much more compact way.

5.6 Conclusion

A novel, generic representation of arbitrary static driving environments in form of so-called PFS maps has been introduced, which describe available free space in a parametric, continuous way. An algorithm has been presented that allows to generate PFS maps in a computationally efficient way by first extracting relevant, arbitrarily shaped free space areas by methods known from the image analysis domain applied to optimized occupancy grid maps. A subsequent free space tracking step provides additional robustness and allows to incorporate spatio-temporal smoothness constraints. The PFS map generation process can thus also be interpreted as a novel method of grid-based free space detection and estimation for intelligent vehicles, which

- performs image analysis on grid maps for free space detection,
- provides robustness due to temporal filtering via free space tracking,
- includes free space not lying directly in the line of sight,

- generates connected, reachable free space areas,
- represents arbitrary free space shapes in a compact, parametric way,
- offers easy to parametrize spatio-temporal smoothness, and
- is decoupled from the sensor principle due to the intermediate grid.

The advantageous properties of the resulting PFS maps are

- the compactness ($\approx$ 0.33 kB), suitable even for automotive CAN transmission and with guaranteed upper limits on the required transmission bandwidth, which is relevant for near-future ADAS in which the representation has to be transferred between many electronic control units,
- the generality to represent a large variety of driving environments as shown in the experimental results section,
- the suppression of irrelevant free space information, which is beneficial not only for trajectory planners that just need to consider reachable areas but also for higher-level scene classification algorithms that do not have to cope with irrelevant details any more,
- the explicit representation of free space, which is vital for evasive trajectory planning as it makes a difference whether an area is really free or has just not yet been observed,
- the continuous, and not discrete, representation, which allows arbitrarily fine evaluations and closed, analytic solutions to problems such as distance measurements to closest boundaries, which simplifies tasks like online creation of potential fields [24],
- the independence of the representation from the ego vehicle orientation, which is an important aspect for scene interpretation algorithms,
- the sensor principle-independent, computationally inexpensive, robust generation with common automotive sensors, and
- the fact that PFS maps can be extracted from common occupancy grid-based representations, which are already in use in many state-of-the-art intelligent vehicles.

On the downside, the PFS map generation process is computationally slightly more demanding than pure grid mapping and the maps cannot directly be used by trajectory planners relying on cells. For planners of this kind, the PFS map can, however, easily be rasterized on a grid again. Another aspect is the loss of information about specific cell occupancy probabilities. As grid mapping algorithms are normally designed to converge rapidly to high free or occupied probabilities [247], this fact is, however, often negligible in practice. If this turns out to be a limitation in a specific system, then the proposed algorithm could just be applied with different thresholds on each grid map in parallel to capture this information. Moreover, the PFS map is a 2D representation and does consequently not encode height information. There might be some future ADAS functions for which such a 2D representation will not be sufficient, but – in the eyes of the author – the advantages of compactness and interpretability not given in more general, higher-dimensional representations outweigh this limitation.[16]

For a complete metric representation, the PFS map of the static environment is supposed to be combined with a dynamic object map as already pointed out in Section 2.3. This separation between static and dynamic structures simplifies subsequent situation interpretation algorithms that need to treat static and dynamic object in a different way. Note that the consistency between both maps is always ensured by the proposed approach because both maps are extracted from the same data base, i.e. a common occupancy grid map. The object abstraction of dynamic entities additionally allows model-based predictions based on the object type, which is vital for adequate trajectory prediction and driving maneuver detection. The complete metric representation including both maps is extracted in just 23.8 ms (3.3 GHz Intel Core i5-2500K) on an experimental vehicle and has been demonstrated in real traffic environments. By further combining it with information about lanes, traffic signs, topological maps, etc., a complete world model can easily be formed.

The representation has already been successfully employed

- for the detection of higher-level, semantic environment information such as junctions and roundabouts in [189],
- for criticality assessment in arbitrary driving environments [222,224], see Chapter 6, and

[16] Often, a simple reinterpretation of drivable free space is sufficient to handle some problems involving height knowledge. For example, a tunnel too low to pass safely could be represented as a 2D free space boundary.

- for potential field-based safety trajectory planning and lateral vehicle guidance with respect to elevated objects in PRORETA 3 [25,56,219, 220,274,275], see Chapter 7.

In conclusion, the proposed metric environment representation describes the *relevant* information contained in conventional occupancy grid maps in a *consistent*, *compact*, *easily interpretable*, *continuous* way and can be generated in a *robust*, *computationally inexpensive*, *sensor-independent* manner. Consequently, the first two main questions raised in the thesis introduction, i.e. how the environment can be represented and how this representation can be obtained, are answered. At this stage, the reader is invited to consult Table 2.1, p. 13 once again for a comparison with other representations.

The following chapter, in addition, presents a novel, integrated approach to probabilistic, long-term trajectory prediction and criticality assessment, which uses the proposed environment representation as a foundation.

6 Prediction and Criticality Assessment

The present chapter presents a novel system for trajectory prediction and criticality assessment. After the introduction, related work on the subtopics short-term trajectory prediction, long-term trajectory prediction, situation recognition and prediction, and criticality assessment is discussed. The system itself is composed of a maneuver detection part, a maneuver-based trajectory prediction part, and a criticality assessment part – all three presented subsequently. Simulation results underline the effectiveness and scope of the approach. The chapter concludes with a summary of the system's key aspects and possible future work.

6.1 Introduction and Motivation

So far, we have seen how to estimate and represent the *current* state of the environment. This representation forms the first level of *Situation Awareness* (SA) [78]. Future ADAS and intelligent vehicles must, however, not just react to this current state but understand the current situation (SA level 2) to be able to anticipate the future evolution of the traffic scene (SA level 3). This is a necessity for correct decisions, warnings, and interventions. Whereas the evolution can reasonably be predicted for a short time interval by just considering physical quantities such as estimated vehicle velocities or yaw rates, the evolution over several seconds is much stronger influenced by the intentions, motivations, and goals of all traffic participants within the specific driving environment. Consequently, three main sources of uncertainty can be identified:

- Uncertainty in state estimates.
- Uncertainty in driver intentions.

- Uncertainty in driving maneuver[1] executions for the realization of the intention.

This raises the complexity for longer-term predictions considerably, but only by taking these factors into account, it becomes possible to warn and intervene early enough. In this context, two major challenges exist in the design of long-term trajectory prediction and criticality assessment algorithms for active collision avoidance and warning systems.

First, it is neither optimal to determine just a single future trajectory for each vehicle nor is it reasonable to predict every physically possible trajectory. In the first case, the one and only future hypothesis will most likely not occur, and any subsequent decision making system relying on this single hypothesis would fail if the traffic scene evolved differently. An example is given in Fig. 6.1, in which both vehicles can turn, stay on their lane, or perform a lane change. Human drivers take these different scene evolutions into account, an automatic system should do that as well. In the second case of considering every possible trajectory, false warnings will be generated. This becomes obvious in the exemplary situation visualized in Fig. 6.2, which shows the reachable sets of two oncoming vehicles on different lanes. The sets overlap after a very short prediction horizon although this standard situation is uncritical and no warnings should thus be triggered even for greater prediction times. Therefore, the first challenge is to strike a balance between the two mentioned extremes and to limit the possible future scene evolutions to reasonable subsets given all available environment information.

The second, less considered challenge is that the further one tries to predict into the future, the more assumptions have to be made, which tempts to model the average, sensible driver in a given traffic situation. As an extreme example, the prediction could be based upon the assumption that every driver obeys the traffic rules or follows the road flawlessly. Although these assumptions are reasonable for microscopic traffic simulations, they are not appropriate for active safety systems because especially actions that do not follow traffic rules and contradict with the standard situation evolution might become dangerous. Thus, all prediction methods that are solely based upon the average driver are not suitable for an

[1] In this chapter, the term driving maneuver is used as in [181] and describes actions such as *road followings*, or *turns*. This is the common interpretation in the intelligent vehicles domain. It should not be confused with the term maneuvering target as it is used in the target tracking community and in Chapter 3 and 4. There, it characterizes all motion patterns that stand in contrast to benign, constant velocity motion.

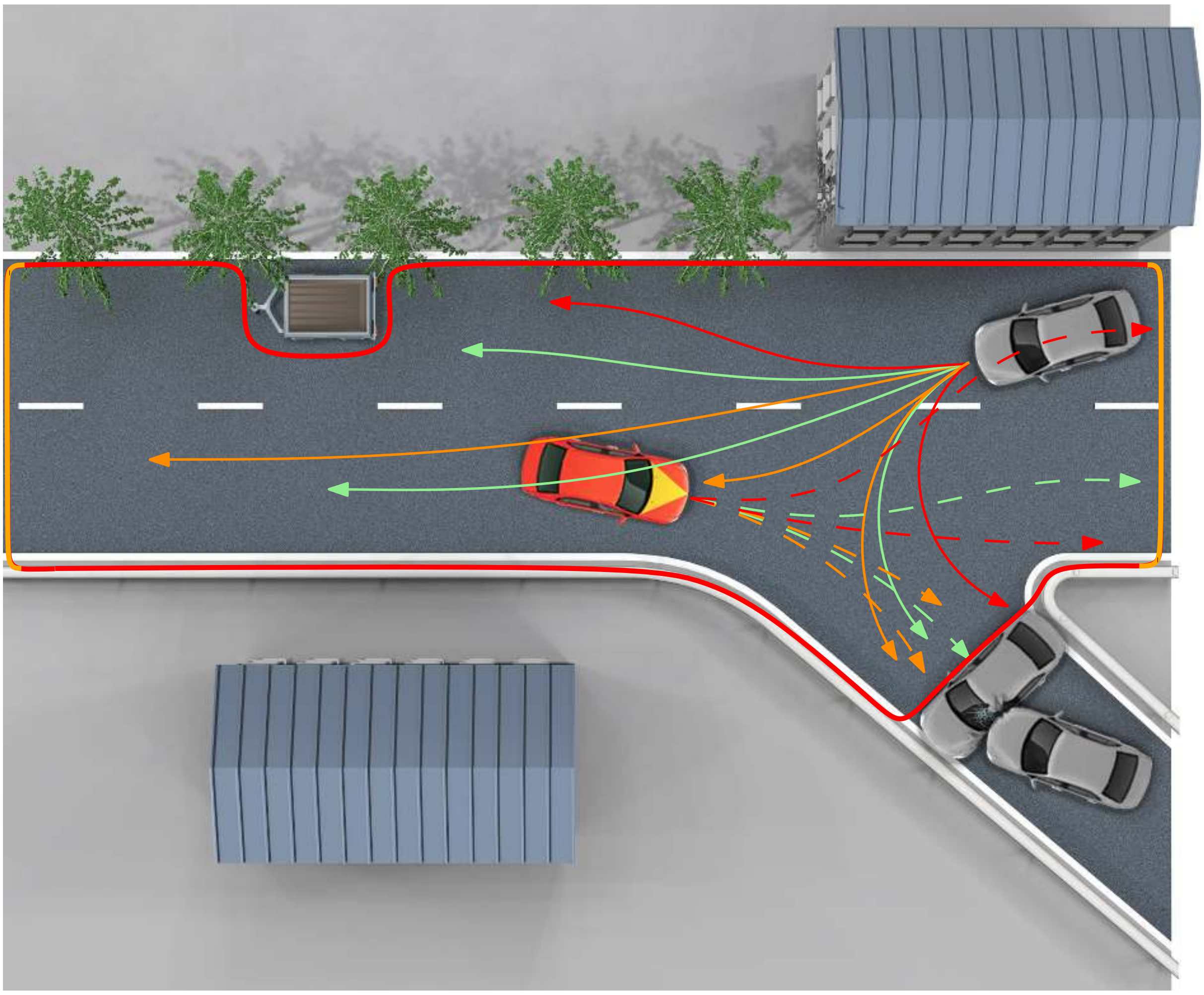


Figure 6.1: Possible future scene evolutions in an exemplary driving scene. Both vehicles can turn, stay on their lane, or perform a lane change. Trajectories of possible future driving maneuver executions are shown in color – more probable ones in green, less probable ones in orange and red, respectively.

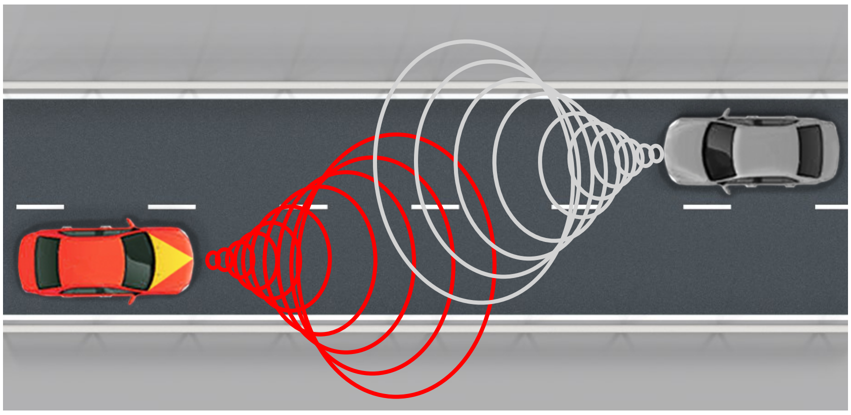


Figure 6.2: Overlapping reachable sets of two oncoming vehicles based on [4].

emergency situation ADAS as exactly these situations are excluded by the prediction assumptions beforehand and therefore cannot be predicted at all. Consequently, a system for long-term criticality assessment should

- be sensitive to exceptional, rarely happening situations,
- not only consider physical quantities but also information about the drivers' intentions and the driving environment,
- allow the inclusion of the different sources of uncertainty, and
- take into account a reasonable subset of possible future scene evolutions.

Before the proposed approach to this problem is presented, an overview of related work on prediction and criticality assessment in traffic environments is first given in the following.[2]

6.2 Related Work

Related work on prediction in traffic environments can be subdivided into short-term trajectory prediction, long-term trajectory prediction, and the more abstract situation prediction. Under short-term trajectory prediction, we understand all methods that are solely based upon physical vehicle[3] states but not upon the traffic environment or driver intentions. Long-term trajectory prediction methods, in contrast, consider these additional factors, whereas situation prediction methods allow the situation recognition and prediction on a higher abstraction level but do not directly aim at the determination of future spatio-temporal trajectories. The main distinguishing characteristics of these three prediction variants are contrasted in Table 6.1 and described in detail together with the related criticality assessment in the following subsections.

[2] See also the surveys [149, 233]. The former contains a section about vision-based approaches of maneuver detection, the latter covers the topic by explicitly separating long-term prediction models into models considering interactions and ones that do not.

[3] We limit the discussion to vehicle predictions. Other traffic participants such as cyclists or pedestrians are omitted as their predictability is much lower and can normally only be performed via short-term methods.

Table 6.1: Differences between short-term trajectory prediction, long-term trajectory prediction, and situation prediction.

Prediction type	Short-term trajectory prediction	Long-term trajectory prediction	Situation recognition and prediction
Future motion cause	System dynamics	Human intention in traffic environment	Human intention in traffic environment
Prediction horizon	ca. $< 1\,\mathrm{s}$	Several seconds	Several seconds
Required environment knowledge	Low	High	High
Abstraction level	Low	Low – Medium	High
Result	Quantitative	Quantitative	Qualitative
System description	Kinematic and dynamic motion models	• Motion pattern databases • Dynamic motion models with behavior descriptions	• State machines • Fuzzy theory • Bayesian networks • Dempster-Shafer theory • Logics
Assumptions	Constant or nearly constant motion state	• Motion follows pattern • Behavior-dependent motion	Diverse
Prediction methodology	• Num. integration • Tracking filter-based prediction	• Pattern recognition • Cost function minimization • Trajectory planning	• Inference • Classification

6.2.1 Short-Term Trajectory Prediction

Within short prediction horizons, the influences of the driver and the environment are minor and the prediction predominantly depends on the driving physics and system dynamics. The underlying assumption of methods of this kind is that future system inputs are equal (or approximately equal) to the current system inputs and the motion model does not change in the future. For the prediction itself, two methods are common:

- Numerical integration of a supposed motion model, in which the initial values are derived from estimated, current states of a preceding tracking filter [50].
- Direct evaluation of the prediction equations of the tracking filter without measurement updates [20,22,163,200].

The latter additionally allows the direct propagation of state uncertainties.[4] In both cases, the crucial factor of short-term prediction methods is the quality of the underlying tracking filter. The prediction models directly correspond to the system dynamics equation of the filter. The models *Constant Velocity* (CV), *Constant Acceleration* (CA), *Constant Turn Rate and Velocity* (CTRV) and *Constant Turn Rate and Acceleration* (CTRA) are prevalent for this purpose [227] and are partly combined within an IMM Filter not only for filtering but also for prediction [20]. More complex models such as single-track or two-track models are not common within the tracking filter as an adequate estimation of the additional states and parameters becomes difficult within the usually short observation time or even impossible due to their unobservability for exteroceptive sensors.

Short-term predictions of the ego vehicle[5] follow the same methods as employed for the other vehicles with the difference that more complex system models such as a single-track model can be used due to the additional available measurement quantities and parameters. This complexity

[4] Care has to be taken in the interpretation of so-calculated uncertainties. For a quantitatively correct uncertainty prediction, the stochastic system dynamic and measurement equations have to match reality exactly. Often, the process and measurement noise terms are, however, used as filter tuning parameters without particular justification. Normally, a larger than correct process noise is artificially induced in the system dynamics equations to capture unmodeled effects of the employed simple motion models. As a consequence, the predicted uncertainty rises unnaturally fast in this approach.

[5] In the case of a fully autonomous ego vehicle, the future path of the ego vehicle is planned and known. This does not hold for ADAS, in which the vehicle is still driven by a human driver, which makes the problem more complicated.

increase is, however, only advantageous up to a certain point as already stated in [227].

Conclusion

Independent of the physical model complexity, short-term prediction models are only reasonable for prediction horizons of typically less than a second [149]. For longer time horizons, the constant input assumption cannot be kept and methods that consider driver behavior in the driving environment context have to be employed. This is the topic of the next subsection.

6.2.2 Long-Term Trajectory Prediction

An obvious variant of extending the scope of the mentioned short-term trajectory models is to include knowledge about driving lanes or road boundaries as realized in [162,192,200]. These methods allow greater prediction times but do not permit the inclusion of additional assumptions about the actions of different traffic participants. Long-term trajectory prediction methods that additionally consider driver behavior can be subdivided into the following two categories:

- Methods of pattern recognition in motion pattern databases [7,99,124,125,255,256].
- Methods that fuse dynamic motion models with behavior and environment descriptions [5,45,75,82,88,144,211,236].

Methods of the first category cluster previously observed trajectories and generally represent each cluster by a trajectory prototype [7]. The prediction task involves finding the most likely cluster (or a distribution over clusters) that fits to a newly observed partial trajectory and the remainder of the trajectory is used for prediction. Although the behavior is not explicitly modeled, it is implicitly included in the learned trajectory prototypes. Representatives of this kind of prediction can be found in [98, 99, 125], in which trajectory clusters of vehicles at intersections are built and a probabilistic online matching of observed object states to saved trajectories is performed on basis of a quaternion-based rotationally invariant longest common subsequence metric via particle filtering. The approach is further developed in [270] to predict a distribution over future trajectories given an observed part of a learned motion pattern via

Gaussian mixture models. In [256], motion patterns based on current and intended goal states are learned incrementally and in parallel to prediction with the help of so-called growing HMMs. The system is used for trajectory prediction in parking lots (Fig. 6.3a), but may be hard to generalize to arbitrary driving environments due to unclear goal states, e.g. in vehicle following scenarios.

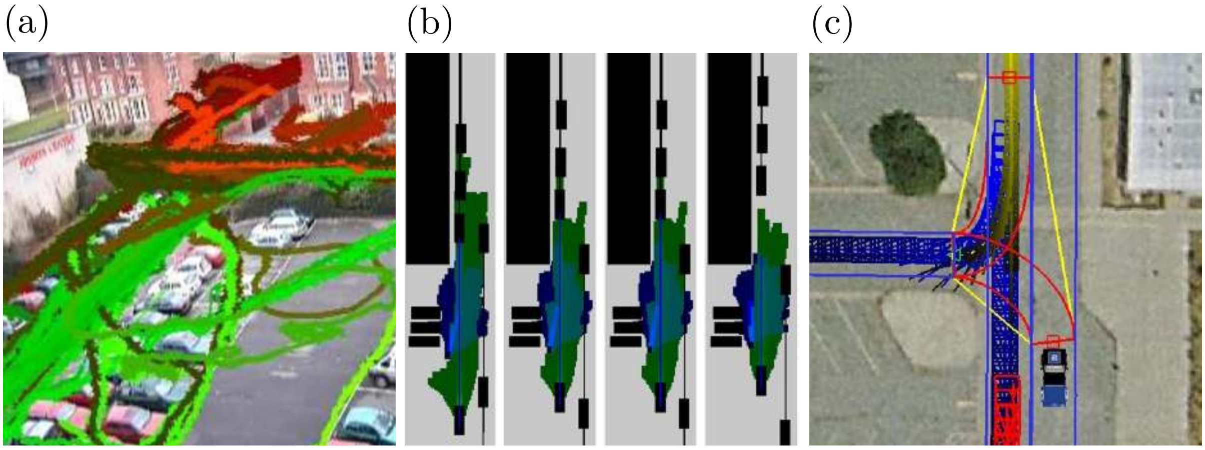


Figure 6.3: Examples of long-term trajectory prediction methods. Motion pattern-based [256] (a), via path planning [45] (b), map-based [82] (c). Images taken from the mentioned sources.

Besides the problem of generating, saving, and fitting large motion pattern data records, methods of the first category have the disadvantage that only trajectories included in the database can be predicted, see also [254]. This limits their applicability to criticality assessment as they cannot predict abnormal situations, which is why they are not regarded further. Some representatives of the second category that additionally consider uncertainty are explained in the following.

The approaches of [45, 75] perform probabilistic trajectory predictions by employing path-planning algorithms from the viewpoint of each traffic participant to generate distributions over future motions of all vehicles. Different combinations of future system inputs are considered via Monte Carlo simulations, in which the stochastic inputs are restricted to specific, typical human driving behaviors and actions such as *lane changes* or *overtaking*. System inputs are the result of goal functions that model behaviors. No explicit maneuver detections are performed. The result is a probability of collision for the complete traffic scene as well as a so-called danger level display for each road position at each future time (Fig. 6.3b).

In [5], a similar approach is presented, which is based on the determination of stochastic reachable sets of all other, interacting traffic participants under the assumption that these follow specific paths along a known road network with a certain accuracy. The ego vehicle's future path is considered available as the system is designed for autonomous vehicles with a known planned future trajectory and not for ADAS. The longitudinal dynamic motion models of other vehicles along their paths are abstracted into discrete Markov chains, the lateral positions are described by predefined, fixed distributions. The future actions *go straight* and *turn* are considered for the other vehicles. As their motions are constrained along road geometries and speed limits, an unintentional leaving of the road, however, cannot be detected.

In [82], vehicles are supposed to follow a known road network as well (Fig. 6.3c). At intersections, simply all possible lanes leaving the intersection are used for the predicted vehicle location simultaneously with equal weight. This results in a very conservative prediction, which might be a reasonable approach for the first generation of defensively behaving autonomous vehicles but not for the task of long-term criticality assessment for ADAS due to the resulting high number of false positive warnings.

The approach of [144] allows longer-term criticality assessment in structured highway environments and focuses on the interaction of traffic participants. Predictions of a considered highway scenario are performed via a path-planning algorithm that generates minimum jerk trajectories. By updating a hand-designed prior intention distribution with results of the corresponding, fictive collision probabilities stemming from the execution of each intention, so-called interaction-aware maneuver probabilities are estimated. They are based on the postulation that drivers do not perform maneuvers with high collision risks as long as safer options are possible. This assumption, however, inevitably prevents the detection of specific dangerous situations such as colliding with a slower vehicle on the same lane if there is still a free adjacent lane for a possible lane change.

Within the so-called PRIDE (Prediction In Dynamic Environments) framework introduced in [211], estimation-theoretic short-term predictions are coupled with situation-based long-term prediction models. For the latter, cost functions are employed that describe the danger a driver would be exposed to if he executed a specific action sequence. A probability is associated with each predicted trajectory corresponding to an action. In a next step, each detected collision of a trajectory is used to adapt these probabilities retrospectively by assigning high costs to dangerous predictions. This procedure, however, excludes the prediction of critical trajectories due to their high costs, which makes it inappropriate for safety warning and intervention systems.

The approach described in [236] also employs behavior cost functions from which, after minimization, the future driver inputs of a decoupled two-dimensional CA model are derived. A combination of longitudinal, lateral, interaction, and comfort cost functions is used. Deviations from a reference velocity, driving too close to the vehicle in front, or abrupt braking and steering maneuvers are punished in the optimization process. The result is a model for an average driver but cannot be used in this form for long-term criticality assessment.

Conclusion

In conclusion, all mentioned methods provide a quantitative trajectory prediction for longer prediction horizons valid for several seconds. They are designed to give better prediction results in *standard* traffic situations than the previously described short-term methods. They are, however, often not designed with the goal of long-term criticality assessment in mind due to their underlying assumption of sensible drivers. Therefore, they do not cope with the second challenge mentioned in the introduction, i.e. the sensitivity to exceptional, rarely happening situations. Moreover, they lack a more abstract, symbolic, qualitative description of the traffic situation. Methods of this kind that can, in particular, predict a situation on a more abstract level, are treated in the next subsection under the term situation recognition and prediction.

6.2.3 Situation Recognition and Prediction

Besides explicit trajectory prediction methods, numerous approaches reason about the future traffic scene on a more abstract level by recognizing and predicting situations. These approaches are mainly not based on low-level dynamic state space models any more but employ a variety of mathematical tools for description, inference, and classification:

- State machines [102, 185, 215, 239]
- Fuzzy theory [102, 216]
- Static BNs [122, 128, 147, 198, 216, 228]
- DBNs [3, 58, 88]
  - HMMs [28, 142, 169]
  - Jump Markov Models [268]

- Dempster-Shafer theory [185, 191, 250]
- Special kinds of logics [97, 103, 111]
- Various classifiers [39, 95, 125]

In the following, representatives of the different groups are explained to make the diversity of the methods graspable. Methods trying to infer driver intentions without any physical vehicle motion indication, e.g. by observing driver-eye-gaze behavior [176], are not considered as they cannot be realized without vehicle-to-vehicle communication.

In [185], a rule-based system is presented to determine current driving maneuvers that consist of several distinct states linked by transition conditions. If a condition is fulfilled, the system reaches the subsequent state, which makes it a deterministic state machine. A maneuver is recognized as soon as the last transition condition is fulfilled. Thus, the information is only available by the time the maneuver is nearly finished, which limits its applicability for predictions. A deterministic maneuver detection is further presented in [239] and coupled with a trajectory prediction system. Driving maneuvers are separated into longitudinal and lateral maneuvers. The detection of the latter is based on the cross correlation between yaw rate and steering wheel angle, which makes the approach only applicable for the ego vehicle. Transitions between different lane change phases and turns are additionally linked in a deterministic state machine for a more robust detection. Another state machine-based system tailored towards the detection of overtaking situations can be found in [215] (Fig. 6.4a). An overtaking is split into the phases *pull out*, *passing*, and *cut-in*. A longitudinal indicator variable is formed via Fuzzy combination of *Time-To-Collision* (TTC) and accelerator pedal position, and is combined with the lateral road position.

In [102], complex driving maneuvers are modeled as a sequence of basic maneuver elements in a probabilistic finite state machine. The basic elements are specified via a Fuzzy rule base (Fig. 6.4b), and a Bayesian filter is used to calculate the probability of the basic elements in the context of the maneuver model to recognize the current driving maneuver. The approach is shown to be able to detect *turn* maneuvers. Another combination of a Fuzzy and a Bayesian approach can be found in [216]. Here, the values of maneuver- and situation-specific features are first transformed with the aid of situation-dependent membership functions. Then, the feature membership levels are used as evidences in a static *Bayesian Network*

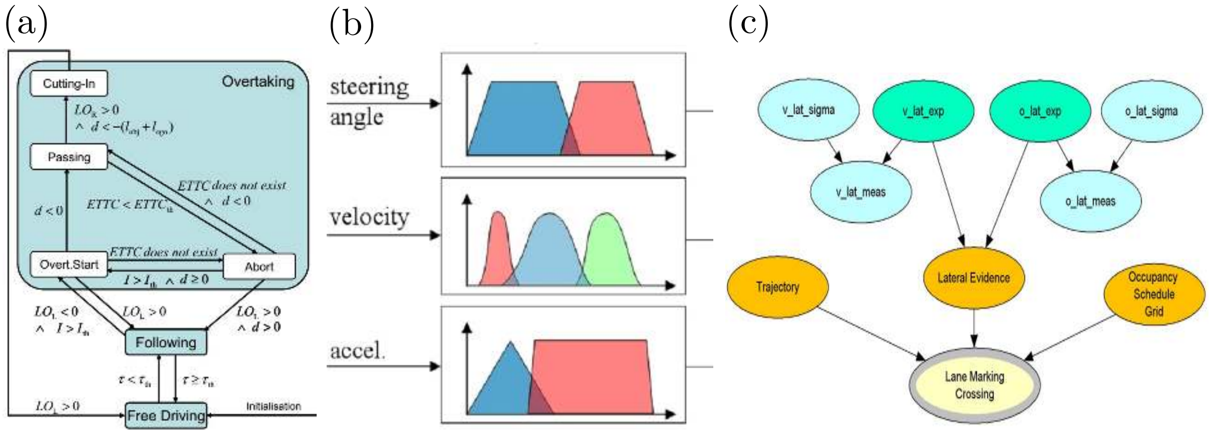


Figure 6.4: Examples of situation recognition and prediction methods. State machine-based [215] (a), via Fuzzy theory [102] (b), Bayesian network-based [122]. Images taken from the mentioned sources.

(BN), which is employed for situation inference. Additionally, feature uncertainties are included in the network to determine inaccuracies of the resulting situation statements.

Examples of purely Bayesian approaches include the object-oriented static BNs employed in [122] for the early detection of *lane changes* (Fig. 6.4c). In [228], a similar method is used to suggest a lane change to the driver under additional consideration of the tracking filter's estimated state uncertainties during situation recognition. In contrast to manually parametrized networks, situation and behavior models can also be learned from data as shown in [198] for intersection scenarios. An early general Bayesian approach can further be found in [58], in which high-level goals and motivations are modeled in a *Dynamic Bayesian Network* (DBN). This thought is taken up in [88] to estimate not only intentions but also the resulting trajectories simultaneously, both modeled in a single DBN. A particle filter is used for inference. It is one of the few publications that connects the topics situation and trajectory prediction. The parametrization of this very general model seems, however, to be rather cumbersome. Additionally, a policy model favors trajectories not resulting in collisions, which might limit the applicability of the approach for long-term criticality assessment.

Besides general DBNs, the subclass of discrete HMMs is prevalent in situation and maneuver detection systems. Individual HMMs are used in [169] for the modeling of the driving situations *vehicle passing*, *aborted passing*, and *vehicle following*. Parameters are learned with the Baum-Welch algorithm, and the situation prediction is carried out via the prediction step of the learned HMM without measurement update – similar to the previously mentioned short-term trajectory prediction method via open-loop tracking filters. Although this kind of prediction is robust against state estimation uncertainties, the authors of [169] point out that the prediction quality strongly depends on the parametrization and state discretization and can give inaccurate prediction results. Similarly, HMMs are employed in [28] for the detection of *lane following*, *lane change*, and *turn* maneuvers. At first, parameters corresponding to the complete maneuver are learned via offline-labeled maneuvers. The complete Markov chain is then decomposed and submodels are extracted. Submodels corresponding to early maneuver stages are matched online, which allows an early maneuver recognition. HMMs are also employed for the detection of *turns*, *overtaking*, and *straight motion* in [142] in a hierarchical way. For each high-level behavior in the upper layer HMM, a corresponding lower layer HMM represents the sequence of finer state transitions of a single behavior. The inferred behaviors are then used for trajectory prediction via Gaussian processes and criticality assessment. Thus, it is one of the few approaches that combines these different research areas, which are most often studied in isolation.

Besides discrete HMMs, also linear jump Markov models, as another subclass of general DBNs, are employed. In [268], *lane change* maneuvers are detected by considering a process model that corresponds to approximately longitudinal motion and another one that models stronger lateral motion prevalent during a lane change maneuver. An IMM Filter infers the mode probabilities, which serve as the basis for the maneuver classification between the classes *lane change* and *lane keeping*.

Besides Bayesian approaches, Dempster-Shafer's theory of evidence is used for maneuver detection, albeit less frequently. An example can be found in [250], where it is employed for the detection of the maneuvers *lane change*, *vehicle following*, *overtaking*, *cut-in*, and *free flow*. Transitions from one maneuver to another are modeled in a state diagram, which are executed based on an estimated belief that in turn results from a modified Dempster-Shafer combination rule.

On an even higher level of scene understanding, different types of logics are employed. Description logic is used in [111] for reasoning at road intersections. As description logic does not allow the inclusion of uncertainty, probabilistic logics in form of Markov logic networks have recently been proposed for the task in [97]. They combine first-order predicate logic with Markov random fields and allow inference under uncertainty. The intended abstraction level of this type of logics, however, imposes very high requirements not only on the environment perception but also on the processing power [97].

Traffic situations can also be recognized and predicted by classifying vehicle trajectories. An example for a classification of a *lane change* of a preceding vehicle can be found in [95]. Features are the preceding vehicle's trajectory together with its appearance change due to the rotation during the lane change. A trained support vector machine is used for classification. More generally, the approach described in [125] recognizes situation classes at road intersections. These situation classes represent combinations of the basic driving maneuvers *turn right*, *turn left*, and *drive straight* for two considered vehicles at an intersection. Predicted motion trajectories are employed as features, which are classified with a polynomial classifier. The motion trajectories themselves stem from a long-term trajectory prediction method based on pattern recognition in previously labeled motion pattern databases as already explained in Section 6.2.2. An interesting aspect of the approach involves the interaction model between vehicles that is – in contrast to many other approaches – not solely based on the assumption that drivers try to avoid hazardous situations completely, but makes this assumptions only for TTC values larger than the typical driver reaction time. The advantage of employing different classifiers for different situations is further highlighted in [39]. Therein, scenario-specific classifiers are organized in a tree, which are activated as soon as a specific scenario is detected to cope with a larger variety of traffic scenes.

Conclusion

In conclusion, methods of situation recognition and prediction typically reason about the future traffic scene on an abstract, qualitative level. They provide answers to the question, which driving maneuver a traffic participant is executing currently or in the near future in a given traffic environment but do not provide predictions on the spatio-temporal trajectory level relevant for quantitative criticality assessment.[6] Moreover, it is important for the task of criticality assessment to detect maneuvers and situations at their beginning and not when they are already finished. This necessity limits the applicability of several approaches that recognize ma-

[6] This holds for all approaches except the already mentioned exceptions [142] and [88]. In the former, layered HMMs for maneuver detection are combined with spatio-temporal predictions, whereas the latter includes an approach to simultaneous intention and trajectory estimation but without criticality assessment.

neuvers not until the last state transition condition is fulfilled. A common denominator in the relevant literature is the need to cope with incomplete, uncertain, or vague information – be it in form of Bayes, Dempster-Shafer, Fuzzy theory, or their combinations.[7] This is understandable if one thinks about the different ways a driving maneuver can be conducted and strict rules consequently lack the necessary flexibility for their description. Another important aspect is that maneuver and situation recognition and prediction systems have to be interpretable by humans. This explains why other mathematical tools such as neuronal networks are not relevant in this area. In addition, the majority of approaches are tailored towards a single or a few specific maneuvers and can hardly be generalized. Finally, similar to the previously mentioned long-term trajectory prediction methods, most approaches favor the detection and prediction of average, reasonable traffic situations and cannot detect abnormal driving maneuvers. This is, however, a necessity for criticality assessment – the topic of the next subsection.

6.2.4 Criticality Assessment

Methods for criticality assessment try to quantify the criticality of the current or near-future traffic situation. Most often, deterministic *Time-To-X* (TTX) metrics are used, which describe the remaining time until an event X happens. The most prominent representative of this type is the *Time-To-Collision* (TTC), which quantifies the remaining time until a collision happens given an assumed dynamic prediction model.[8] Other time metrics are the *Time-To-React* (TTR) [101, 242] (Fig. 6.5a), and its approximations via *Time-To-Brake* (TTB), *Time-To-Kickdown* (TTK), and *Time-To-Steer* (TTS) introduced in [101] and used in [214] for overtaking situations. In contrast to TTC, the TTR corresponds to the time a driver has left to start a maneuver that circumvents a collision. The TTR thus additionally takes possible, future evasive driver actions into account. The same holds for the two approaches described in [44, 129]. In the former, a critical situation, which results in triggering an autonomous

[7] Note that combining Fuzzy and probability theory as done in some mentioned approaches lacks a theoretical basis as a fuzziness cannot be interpreted or converted to a probability. Fuzzy logic focuses on the ambiguities in describing events, whereas probability theory deals with uncertainty about the occurrence of events [190].

[8] Note that for some authors, e.g. [272], the term TTC already implies a CV motion assumption. In this thesis, TTC is to be seen in the more general form with an arbitrary, specified, underlying prediction model.

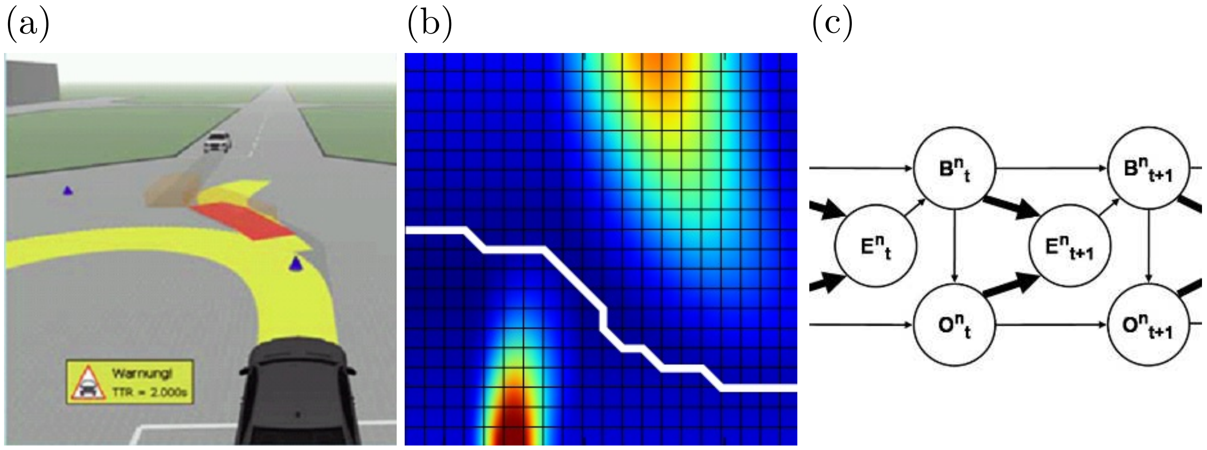


Figure 6.5: Examples of criticality assessment methods. TTR-based [242] (a), via risk map [60] (b), by mismatch detection between intention and expectation [148] (c). Images taken from the mentioned sources.

emergency brake, is defined as the point in time in which no driver action can prevent the collision under consideration of all physically possible trajectories, whereas the latter restricts the space of evasive maneuvers the driver is supposed to perform. Both approaches, however, do not allow a timely warning of the driver before the intervention. Other metrics are the so-called *Predicted-Minimum-Distance* (PMD) and the *Time-To-Predicted-Minimum-Distance* (TPMD) introduced in [199]. They provide a method to warn in near-collision scenarios as a situation is already considered as critical if predicted trajectories come close to each other and the minimum object distances fall below a threshold. Predicted positions are calculated as point estimates stemming from an overlay of different state prediction models and are treated as deterministic, true values in the calculation of PMD and TPMD. The problem of combining point estimates, such as MMSE estimates, of different prediction models is that the combined position estimate can be completely wrong in the multimodal case.[9]

All time metrics have long been treated as deterministic quantities. Therefore, the calculated values are only correct if the predicted future coincides exactly with reality. This is generally not the case as the previous discussion about the complexity of trajectory prediction and the

[9] As an example, we consider one point estimator is based on the assumption of a right turning vehicle and another one on a left turning vehicle. In this case, the combination of both, equally weighted position point estimates would falsely result in a combined position point estimate straight ahead.

involved uncertainties revealed. Therefore, a recent extension [29] additionally considers state uncertainty by calculating a probability distribution over TTCs via the unscented transformation. Driving maneuver uncertainties are, however, not taken into account. In [109], the authors calculate an evolution of collision probability over predicted time instants based on a weighted combination of a lane change maneuver model and a CTRA model. A problem considering only individual future time instants for criticality assessment as done here is that possibly small collision probabilities within each future time step do not necessarily imply an uncritical situation. The criticality of the current situation can arise by an accumulation of criticality over a future time horizon – a topic treated further in Section 6.3.4.

Another criticality assessment approach is based on so-called continuous future risk indicator functions [74], which incorporate uncertainty via critical event rates. This corresponds to modeling uncertainty not on trajectory level but on risk level, and a possible generalization of TTC is derived by considering distance-dependent event rates. A so-calculated risk can be represented in a risk map [60] (Fig. 6.5b), and the approach allows to generate warnings in near-collision scenarios in contrast to deterministic TTX metrics. In [74], the risk is, however, only “blurred” unimodally and symmetrically around the distance to closest encounter of two predicted vehicles and neither reflects the full multimodal uncertainties from different driving maneuvers nor nonlinear risk distortions.

In contrast to the approaches mentioned so far, risk assessment can also be realized by comparing between what a driver is expected to do in a given traffic situation and the driver’s actual intention. Expectations and intentions are estimated jointly in a Bayesian framework in [148] (Fig. 6.5c), and risk is computed as the probability that expectation and intention do not match via particle filtering. This criticality metric, however, does not include any information *when* the situation is likely to become critical but only *whether* it becomes critical. Thus, it cannot be coupled with warning and intervention time thresholds in contrast to TTX metrics.

Conclusion

In conclusion, few criticality assessment systems consider uncertainty in the first place, yet alone ambiguities in driving maneuvers. This is, however, essential for correct situation criticality assessment. Existing systems are also often designed only for specific driving scenarios and cannot cope with arbitrary static and dynamic objects, or are even restricted to a single object that is considered the only relevant one. TTX and especially TTC metrics have a long history and are predominant in this regard due to their ease of interpretation and because they provide a time quantity, which can easily be related to reaction times or system dead times. To permit a straightforward combination with existing systems relying on TTC, it seems reasonable to keep the advantageous properties of time metrics but to generalize them to *arbitrary*, *uncertain*, *multi-object* driving environments and *longer prediction time spans*. This generalization is the ultimate goal of the proposed approach outlined in the next section, in which the so-called *Time-To-Critical-Collision-Probability* (TTCCP) time metric is introduced. As the drivers' intentions manifest in form of high-level driving maneuvers as shown in Section 6.2.3, it seems beneficial to infer these more abstract hidden states in the first place. The results of the abstract, qualitative maneuver detection are then connected with the quantitative trajectory prediction domain for calculating the TTCCP in a probabilistic way.

6.3 Proposed Approach

In the following, the proposed approach to long-term trajectory prediction and criticality assessment is presented, which builds upon the introduced environment representation of Section 2.3. Excerpts have already been published by the author in [222, 224].

6.3.1 System Overview

The proposed system is subdivided into three main parts: The maneuver detection, the prediction, and the criticality assessment as shown in Fig. 6.6.

In the maneuver detection part (blue), the current driving maneuver of every vehicle is estimated via Bayesian inference. For this purpose, a BN is modeled and fed with measured vehicle and environment evidences. The inference result is a pmf $p(M_{\mathrm{E},k})$ of the discrete driving maneuver random variable $M_{\mathrm{E},k}$ for the ego vehicle E at each time step k as well as for the $i \in \{1, \dots, f\}$ other vehicles V_i within the observed traffic scene, $p(M_{\mathrm{V}_i,k})$.

In the prediction part (orange), maneuver-specific prediction models are employed to predict the configuration $\boldsymbol{x} = (x, y, \psi)^T$ of each vehicle forward in time within a common global coordinate system. The individ-

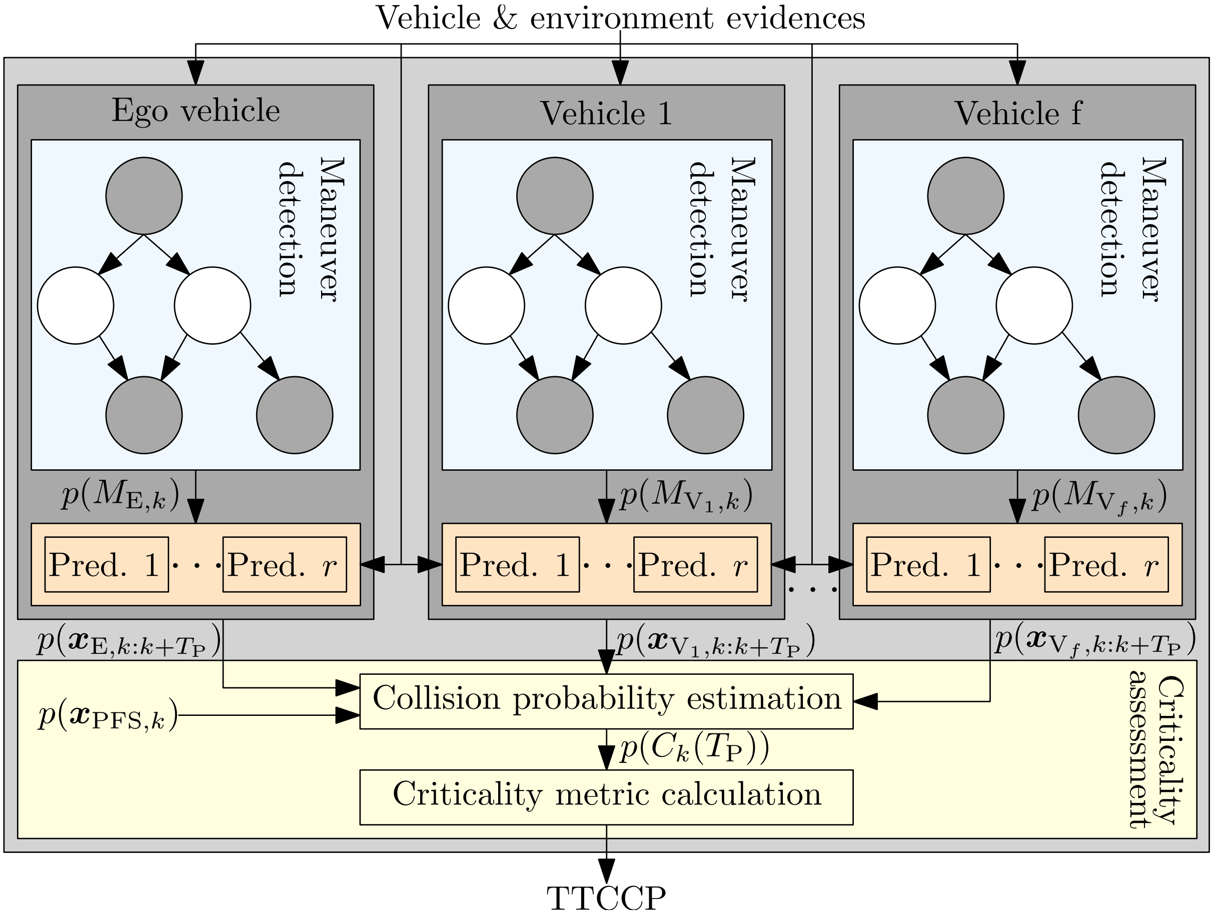

Vehicle & environment evidences
Ego vehicle
Vehicle 1
Vehicle f
Maneuver detection
$p(M_{\mathrm{E},k})$
$p(M_{\mathrm{V}_1,k})$
$p(M_{\mathrm{V}_f,k})$
Pred. 1
Pred. r
$p(\boldsymbol{x}_{\mathrm{E},k:k+T_\mathrm{P}})$
$p(\boldsymbol{x}_{\mathrm{V}_1,k:k+T_\mathrm{P}})$
$p(\boldsymbol{x}_{\mathrm{V}_f,k:k+T_\mathrm{P}})$
$p(\boldsymbol{x}_{\mathrm{PFS},k})$
Collision probability estimation
$p(C_k(T_\mathrm{P}))$
Criticality metric calculation
Criticality assessment
TTCCP

Figure 6.6: Overview of the proposed long-term trajectory prediction and criticality assessment system.

ual r prediction models are adapted to the current driving environment and uncertainties in the drivers' future maneuver realizations are taken into account by introducing uncertainties within these models. Therefore, even if it was known that a driver is performing a specific maneuver for sure, the prediction model would nevertheless generate many possible trajectory realizations of this maneuver. The result of the prediction part is a joint pdf $p(\boldsymbol{x}_{k:k+T_\mathrm{P}})$ of future configurations over the $T_\mathrm{P} \in \mathbb{N}$ prediction time steps for each vehicle.

In the criticality assessment part (yellow), these individual joint distributions are used, together with a PFS map-based representation of the static environment with pdf $p(\boldsymbol{x}_{\mathrm{PFS},k})$, to estimate the collision probability $p\big(C_k(T_\mathrm{P})\big)$ of the event that the ego vehicle collides with at least one other vehicle or the static driving environment at least once within the prediction horizon $\{k{:}k+T_\mathrm{P}\}$ via Monte Carlo simulation. Subsequently, the TTCCP criticality metric is calculated as the necessary prediction time until the probability of this collision event exceeds a certain value. It is therefore a time metric analogous to TTC, which, however, takes many possible future traffic scene evolutions into account. In the following, the three parts are explained in more detail.

6.3.2 Maneuver Detection with Bayesian Networks

For maneuver detection, a BN is used because of

- its good interpretability by human experts,
- the straightforward expandability,
- the possibility to fuse heterogeneous information,
- its handling of incomplete and uncertain knowledge,
- the capability to perform diagnostic and causal reasoning, and
- its seamless incorporation into the complete system due to the same underlying Bayesian theory as the preceding mapping and tracking algorithms.

Relevant background on modeling domain knowledge in BNs can be found in [30, 115, 130, 133, 182, 190].

An important aspect in the proposed BN is that it is modeled with the application *criticality assessment* in mind and therefore needs to permit the detection of exceptional, non-standard situations. This can hardly be accomplished by modeling a single, complex network that takes interactions between all traffic participants into account. This approach would inevitably lead to the design of average traffic situations, thus preventing the detection and subsequent prediction of rare events.[10] In contrast, a separate network is instantiated for every vehicle, which primarily obtains evidences independent from other traffic participants. The last statement can be justified by thinking about, for example, adding the evidence that a neighboring vehicle occupies the ego vehicle's adjacent lane such as done in [122]. Then, this evidence could only be reasonably included into the network in a way that it lowers the probability of an impending lane change for the ego vehicle due to the occupied adjacent lane. If weighted

[10] This could, however, be a reasonable approach for other applications such as cognitive, autonomous vehicles that need the understanding of the standard future traffic scene evolution for adequate decision making.

too strongly, this evidence would prevent the prediction of a collision if the ego vehicle indeed performed the unlikely lane change, and is therefore intentionally not included in the proposed network. As a result of these considerations, the network is designed under the implicit premise of drivers overlooking each other, at least, unless physical evidence contradicts this premise. An example is a vehicle that already brakes strongly in front of an obstacle. In this situation, the prediction of the timely stop prevents false warnings and is therefore incorporated.

The developed network[11] is shown in Fig. 6.7. It consists of 8 binary (true and false) maneuver nodes within the maneuver layer (blue), 6 helper nodes (yellow), and 16 evidence nodes (shaded), which are summarized in Table 6.2. All naturally continuous random variables, such as velocities, are discretized by dividing their values into several reasonable intervals. Thus, the complete network is discrete. Note further that not either causal (predictive – from cause to effect as in [122]) or diagnostic (evidential – from effect to cause as in [216]) reasoning is used, but

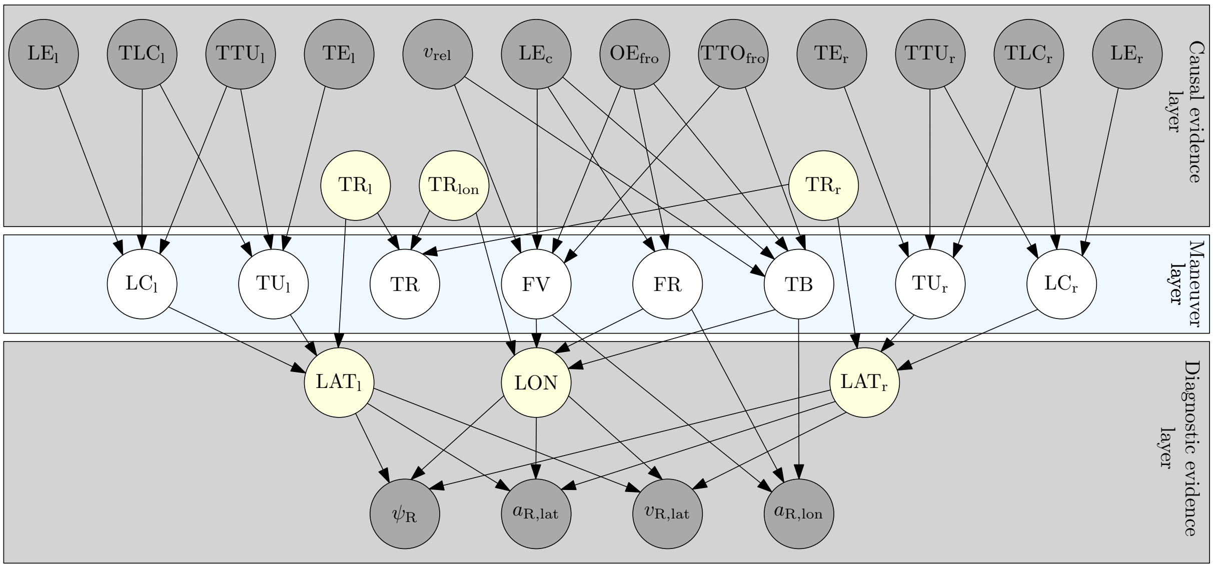


Figure 6.7: Bayesian network for maneuver detection instantiated for each vehicle in the traffic scene at each time step k. Hidden maneuver nodes (blue) are inferred via Bayesian inference given causal and diagnostic evidence nodes (shaded). Helper nodes (yellow) are used to facilitate parametrization. Abbreviations are explained in Table 6.2, a large-format of this figure is included in appendix A.

[11] Thanks to Yuanwen Qin for his help in the parametrization of the network in his thesis [202].

Table 6.2: Random variables of type *Causal Evidence* (CE), *Diagnostic Evidence* (DE), *Helper Node* (HN), and *Maneuver Node* (MN).

Variable	Type	Explanation
$\mathrm{LE}_{\mathrm{l/r/c}}$	CE	Lane Existence left/right/current
$\mathrm{TLC}_{\mathrm{l/r}}$	CE	Time-To-Line-Crossing left/right (CV assumption)
$\mathrm{TTU}_{\mathrm{l/r}}$	CE	Time-To-Turning left/right (CV assumption)
$\mathrm{TE}_{\mathrm{l/r}}$	CE	Turning Existence left/right
v_{rel}	CE	Relative velocity to object in front
$\mathrm{OE}_{\mathrm{fro}}$	CE	Object Existence in front
$\mathrm{TTO}_{\mathrm{fro}}$	CE	Time-To-Object in front (CV assumption)
ψ_{R}	DE	Yaw angle with respect to road course
$a_{\mathrm{R,lat}}$	DE	Lateral acceleration perpendicular to road course
$v_{\mathrm{R,lat}}$	DE	Lateral velocity perpendicular to road course
$a_{\mathrm{R,lon}}$	DE	Longitudinal acceleration along road course
$\mathrm{TR}_{\mathrm{l/lon/r}}$	HN	Trash class helper left/longitudinal/right
$\mathrm{LAT}_{\mathrm{l/r}}$	HN	Lateral motion left/right
LON	HN	Longitudinal motion
$\mathrm{LC}_{\mathrm{l/r}}$	MN	Lane Change to the left/right
$\mathrm{TU}_{\mathrm{l/r}}$	MN	Turn to the left/right
TR	MN	Trash maneuver class (no maneuver)
FV	MN	Follow Vehicle
FR	MN	Follow Road
TB	MN	Target Brake

both by embedding the hidden maneuver node layer in-between the observable causal and diagnostic evidence node layers, which additionally allows intercausal reasoning. These different types of Bayesian reasoning are contrasted in a simplified BN example shown in Fig. 6.8.

The main consideration in the whole network design process is that the causal evidence layer models requirements for specific maneuvers to happen. The existence of a neighboring lane, for example, is a necessity for a lane change maneuver, or the time until a turning is reached influences the probability of an oncoming turn maneuver. The diagnostic evidence layer, in contrast, models the maneuvers' consequences (symptoms) in form of measurable physical motion states. These diagnostic evidences are longitudinal and lateral vehicle acceleration $a_{\mathrm{R,lon/lat}}$, lateral velocity $v_{\mathrm{R,lat}}$,

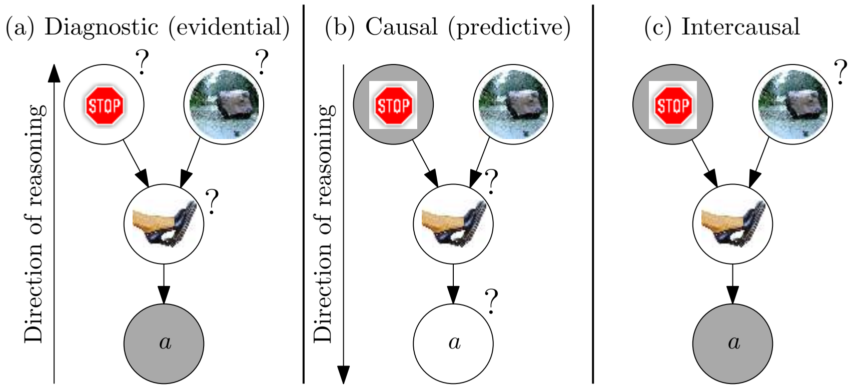


Figure 6.8: Simplified BN example for the explanation of diagnostic (a), causal (b), and intercausal (c) Bayesian reasoning. Variables are stop sign existence, obstacle existence, brake maneuver, and acceleration a. Evidence variables are shaded in gray, query variables, whose posterior distributions are inferred by the specific type of reasoning, are shown with question marks. A reasonable BN model considers both the existence of the stop sign and the obstacle as possible causes for a brake maneuver, therefore the converging connection. The brake maneuver in turn has a negative vehicle acceleration as an effect. Diagnostic reasoning (a) allows reasoning from effects back to its causes, thus in the opposite direction of the network's arcs. If a negative acceleration is observed, the probability of a brake maneuver will rise, which in turn increases the chances of an existing stop sign or an obstacle as possible causes of the observed effect. Causal reasoning (b), in contrast, allows reasoning from causes to effects, thus in the direction of the network's arcs. If the existence of a stop sign is known, the probability of a brake maneuver will rise, which in turn makes a negative acceleration more probable. Intercausal reasoning (c) provides statements about mutual causes of a common effect [133] in the case that something is known about the effect. In the example, the knowledge of a negative acceleration increases the chances of an executed brake maneuver. The additional knowledge about an existing stop sign now lowers the probability that an obstacle is present because the brake maneuver is already explained by the stop sign. This special kind of intercausal reasoning is also called explaining away [190].

and yaw angle ψ_{R} in a road-fixed coordinate system, see Fig. 6.9.[12] These are partly not directly connected with the maneuver layer but via helper nodes $\mathrm{LAT}_{\mathrm{l/r}}$ and LON for lateral and longitudinal motion, respectively. This simplifies the parametrization of the corresponding CPTs, which are summarized in appendix A.

The combination of causal and diagnostic evidences allows so-called ex-

[12] The figure already contains additional quantities necessary for the later introduced prediction models.

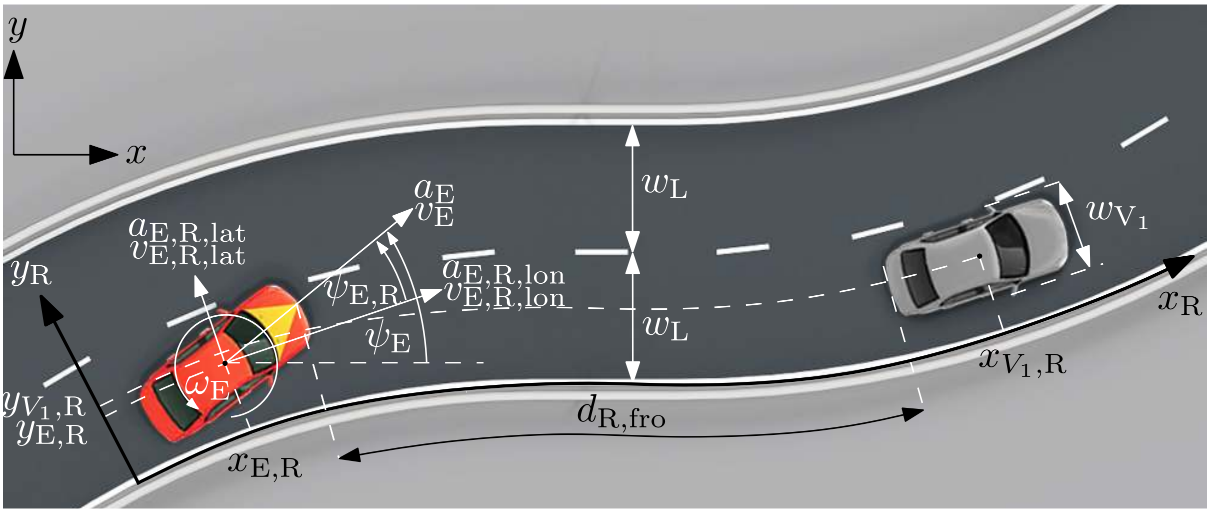


Figure 6.9: Ego vehicle (red, index E) with vehicle in front (gray, index V_1) in road-fixed and global coordinate systems.

plaining away [190]. If, for example, the observed states of the diagnostic nodes are indistinguishable consequences of either a lane change or a turn maneuver, then the additional causal knowledge of a non-existing turning automatically raises the probability that the observed diagnostic evidence stems from a lane change. Moreover, the design approach allows to integrate a trash maneuver class TR for modeling all motions not belonging to a specific standard maneuver class, e.g. irrational movements of a drunken driver, by explaining away all contradictory evidence to the trash maneuver class via the helper nodes $\mathrm{TR}_{\mathrm{l/lon/r}}$. These are connected to TR in a noisy-or manner [115]. The trash maneuver class also gets a high probability if all other maneuver requirements are missing, e.g. no lane or turning information is available such as in completely unstructured environments like parking zones or dirt tracks.

In every time step k, the network is updated with all available evidences. This is realized with the help of the junction tree algorithm [143], which provides exact and fast inference results for discrete random variables as used here. Then, the individual maneuver nodes are normalized so that they form a single, valid, discrete maneuver random variable M with the eight driving maneuver states and corresponding pmf $p(M)$.

6.3.3 Maneuver-Based, Long-Term Trajectory Prediction

For each maneuver, a prediction model is set up in this section. Each model is adapted to the current traffic scene by taking vehicle and environment evidences, such as lane width or the distance to a stopping point, into account. The models additionally contain random components to describe different driving styles. Possible predicted trajectories of each maneuver class are qualitatively visualized in Fig. 6.10. Note that the models introduced in the following are not to be seen as the sole correct ones for the different maneuver predictions but rather as exemplary, working implementations, which can easily be replaced by arbitrary other stochastic processes due to the generality of the approach.

Follow Road **(FR)**

The FR model predicts the vehicle's longitudinal position along the road according to a (nearly) *Constant Acceleration* (CA) model, also called discrete Wiener process acceleration model [18], given by

$$\underbrace{\begin{pmatrix} x_{\mathrm{R},m+1} \\ v_{\mathrm{R,lon},m+1} \\ a_{\mathrm{R,lon},m+1} \end{pmatrix}}_{\boldsymbol{x}_{\mathrm{R},m+1}} = \underbrace{\begin{pmatrix} 1 & T & \frac{1}{2}T^2 \\ 0 & 1 & T \\ 0 & 0 & 1 \end{pmatrix}}_{\boldsymbol{A}_m} \underbrace{\begin{pmatrix} x_{\mathrm{R},m} \\ v_{\mathrm{R,lon},m} \\ a_{\mathrm{R,lon},m} \end{pmatrix}}_{\boldsymbol{x}_{\mathrm{R},m}} + \underbrace{\begin{pmatrix} \frac{1}{2}T^2 \\ T \\ 1 \end{pmatrix}}_{\boldsymbol{\gamma}_m} w_{\mathrm{a}}, \tag{6.1}$$

with state vector $\boldsymbol{x}_{\mathrm{R},m}$ including position $x_{\mathrm{R},m}$, longitudinal velocity $v_{\mathrm{R,lon},m}$ and acceleration $a_{\mathrm{R,lon},m}$ along the road, noise gain vector $\boldsymbol{\gamma}_m$, process noise scalar w_{a} and prediction sampling time T within the prediction horizon $m \in \{k{:}k + T_{\mathrm{P}}\}$. The noise corresponds to a zero-mean, white, Gaussian sequence representing acceleration increments, so that the covariance matrix of the process noise multiplied by the gain is given by

$$\boldsymbol{Q}_m = \boldsymbol{\gamma}_m \sigma^2_{\Delta a_{\mathrm{R,lon}}} \boldsymbol{\gamma}_m^T = \begin{pmatrix} \frac{1}{4}T^4 & \frac{1}{2}T^3 & \frac{1}{2}T^2 \\ \frac{1}{2}T^3 & T^2 & T \\ \frac{1}{2}T^2 & T & 1 \end{pmatrix} \sigma^2_{\Delta a_{\mathrm{R,lon}}}. \tag{6.2}$$

The predicted acceleration is therefore normally distributed around the current acceleration with linearly growing variance over the prediction horizon. This models the rising uncertainty in the driver's acceleration

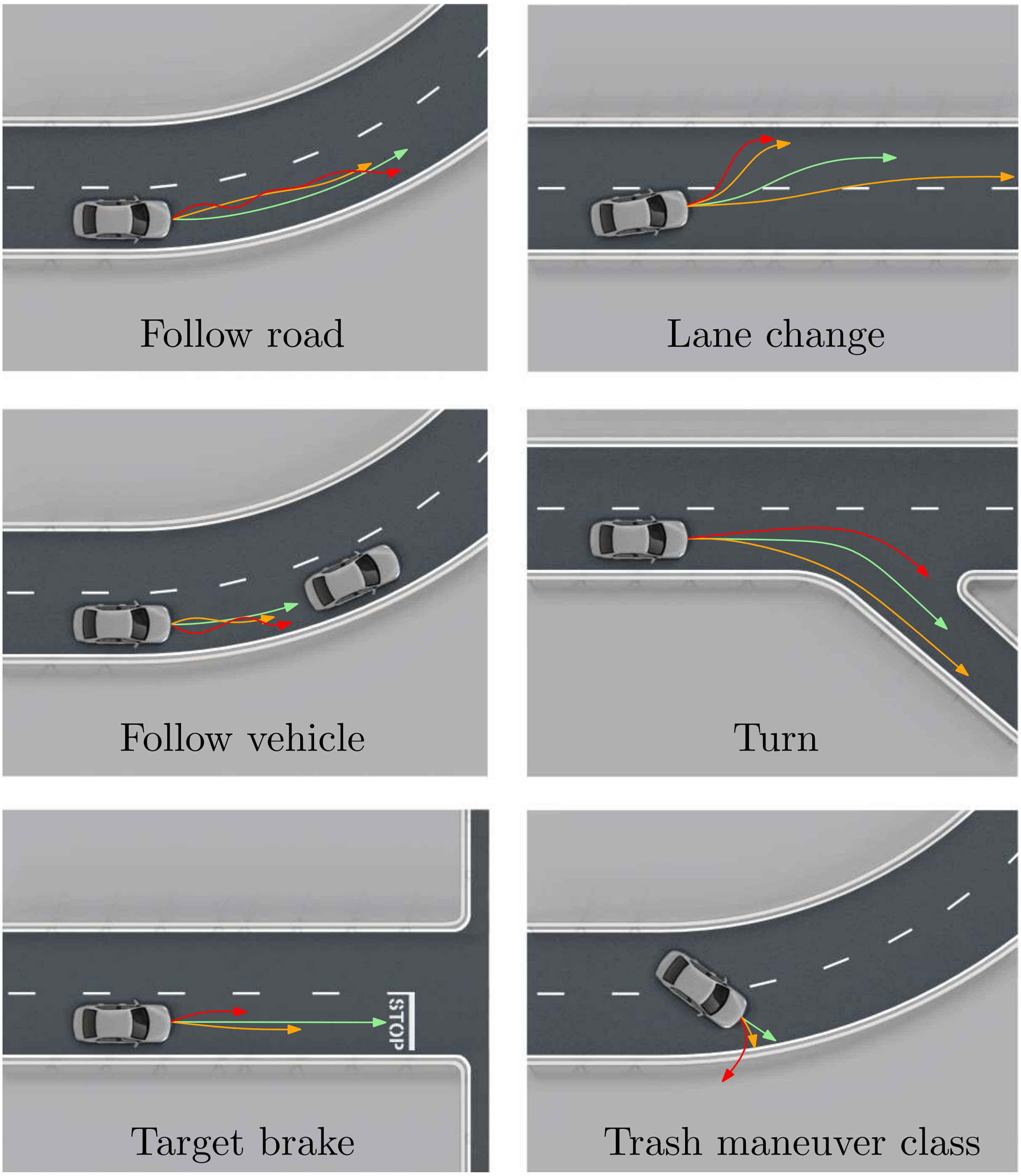


Figure 6.10: Exemplary predicted trajectories of the maneuver-based prediction models. More probable predictions are shown in green, less probable ones in orange and red.

profile the further we predict into the future. Negative velocities are suppressed as we do not want to predict that a decelerating vehicle moves backwards after braking to a standstill.

The vehicle's lateral position along the road $y_\mathrm{R}(t)$ is modeled by an intuitively parametrizable, continuous-time Ornstein-Uhlenbeck process [18], described by the stochastic differential equation

$$\dot{y}_\mathrm{R}(t) = -\alpha y_\mathrm{R}(t) + \alpha u + w_\mathrm{c}(t), \quad \alpha > 0, \tag{6.3}$$

with constant input scalar u for shifting the long-term mean as well as a zero-mean, white, Gaussian noise process $w_\mathrm{c}(t)$ with autocorrelation function

$$\mathbb{E}\big(w_\mathrm{c}(t)w_\mathrm{c}(t+\tau)\big) = Q_\mathrm{n}\delta(\tau) = 2\alpha\sigma_{y_\mathrm{R}}^2\delta(\tau) \tag{6.4}$$

and constant noise intensity scalar Q_n. The process has an exponentially decaying autocovariance function

$$\mathbb{E}\big((y_\mathrm{R}(t) - \bar{y}_\mathrm{R}(t))(y_\mathrm{R}(t+\tau) - \bar{y}_\mathrm{R}(t+\tau))\big) = \sigma_{y_\mathrm{R}}^2 e^{-\alpha|\tau|}, \tag{6.5}$$

with $\bar{y}_\mathrm{R}(\cdot) = \mathbb{E}[y_\mathrm{R}(\cdot)]$ and time constant $T_\mathrm{c} = \frac{1}{\alpha}$. The free parameters to be specified are u, T_c, and σ_{y_R}. The long-term mean is supposed to lie in the middle of the lane of width w_L, therefore $u = \frac{1}{2}w_\mathrm{L}$. The time constant T_c affects the decay rate of the mean evolution and models how fast – on average – a vehicle currently not in the middle of the lane reaches the middle of the lane again. The parameter $\sigma_{y_\mathrm{R}}^2$ in Q_n equals the limiting value of the variance as the variance evolution follows[13]

$$\dot{P}_{y_\mathrm{R}}(t) = -2\alpha P_{y_\mathrm{R}}(t) + Q_\mathrm{n}, \tag{6.6}$$

therefore

$$P_{y_\mathrm{R}}(t) = e^{-2\alpha t}P_{y_\mathrm{R}}(0) + \frac{Q_\mathrm{n}}{2\alpha}(1 - e^{-2\alpha t}) \tag{6.7}$$

and

$$\lim_{t\to\infty} P(t) = \frac{Q_n}{2\alpha} = \sigma_{y_\mathrm{R}}^2. \tag{6.8}$$

The parameter $\sigma_{y_\mathrm{R}}^2$ is set in a way that a vehicle of width w_V driving in the middle of a lane of width w_L is completely inside the lane corresponds to

[13] This follows from the Lyapunov equation $\dot{\boldsymbol{P}}(t) = \boldsymbol{A}_\mathrm{c}(t)\boldsymbol{P}(t) + \boldsymbol{P}(t)\boldsymbol{A}_\mathrm{c}^T(t) + \boldsymbol{Q}_\mathrm{n}(t)$, which describes the covariance matrix evolution $\boldsymbol{P}(t)$ of a stochastic state space system with system matrix $\boldsymbol{A}_\mathrm{c}(t)$, triggered by white noise with noise intensity matrix $\boldsymbol{Q}_\mathrm{n}(t)$, see [18], by setting $\boldsymbol{A}_\mathrm{c}(t) = -\alpha$ and $\boldsymbol{Q}_\mathrm{n}(t) = Q_\mathrm{n}$.

the three sigma interval. Equation (6.3) is now discretized via zero-order hold, see (3.15), p. 24, which results in

$$y_{\mathrm{R},m+1}=e^{-\alpha T}y_{\mathrm{R},m}+(1-e^{-\alpha T})u+w_{y_{\mathrm{R}}}\ ,\forall m\in\{k{:}k+T_{\mathrm{P}}\},\qquad(6.9)$$

with process noise scalar $w_{y_{\mathrm{R}}}$ with variance $Q_m=\sigma^2_{y_{\mathrm{R}}}(1-e^{-2\alpha T})$. Note that the variance is the effect of an exact noise discretization according to (3.16), p. 24. The advantageous properties for suggesting the Ornstein-Uhlenbeck process for the prediction of the vehicle's lateral position are

- the growing variance to model the rising uncertainty of future lateral positions,
- the boundedness of the variance with respect to the lane width, which prevents false warnings, e.g. in the case of an oncoming vehicle on an adjacent lane, and
- the stochastic dependence between future time steps that prevents unrealistic sudden jumps in lateral position.

Fig. 6.11 shows exemplary realizations of the process along with its mean and variance evolutions.

It remains to specify the yaw angle distribution. The yaw angle along the road is modeled as normally distributed around zero with small allowed deviations as the vehicle's orientation is supposed to nearly correspond to the road orientation in a FR maneuver. Consequently, it reads

$$p(\psi_{\mathrm{R},m}|M=\mathrm{FR})=\mathcal{N}\left(\psi_{\mathrm{R},m};0,\sigma^2_{\psi_{\mathrm{R}}}\right),\ \forall m\in\{k:k+T_{\mathrm{P}}\}.\qquad(6.10)$$

The corresponding joint distributions over the prediction horizon can now be derived straightforwardly. The longitudinal joint distribution $p(\boldsymbol{x}_{\mathrm{R},k:k+T_{\mathrm{p}}}|M=\mathrm{FR})$ including the longitudinal position joint distribution $p(x_{\mathrm{R},k:k+T_{\mathrm{p}}}|M=\mathrm{FR})$ along the road, is given by

$$p(\boldsymbol{x}_{\mathrm{R},k:k+T_{\mathrm{p}}}|M=\mathrm{FR})=p(\boldsymbol{x}_{\mathrm{R},k})\prod_{m=k}^{k+T_{\mathrm{p}}-1}p(\boldsymbol{x}_{\mathrm{R},m+1}|\boldsymbol{x}_{\mathrm{R},m}),\qquad(6.11)$$

with $p(\boldsymbol{x}_{\mathrm{R},m+1}|\boldsymbol{x}_{\mathrm{R},m})=\mathcal{N}(\boldsymbol{x}_{\mathrm{R},m+1};\boldsymbol{A}_m\boldsymbol{x}_{\mathrm{R},m},\boldsymbol{Q}_m)$ because of the first-order Markov property of the linear CA model. The lateral position joint distribution can analogously be formulated as

$$p(y_{\mathrm{R},k:k+T_{\mathrm{p}}}|M=\mathrm{FR})=p(y_{\mathrm{R},k})\prod_{m=k}^{k+T_{\mathrm{p}}-1}p(y_{\mathrm{R},m+1}|y_{\mathrm{R},m}),\qquad(6.12)$$

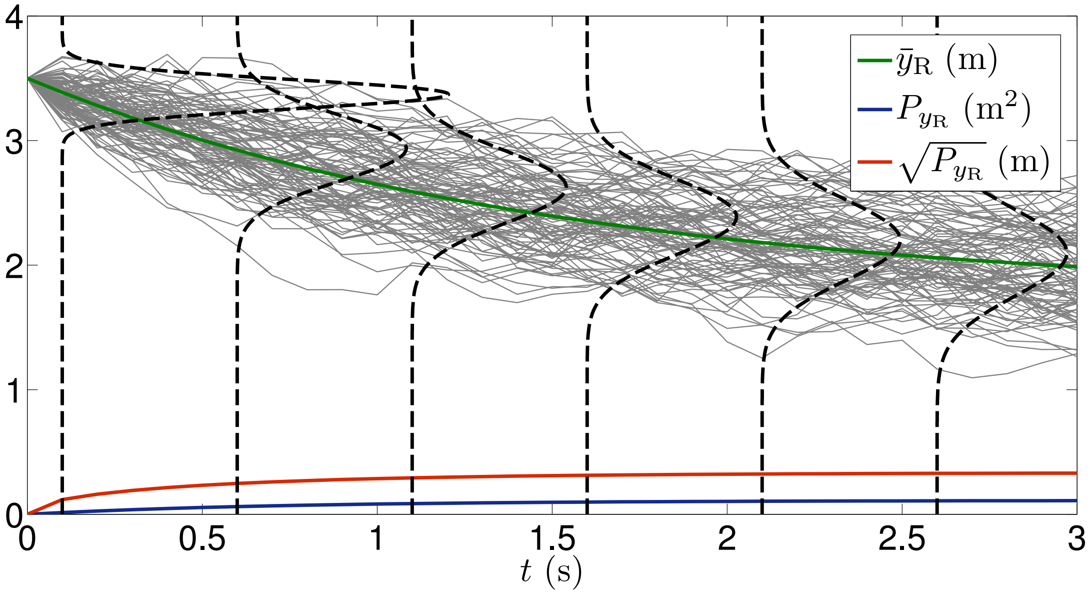


Figure 6.11: Ornstein-Uhlenbeck lateral prediction model for a lane width $w_{\mathrm{L}} = 3.5\,\mathrm{m}$, vehicle width $w_{\mathrm{V}} = 1.5\,\mathrm{m}$ with parameters $u = 1.75\,\mathrm{m}$, $T_{\mathrm{c}} = 1.5\,\mathrm{s}$, $\sigma_{y_{\mathrm{R}}} = \frac{1}{3}\,\mathrm{m}$, and initial values $y_{\mathrm{R}}(0) = 3.5\,\mathrm{m}$ and $P_{y_{\mathrm{R}}}(0) = 0$. Exemplary process realizations are shaded in gray, marginal prediction densities at different predicted time instants are additionally shown dashed.

with $p(y_{\mathrm{R},m+1}|y_{\mathrm{R},m}) = \mathcal{N}(y_{\mathrm{R},m+1}; e^{-\alpha T} y_{\mathrm{R},m} + (1-e^{-\alpha T})u, Q_m)$, whereas the yaw angle joint distribution follows

$$p(\psi_{\mathrm{R},k:k+T_{\mathrm{p}}}|M = \mathrm{FR}) = \prod_{m=k}^{k+T_{\mathrm{p}}} p(\psi_{\mathrm{R},m}|M = \mathrm{FR}) \tag{6.13}$$

due to a simplistic stochastic independence assumption.

Follow Vehicle (FV)

The FV model is based on the assumption that the driver reacts to another vehicle in front and accelerates comfortably in a way that an adequate mean reference time gap τ_{r} occurs in the near future. It assumes moderate reaction of the driver similar to adaptive cruise control systems [276] and does therefore not predict abrupt braking or acceleration maneuvers. The model can predict collisions if the driver reacts to a vehicle in front too slightly. It can also suppress false warnings in situations in which a FR maneuver might already falsely predict a collision, e.g. if a driver reacts moderately to vehicles further away.

The acceleration distribution along the road is modeled as normally distributed around an assumed mean reference acceleration

$$a_{\mathrm{R,lon,r},m} = \begin{cases} c_{\mathrm{FV}}(\Delta\tau_m) a_{\mathrm{f,min}}, & \Delta\tau_m < 0, \\ c_{\mathrm{FV}}(\Delta\tau_m) a_{\mathrm{f,max}}, & \Delta\tau_m \geq 0, \end{cases} \tag{6.14}$$

which depends on the difference between predicted time gap τ_m and an assumed mean reference time gap τ_r, $\Delta\tau_m = \tau_m - \tau_\mathrm{r}$, with τ_m given by[14]

$$\tau_m = \frac{d_{\mathrm{R,fro},m}}{v_{\mathrm{R,lon},m}}, \tag{6.15}$$

with distance to the vehicle in front $d_{\mathrm{R,fro},m}$ and scaling function [266]

$$c_{\mathrm{FV}}(\Delta\tau_m) = \min\left(1; \frac{(\Delta\tau_m)^2}{2\tau_m \mathrm{s}}\right), \quad \forall m \in \{k : k + T_\mathrm{P}\}. \tag{6.16}$$

This unitless scaling function, see Fig. 6.12, models how much the driver exploits the assumed minimum and maximum acceleration, $a_{\mathrm{f,min}}$ and $a_{\mathrm{f,max}}$, in dependence on the predicted time gap and its deviation from the reference time gap. It is based on the assumption that a driver will accelerate smoothly in the case of positive time gap differences but decelerate more strongly in the case of negative $\Delta\tau$. The rising uncertainties in the so-calculated mean reference accelerations are again modeled by a linearly growing acceleration variance over the prediction horizon via

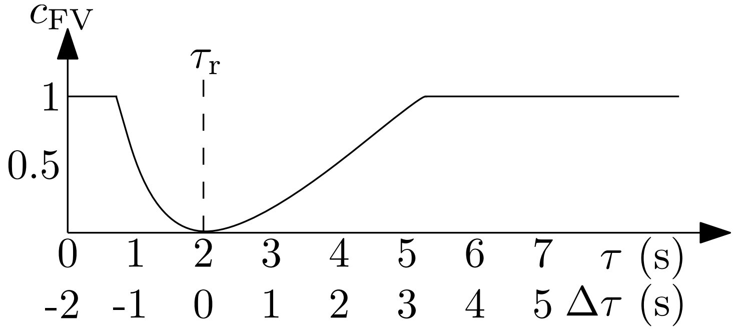


Figure 6.12: Scaling function based on [266].

$$a_{\mathrm{R,lon},m} = a_{\mathrm{R,lon,r},m} + \tilde{a}_{\mathrm{R,lon},m}, \quad \forall m \in \{k : k + T_\mathrm{P}\}, \tag{6.17}$$

[14] For the prediction of the time gap evolution, the preceding vehicle's position is calculated under a constant velocity assumption.

in which $\tilde{a}_{\mathrm{R,lon},m}$ follows the discrete Wiener process

$$\tilde{a}_{\mathrm{R,lon},m+1} = \tilde{a}_{\mathrm{R,lon},m} + w_a, \tag{6.18}$$

with white, Gaussian noise scalar w_a with variance $\sigma^2_{\Delta a_{\mathrm{R,lon}}}$ and $\tilde{a}_{\mathrm{R,lon},k} = 0$. Velocities and positions along the road are given by integration, lateral position and yaw angle distributions are modeled analogous to the FR maneuver, negative velocities are suppressed.

Target Brake (TB)

The TB model predicts under the assumption that the vehicle stops in front of a braking target, such as an obstacle, another vehicle, a stop line, or a red traffic light. The predicted acceleration equals the necessary constant acceleration for the vehicle with current velocity $v_{\mathrm{R,lon},k}$ to stop in a specific safety distance Δd to the braking target under consideration of an achievable minimum acceleration $a_{\min}$, therefore

$$a_{\mathrm{R,lon},m} = \max\Big(\frac{-v^2_{\mathrm{R,lon},k}}{2(d_{\mathrm{R,fro},k} - \Delta d)}; a_{\min}\Big), \quad \forall m \in \{k : k + T_{\mathrm{P}}\}, \tag{6.19}$$

in which the current distance to the target is given by $d_{\mathrm{R,fro},k}$. The safety distance Δd is treated as normally distributed around a mean reference safety distance d_{r} with a standard deviation designed in a way that a vehicle stopping directly in front of the braking target with no safety margin corresponds to the three sigma interval, therefore

$$p(\Delta d | M = \mathrm{TB}) = \mathcal{N}\left(\Delta d; d_{\mathrm{r}}, \left(\tfrac{d_{\mathrm{r}}}{3}\right)^2\right). \tag{6.20}$$

Lateral position and yaw angle distributions are modeled analogous to the FR maneuver, negative velocities are again suppressed.

Lane Change (LC)

The LC model predicts lane changes according to a prototype model that is represented by a sine half-cycle in road coordinates,

$$y_{\mathrm{R}}(\Delta x_{\mathrm{R}}) = \frac{w_{\mathrm{L}}}{2} \sin\left(\frac{\pi}{l_{\mathrm{r}}}\Delta x_{\mathrm{R}} - \frac{\pi}{2}\right) + w_{\mathrm{L}}, \quad \forall \Delta x_{\mathrm{R}} \in [0, l_{\mathrm{r}}], \tag{6.21}$$

with $\Delta x_{\mathrm{R}} = x_{\mathrm{R}} - x_{\mathrm{R,s}}$, maneuver start coordinate $x_{\mathrm{R,s}}$, width of each lane w_{L}, reference maneuver length l_{r}, and gradient angle

$$\begin{aligned}\psi_{\mathrm{R}}(\Delta x_{\mathrm{R}}) &= \arctan\left(\frac{\mathrm{d}y_{\mathrm{R}}(\Delta x_{\mathrm{R}})}{\mathrm{d}(\Delta x_{\mathrm{R}})}\right) \\ &= \arctan\left(\frac{w_{\mathrm{L}}\pi}{2l_{\mathrm{r}}}\cos\left(\frac{\pi}{l_{\mathrm{r}}}\Delta x_{\mathrm{R}} - \frac{\pi}{2}\right)\right),\ \forall \Delta x_{\mathrm{R}} \in [0, l_{\mathrm{r}}].\end{aligned} \tag{6.22}$$

This model is adapted to the current lateral vehicle position and yaw angle state at each time step k by setting $y_{\mathrm{R}}(\Delta x_{\mathrm{R}}) \overset{!}{=} y_{\mathrm{R},k}$ and $\psi_{\mathrm{R}}(\Delta x_{\mathrm{R}}) \overset{!}{=} \psi_{\mathrm{R},k}$ in (6.21) and (6.22), respectively, and solving for $l_{\mathrm{r},k}$ and $\Delta x_{\mathrm{R},k}$, which results in

$$\begin{aligned} l_{\mathrm{r},k} &= \frac{w_{\mathrm{L}}\pi}{2\tan\psi_{\mathrm{R},k}}\cos\left(\arcsin\left(\frac{2y_{\mathrm{R},k}}{w_{\mathrm{L}}} - 2\right)\right), \\ \Delta x_{\mathrm{R},k} &= \left(\frac{1}{2} + \frac{1}{\pi}\arcsin\left(\frac{2y_{\mathrm{R},k}}{w_{\mathrm{L}}} - 2\right)\right) l_{\mathrm{r},k}, \end{aligned} \tag{6.23}$$

to obtain the remaining maneuver length $l_{\mathrm{r},k} - \Delta x_{\mathrm{R},k}$ for the prediction.[15] Uncertainties are introduced by modeling the lateral maneuver origin for these calculations as normally distributed around the middle of the lane with

$$p\left(y_{\mathrm{R},k}\left(\Delta x_{\mathrm{R},k} = 0\right)\right) = \mathcal{N}\left(y_{\mathrm{R},k}\left(\Delta x_{\mathrm{R},k} = 0\right); \tfrac{w_{\mathrm{L}}}{2}, \sigma^2_{y_{\mathrm{R,s}}}\right). \tag{6.24}$$

Consequently, the quantities $l_{\mathrm{r},k}$, $\Delta x_{\mathrm{R},k}$, and $x_{\mathrm{R,s,k}}$ also become random variables.[16] Fig. 6.13 visualizes the mentioned quantities on two exemplary trajectory samples. The prediction itself is carried out with the so-adapted prototype model. The uncertain tangential acceleration along the sine curve is modeled as a discrete Wiener process analogous to the

[15] To perform lane change maneuver predictions before an observable maneuver execution actually started, e.g. the vehicle is still following the road with $\psi_{\mathrm{R},k} \approx 0$, the maneuver length and maneuver start coordinate have to be predefined. In these rare cases (the probability of a lane change is very low but not necessarily zero), we consider the maneuver start coordinate as equal to the current vehicle position at time step k, therefore $\Delta x_{\mathrm{R},k} = 0$, and a reference maneuver length $l_{\mathrm{r},k} = v_{\mathrm{R,lon,k}} T_{\mathrm{r}} + \frac{1}{2} a_{\mathrm{R,lon},k} T_{\mathrm{r}}^2$, depending on the current vehicle velocity and acceleration, with reference maneuver time $T_{\mathrm{r}} = 3\,\mathrm{s}$.

[16] Note that this is just a method to obtain uncertain predictions consistent with the current vehicle configuration and does not imply that the vehicle has truly been on one of the resulting trajectories in the past.

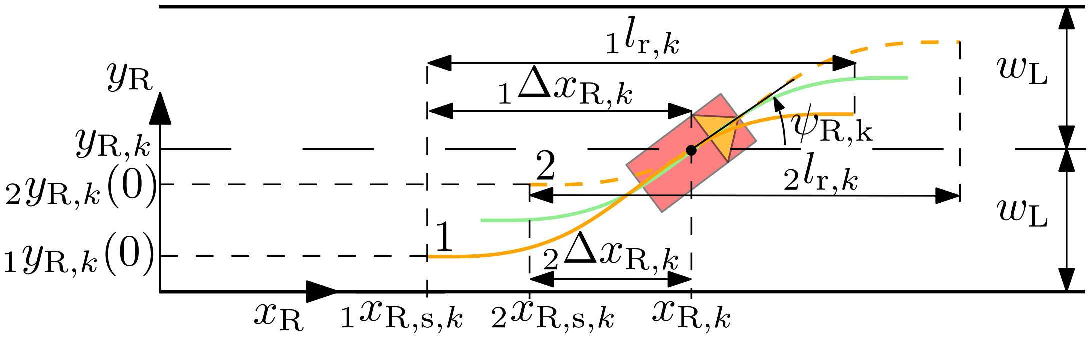


Figure 6.13: Lane change maneuver model for a given vehicle configuration at time step k with relevant quantities shown on two predicted trajectory samples (orange) along with the mean predicted trajectory (green).

FR model. After double integration and transformation of the resulting position along the curve to the road coordinate system[17], this gives the function arguments Δx_R of (6.21) and (6.22), which are in turn used for the uncertain prediction of y_R and ψ_R. As soon as the end of the sine half-cycle is reached, the prediction is carried out with the FR model for the remaining prediction horizon.

Turn (TU)

The TU model predicts turns by partitioning the maneuver into the following three sections:

- Section 1: Before the turn on the initial road.
- Section 2: During the turn.
- Section 3: After the turn on the destination road.

The initial and the destination road are considered straight in the vicinity of the turn, the destination road is rotated versus the initial road by the angle $\alpha \in]0, \pi[$. The right road border[18] is supposed to follow a circular arc $(x_\mathrm{R} - x_\mathrm{c})^2 + (y_\mathrm{R} - y_\mathrm{c})^2 = r_\mathrm{c}^2$ during the road transition with radius r_c and center $M_\mathrm{c} = (x_\mathrm{c}, y_\mathrm{c})$, sketched in Fig. 6.14.

[17] Note that the exact arc length of a sine curve $y = \sin x$ between 0 and x is given by the elliptic integral $\int_0^x \sqrt{1 + (\cos \tilde{x})^2} \mathrm{d}\tilde{x}$, which has no closed solution. The curve is therefore approximated by piecewise linear segments for calculating the resulting longitudinal position along the road.

[18] Explanations are limited to the turn right maneuver.

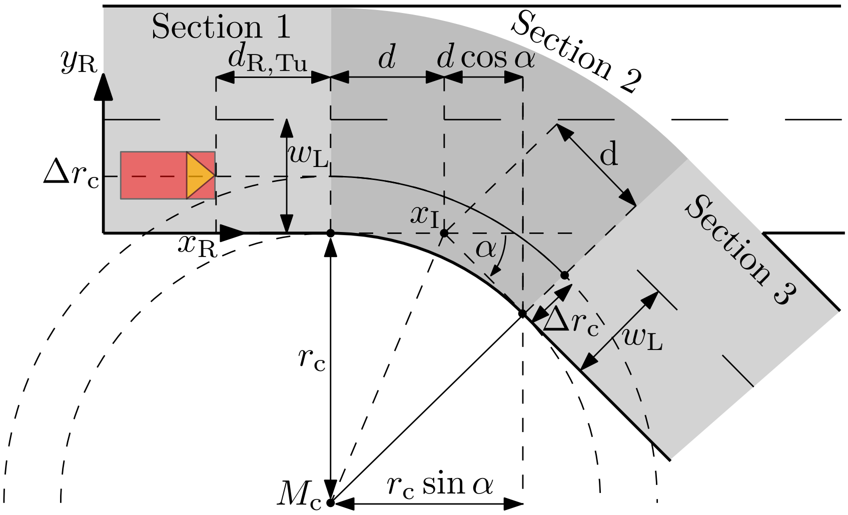


Figure 6.14: Turn maneuver sections with relevant quantities explained in the text.

In the first section, lateral and longitudinal predictions are performed analogous to the FR model with the difference that the predicted longitudinal acceleration is considered normally distributed around a constant reference acceleration $a_{\mathrm{R,lon,r},m}$ over the prediction horizon. This reference acceleration is calculated as the necessary constant acceleration that ensures that a maximum velocity $v_\mathrm{R,lon,max}$ is not exceeded at the start of the turn according to

$$a_{\mathrm{R,lon,r},m} = \begin{cases} \tilde{a} = \max\left(\frac{v_{\mathrm{R,lon},k}\Delta v_k + \frac{1}{2}(\Delta v_k)^2}{d_{\mathrm{R,Tu},k}}; a_\mathrm{min}\right), & \tilde{a} < a_{\mathrm{R,lon},k}, \\ a_{\mathrm{R,lon},k}, & \text{else}, \end{cases} \tag{6.25}$$

with $\Delta v_k = v_\mathrm{R,lon,max} - v_{\mathrm{R,lon},k}$, remaining distance to the turn $d_{\mathrm{R,Tu},k}$ and minimum acceleration a_min. If the current acceleration does not result in an exceeding of $v_\mathrm{R,lon,max}$, the current acceleration value is used for prediction instead. The maximum velocity itself is determined under the assumption that a driver only exposes himself to a maximum centripetal acceleration $a_\mathrm{c,max}$ during the turn phase, so that

$$v_\mathrm{R,lon,max} = \sqrt{a_\mathrm{c,max}(r_\mathrm{c} + \Delta r_\mathrm{c})} \tag{6.26}$$

holds, under consideration of a CV model in this second section. Here, the circular arc's center lies on the angle bisector between the two road tangents. The radius is adapted to the road geometry so that initial and destination roads' right borders are also tangents of the circle. For the calculation of M_c and r_c, we consider the distance d between the start/end of the rounded road transition and the intersection point's longitudinal road coordinate x_I of the two road tangents as known and consequently reach

$$\begin{aligned} x_\mathrm{c} &= x_\mathrm{I} - d = x_\mathrm{I} + d\cos\alpha - r_\mathrm{c}\sin\alpha, \\ \Rightarrow r_\mathrm{c} &= \frac{d(1+\cos\alpha)}{\sin\alpha}, \\ y_\mathrm{c} &= -r_\mathrm{c}. \end{aligned} \tag{6.27}$$

The prediction itself is performed by considering the lateral positions on the initial and destination road as random variables following a normal distribution around the middle of the road to model different turn executions as every driver cuts the corner differently. The predicted circular paths are therefore given by

$$(x_\mathrm{R} - x_\mathrm{c})^2 + (y_\mathrm{R} - y_\mathrm{c})^2 = (r_\mathrm{c} + \Delta r_\mathrm{c})^2, \tag{6.28}$$

with

$$p\left(\Delta r_\mathrm{c}\right) = \mathcal{N}\left(\Delta r_\mathrm{c}; \tfrac{w_\mathrm{L}}{2}, \sigma_{y_\mathrm{R}}^2\right). \tag{6.29}$$

Along these paths, the prediction is performed with a CV model assumption. The third section is again predicted with the FR maneuver model, which is initialized with the final values of the second section.

***Trash Maneuver Class* (TR)**

The TR model predicts the motion according to a (nearly) *Constant Turn Rate and Acceleration* (CTRA) model [227] with

$$\begin{pmatrix} x_{m+1} \\ y_{m+1} \\ \psi_{m+1} \\ v_{m+1} \\ a_{m+1} \\ \omega_{m+1} \end{pmatrix} = \begin{pmatrix} x_m + \tilde{x}_m \\ y_m + \tilde{y}_m \\ \psi_m + \omega_m T \\ v_m + a_m T \\ a_m \\ \omega_m \end{pmatrix} + \boldsymbol{w}_m, \quad \forall m \in \{k{:}k+T_\mathrm{P}\}, \tag{6.30}$$

$$\begin{aligned}
\tilde{x}_m &= \frac{1}{\omega_m^2}[(v_m\omega_m + a_m\omega_m T)\sin(\psi_m + \omega_m T) \\
&\quad + a_m\cos(\psi_m + \omega_m T) - v_m\omega_m \sin\psi_m - a_m\cos\psi_m], \\
\tilde{y}_m &= \frac{1}{\omega_m^2}[(-v_m\omega_m - a_m\omega_m T)\cos(\psi_m + \omega_m T) \\
&\quad + a_m\sin(\psi_m + \omega_m T) + v_m\omega_m \cos\psi_m - a_m\sin\psi_m],
\end{aligned} \tag{6.31}$$

with global position x_m, y_m, yaw angle ψ_m, yaw rate ω_m, and velocity and acceleration magnitude in driving direction v_m and a_m, respectively. The noise with process noise vector $\boldsymbol{w}_m$ corresponds to a zero-mean, white, Gaussian sequence representing acceleration and yaw rate increments with covariance matrix $\boldsymbol{Q}_m = \operatorname{diag}\left(0, 0, 0, 0, \sigma_{\Delta a}^2, \sigma_{\Delta\omega}^2\right)$. The model is purely based on physical vehicle states. It can therefore also be employed if no environment knowledge is available and is used if the vehicle motion does not correspond to any other driving maneuver.

Maneuver Model Combination

In summary, every maneuver prediction model generates, after transformation to a common, global reference frame, a joint pdf $p(\boldsymbol{x}_{k:k+T_\mathrm{P}}|M_{j,k})$ over the prediction horizon with configuration $\boldsymbol{x} = (x, y, \psi)^T$ for all $j \in \{1, \ldots, r\}$ maneuvers for each vehicle. By marginalizing out the maneuver states from the combined pdf $p(\boldsymbol{x}_{\mathrm{E},k:k+T_\mathrm{P}}, M_k)$ with the help of $p(M_{\mathrm{E},k})$ and $p(M_{\mathrm{V}_i,k})$ from the BN inference result of Section 6.3.2, we reach

$$p\left(\boldsymbol{x}_{\mathrm{E},k:k+T_\mathrm{P}}\right) = \sum_{j=1}^{r} p\left(\boldsymbol{x}_{\mathrm{E},k:k+T_\mathrm{P}}|M_{\mathrm{E},k} = j\right) p(M_{\mathrm{E},k} = j), \tag{6.32a}$$

$$p\left(\boldsymbol{x}_{\mathrm{V}_i,k:k+T_\mathrm{P}}\right) = \sum_{j=1}^{r} p\left(\boldsymbol{x}_{\mathrm{V}_i,k:k+T_\mathrm{P}}|M_{\mathrm{V}_i,k} = j\right) p(M_{\mathrm{V}_i,k} = j), \tag{6.32b}$$

for the ego vehicle and the other $i \in \{1, \ldots, f\}$ vehicles, respectively. The emerging densities are therefore a combination of all prediction models, which are weighted by their estimated maneuver probabilities. These densities are the final, condensed result of the trajectory prediction step and are the foundation of performing probabilistic criticality assessment as described in the following.

6.3.4 Criticality Assessment

The criticality assessment part is subdivided into probabilistic collision checking i) with respect to dynamic environment objects such as other vehicles, and ii) with respect to static environment structures described by PFS maps. Subsequently, both results are combined and the TTCCP metric is introduced.

Probabilistic Collision Checking with Dynamic Objects

Probabilistic collision checking is a generalization of deterministic collision checking in the case of uncertain object configurations. Different complexities arise depending on

- whether just one object or every object is considered uncertain,
- whether objects are considered as point objects or as extended objects, and
- which family of distributions, e.g. Gaussians, is used to describe object uncertainty.

In ADAS, *all* objects, including the ego vehicle, are uncertain. Moreover, it is not reasonable to employ a point object abstraction in driving environments due to the narrow road space. Therefore, *extended* objects have to be considered. Last but not least, object configuration distributions that take different driving maneuvers into account are highly multimodal and non-Gaussian. Consequently, the most general form of probabilistic collision checking has to be employed. In [71], a general formula for calculating the collision probability of two uncertain, extended objects at a specific *point in time* is presented. Adapted to the road context, the probability of the event C_I: *The ego vehicle collides exactly with vehicle V_i at a specific, common instant of time*, is given by [71]

$$p(C_\mathrm{I}) = \int_{\boldsymbol{x}_{\mathrm{V}_i}} \int_{\boldsymbol{x}_\mathrm{E}} I_{C_\mathrm{I}}(\boldsymbol{x}_\mathrm{E}, \boldsymbol{x}_{\mathrm{V}_i}) p(\boldsymbol{x}_\mathrm{E}, \boldsymbol{x}_{\mathrm{V}_i}) \mathrm{d}\boldsymbol{x}_\mathrm{E} \mathrm{d}\boldsymbol{x}_{\mathrm{V}_i}, \tag{6.33}$$

with joint density over the uncertain configurations of both vehicles $p(\boldsymbol{x}_\mathrm{E}, \boldsymbol{x}_{\mathrm{V}_i})$ and collision indicator function

$$I_{C_\mathrm{I}}(\boldsymbol{x}_\mathrm{E}, \boldsymbol{x}_{\mathrm{V}_i}) = \begin{cases} 1, & \mathcal{S}(\boldsymbol{x}_\mathrm{E}) \bigcap \mathcal{S}(\boldsymbol{x}_{\mathrm{V}_i}) \neq \emptyset, \\ 0, & \text{else.} \end{cases} \tag{6.34}$$

Therein, $\mathcal{S}(\boldsymbol{x}_{\mathrm{E}})$ and $\mathcal{S}(\boldsymbol{x}_{\mathrm{V}_i})$ refer to the set of points in space that are occupied by the ego and obstacle vehicle when their configurations are given by $\boldsymbol{x}_{\mathrm{E}}$ and $\boldsymbol{x}_{\mathrm{V}_i}$. This general formula holds for a specific *time instant.* Evaluated at different prediction time instants, the progression of collision probability could be calculated. This approach is, however, insufficient as it is of greater interest to know how critical a situation gets within a future *time span*, for example within the next 3 s. For criticality assessment over a future *time span*, i.e. the calculation of the probability of the event $C_{\mathrm{V}_i,k}(T_{\mathrm{P}})$: *The ego vehicle collides with vehicle V_i at least once within the prediction horizon $m \in \{k : k + T_P\}$*, evaluated at time step k, the formula has to be generalized as follows:

$$\begin{aligned} p\big(C_{\mathrm{V}_i,k}(T_{\mathrm{P}})\big) = &\int_{\boldsymbol{x}_{\mathrm{V}_i,k+T_{\mathrm{P}}}} \int_{\boldsymbol{x}_{\mathrm{E},k+T_{\mathrm{P}}}} \cdots \int_{\boldsymbol{x}_{\mathrm{V}_i,k}} \int_{\boldsymbol{x}_{\mathrm{E},k}} \\ &I_C(\boldsymbol{x}_{\mathrm{E},k:k+T_{\mathrm{P}}}, \boldsymbol{x}_{\mathrm{V}_i,k:k+T_{\mathrm{P}}}) p(\boldsymbol{x}_{\mathrm{E},k:k+T_{\mathrm{P}}}, \boldsymbol{x}_{\mathrm{V}_i,k:k+T_{\mathrm{P}}}) \\ &\mathrm{d}\boldsymbol{x}_{\mathrm{E},k}\mathrm{d}\boldsymbol{x}_{\mathrm{V}_i,k} \dots \mathrm{d}\boldsymbol{x}_{\mathrm{E},k+T_{\mathrm{P}}}\mathrm{d}\boldsymbol{x}_{\mathrm{V}_i,k+T_{\mathrm{P}}}, \end{aligned} \tag{6.35}$$

with new, modified collision indicator function

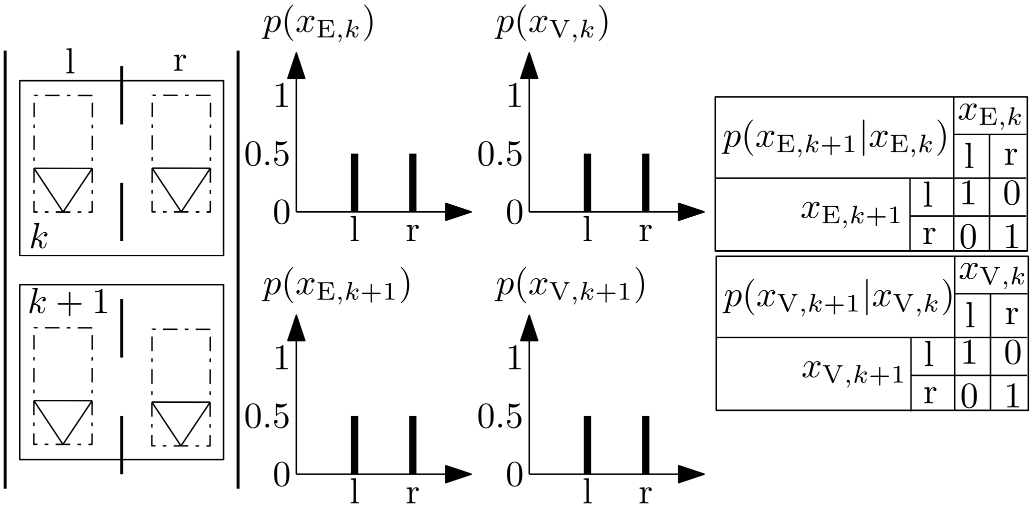
$$I_C(\boldsymbol{x}_{\mathrm{E},k:k+T_{\mathrm{P}}}, \boldsymbol{x}_{\mathrm{V}_i,k:k+T_{\mathrm{P}}}) = \begin{cases} 1, & \exists m \in \{k{:}k + T_{\mathrm{P}}\} : \\ & \mathcal{S}(\boldsymbol{x}_{\mathrm{E},m}) \bigcap \mathcal{S}(\boldsymbol{x}_{\mathrm{V}_i,m}) \neq \emptyset, \\ 0, & \text{else.} \end{cases} \tag{6.36}$$

Therefore, an integration of the joint density of both vehicles' configurations of all future time steps, $p(\boldsymbol{x}_{\mathrm{E},k:k+T_{\mathrm{P}}}, \boldsymbol{x}_{\mathrm{V}_i,k:k+T_{\mathrm{P}}})$, is necessary. This integration needs to be performed over a volume that is defined by the modified collision indicator function. This function in turn is equal to one if there exists at least one time step in the prediction horizon in which the occupancy sets of the specific vehicle configurations overlap, i.e. a predicted collision occurs. Note that just considering the marginal densities $p(\boldsymbol{x}_{\mathrm{E},k}, \boldsymbol{x}_{\mathrm{V}_i,k}), p(\boldsymbol{x}_{\mathrm{E},k+1}, \boldsymbol{x}_{\mathrm{V}_i,k+1}), \dots, p(\boldsymbol{x}_{\mathrm{E},k+T_{\mathrm{P}}}, \boldsymbol{x}_{\mathrm{V}_i,k+T_{\mathrm{P}}})$ is insufficient for the calculation of the event probability as the same marginal prediction densities can lead to different $p\big(C_{\mathrm{V}_i,k}(T_{\mathrm{P}})\big)$. This is clarified with the help of a one-dimensional example in Fig. 6.15.

We further consider the trajectories of all vehicles as independent[19], therefore the joint density in (6.35) factorizes according to

$$p(\boldsymbol{x}_{\mathrm{E},k:k+T_{\mathrm{P}}}, \boldsymbol{x}_{\mathrm{V}_i,k:k+T_{\mathrm{P}}}) = p(\boldsymbol{x}_{\mathrm{E},k:k+T_{\mathrm{P}}})p(\boldsymbol{x}_{\mathrm{V}_i,k:k+T_{\mathrm{P}}}), \tag{6.37}$$

[19] As mentioned already, a collision prediction is hardly possible otherwise as the necessary modeling of the combined traffic scene pdf would require the postulation of sensible individual drivers that try not to collide.



$p(x_{\mathrm{E},k+1}\mid x_{\mathrm{E},k})$		$x_{\mathrm{E},k}$	
		l	r
$x_{\mathrm{E},k+1}$	l	1	0
	r	0	1

$p(x_{\mathrm{V},k+1}\mid x_{\mathrm{V},k})$		$x_{\mathrm{V},k}$	
		l	r
$x_{\mathrm{V},k+1}$	l	1	0
	r	0	1

Figure 6.15: Example showing that identical marginal prediction densities can lead to different collision event probabilities. We suppose that two vehicles, E and V, have independent trajectories and are either on the left (state l) or on the right (state r) lane at each of the two considered time steps k and $k+1$. The vehicles collide if they are on the same lane at the same time. The fictive marginal pmfs of both time steps and vehicles are given in the middle. Each vehicle is thus on the left or right lane with an equal probability of 0.5 at both time steps. First, we consider each vehicle's positions over time as independent, therefore uncorrelated, random variables. In this case, the probability of at least one collision within the time interval $\{k{:}k+T_{\mathrm{P}}\} = \{k, k+1\}$ is straightforwardly given by $p\big(C_{V,k}(T_{\mathrm{P}})\big) = 1 - (0.5^2 + 0.5^2)^2 = 0.75$, i.e. as one minus the product of no collision in both steps. In contrast, we now consider the lane positions of each vehicle over time to be dependent, perfectly correlated random variables with transition probabilities given on the right. Each vehicle is thus modeled as staying on the same lane for both time steps. The marginal pmfs remain the same, whereas the event probability now changes to $p\big(C_{V,k}(T_{\mathrm{P}})\big) = 0.5^2 + 0.5^2 = 0.5$ as a collision in one time step now automatically results in a collision in the next.

with the pdfs on the right hand side given by (6.32). For the evaluation of the indicator function I_C, the vehicles' oriented bounding rectangles are analytically checked for overlap. For performance reasons, this is accompanied by a fast preliminary collision check between the corresponding bounding circles.

Probabilistic Collision Checking with PFS Maps

So far, only other dynamic vehicles have been considered. For a comprehensive system, however, also static obstacles need to be taken into account as it is also dangerous if the driver collides with static structures, such as guardrails or tree alleys. In the following, it is shown how the compact PFS map representation of the static driving environment as explained in Chapter 5 can advantageously be embedded into the concept. The PFS map's state

$$\boldsymbol{x}_{\mathrm{PFS},k} = \left(\boldsymbol{x}_{\mathrm{P}}^{T}\ \boldsymbol{x}_{\mathrm{c},1}^{T} \dots \boldsymbol{x}_{\mathrm{c},N_{\mathrm{c}}}^{T}\ \boldsymbol{x}_{\mathrm{r},1}^{T} \dots \boldsymbol{x}_{\mathrm{r},N_{\mathrm{r}}}^{T}\right)^{T} \in \mathbb{R}^{2N_{\mathrm{b}}+3N_{\mathrm{c}}+5N_{\mathrm{r}}} \tag{6.38}$$

at time step k, see (5.19), p.123, remains constant over the prediction horizon as the static environment does not change during the prediction time. We additionally consider the PFS map as deterministic because uncertainties of the static environment are much smaller than the rising uncertainties involved in the future motion of other dynamic vehicles. Therefore, the collision probability of the event $C_{\mathrm{PFS},k}(T_{\mathrm{P}})$: *The ego vehicle collides with the static driving environment at least once within the prediction horizon* $\{k{:}k+T_P\}$, evaluated at time step k, is given by

$$\begin{aligned} p\big(C_{\mathrm{PFS},k}(T_{\mathrm{P}})\big) = &\int_{\boldsymbol{x}_{\mathrm{E},k+T_{\mathrm{P}}}} \cdots \int_{\boldsymbol{x}_{\mathrm{E},k}} I_{C_{\mathrm{PFS}}}(\boldsymbol{x}_{\mathrm{E},k:k+T_{\mathrm{P}}}) \\ &\qquad p(\boldsymbol{x}_{\mathrm{E},k:k+T_{\mathrm{P}}}) \mathrm{d}\boldsymbol{x}_{\mathrm{E},k} \dots \mathrm{d}\boldsymbol{x}_{\mathrm{E},k+T_{\mathrm{P}}}, \end{aligned} \tag{6.39}$$

with collision indicator function

$$I_{C_{\mathrm{PFS}}}(\boldsymbol{x}_{\mathrm{E},k:k+T_{\mathrm{P}}}) = \begin{cases} 1, & \exists m \in \{k{:}k+T_{\mathrm{P}}\} : \\ & \mathcal{S}(\boldsymbol{x}_{\mathrm{E},m}) \bigcap \mathcal{S}(\boldsymbol{x}_{\mathrm{PFS},k}) \neq \emptyset, \\ 0, & \text{else,} \end{cases} \tag{6.40}$$

in which $\mathcal{S}(\boldsymbol{x}_{\mathrm{PFS},k})$ describes the set of points within the PFS map that are obstacle boundary delimiters. Note that we only need to integrate the ego vehicle's joint pdf over the specified volume in this case. If the uncertainties in PFS maps with pdf $p(\boldsymbol{x}_{\mathrm{PFS},k})$ shall additionally be taken into account, then (6.39) can be generalized to

$$\begin{aligned} p\big(C_{\mathrm{PFS},k}(T_{\mathrm{P}})\big) = &\int_{\boldsymbol{x}_{\mathrm{PFS},k}} \int_{\boldsymbol{x}_{\mathrm{E},k+T_{\mathrm{P}}}} \cdots \int_{\boldsymbol{x}_{\mathrm{E},k}} I_{C_{\mathrm{PFS}}}(\boldsymbol{x}_{\mathrm{E},k:k+T_{\mathrm{P}}}, \boldsymbol{x}_{\mathrm{PFS},k}) \\ &\qquad p(\boldsymbol{x}_{\mathrm{E},k:k+T_{\mathrm{P}}}, \boldsymbol{x}_{\mathrm{PFS}_k}) \mathrm{d}\boldsymbol{x}_{\mathrm{E},k} \dots \mathrm{d}\boldsymbol{x}_{\mathrm{E},k+T_{\mathrm{P}}} \mathrm{d}\boldsymbol{x}_{\mathrm{PFS},k} \end{aligned} \tag{6.41}$$

with identical indicator function. This indicator function is evaluated by analytically checking whether the predicted ego vehicle's bounding rectangle overlaps with $\mathcal{S}(\boldsymbol{x}_{\mathrm{PFS},k})$. For the collision detection with the tracked

B-Spline curve, this evaluation is performed by checking whether at least one of the line segments

$$\boldsymbol{l}_i(s_\mathrm{l}) = \begin{pmatrix} x_l(s_l) \\ y_l(s_l) \end{pmatrix} = \begin{pmatrix} C_{x,i} \\ C_{y,i} \end{pmatrix} + s_\mathrm{l} \begin{pmatrix} C_{x,i+1} - C_{x,i} \\ C_{y,i+1} - C_{y,i} \end{pmatrix}, \quad \forall s_\mathrm{l} \in [0,1], \quad (6.42)$$

between the ego vehicle's $i \in \{1, \ldots, 4\}$ corner points $C_i = (C_{x,i}, C_{y,i})$ with $C_1 = C_5$ and the B-spline curve $\boldsymbol{r}(s)$ intersect. Considering uniform, quadratic ($d = 3$), periodic B-spline basis functions such as in Chapter 5, therefore $L = N_\mathrm{b} = N_\mathrm{k} - 1$ and $\{k_0, \ldots, k_{N_\mathrm{k}-1}\} = \{0, \ldots, L\}$, then the B-spline curve within span S_m is given by

$$\boldsymbol{r}_m(\Delta s) = \begin{pmatrix} x_m(\Delta s) \\ y_m(\Delta s) \end{pmatrix} = \begin{pmatrix} \boldsymbol{b}_m(\Delta s)^T & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{b}_m(\Delta s)^T \end{pmatrix} \boldsymbol{x}_{\mathrm{P},m}, \quad (6.43)$$

with

$$\boldsymbol{b}_m(\Delta s) = \begin{pmatrix} \frac{1}{2}(\Delta s)^2 - \Delta s + \frac{1}{2} \\ -(\Delta s)^2 + \Delta s + \frac{1}{2} \\ \frac{1}{2}(\Delta s)^2 \end{pmatrix} \in \mathbb{R}^3, \quad \forall \Delta s \in [0,1], \quad (6.44)$$

and $\boldsymbol{x}_{\mathrm{P},m} = (q_{x,m-2}\ q_{x,m-1}\ q_{x,m}\ q_{y,m-2}\ q_{y,m-1}\ q_{y,m})^T \in \mathbb{R}^6$ with $m \in \{0, \ldots, L-1\}$, $\Delta s = s - k_m$ as well as $q_{x,-2} = q_{x,L-2}, q_{y,-2} = q_{y,L-2}$ and $q_{x,-1} = q_{x,L-1}, q_{y,-1} = q_{y,L-1}$ due to the periodicity. For collision checking, the system of quadratic equations

$$\boldsymbol{l}_i(s_\mathrm{l}) \overset{!}{=} \boldsymbol{r}_m(\Delta s), \quad \forall i \in \{1, \ldots, 4\} \ \wedge \ m \in \{0, \ldots, L-1\} \quad (6.45)$$

is solved for real solutions of Δs and s_l, which both have to be inside the interval $[0,1]$ for an occurring collision. Fig. 6.16 visualizes the relations.

The collision detection with the rectangular and circular inner free space boundaries, i.e. the free space holes, is performed analogous to the collision detection with other predicted vehicles.

The advantage of using PFS maps for criticality assessment is that arbitrary static environment structures can be embedded into the criticality assessment part analytically, and the complete static environment can be checked for collisions with future ego vehicle configurations in the same way.

Combination

So far, *static* and *individual dynamic* objects have been treated in isolation. For criticality assessment with respect to the complete traffic scene, the

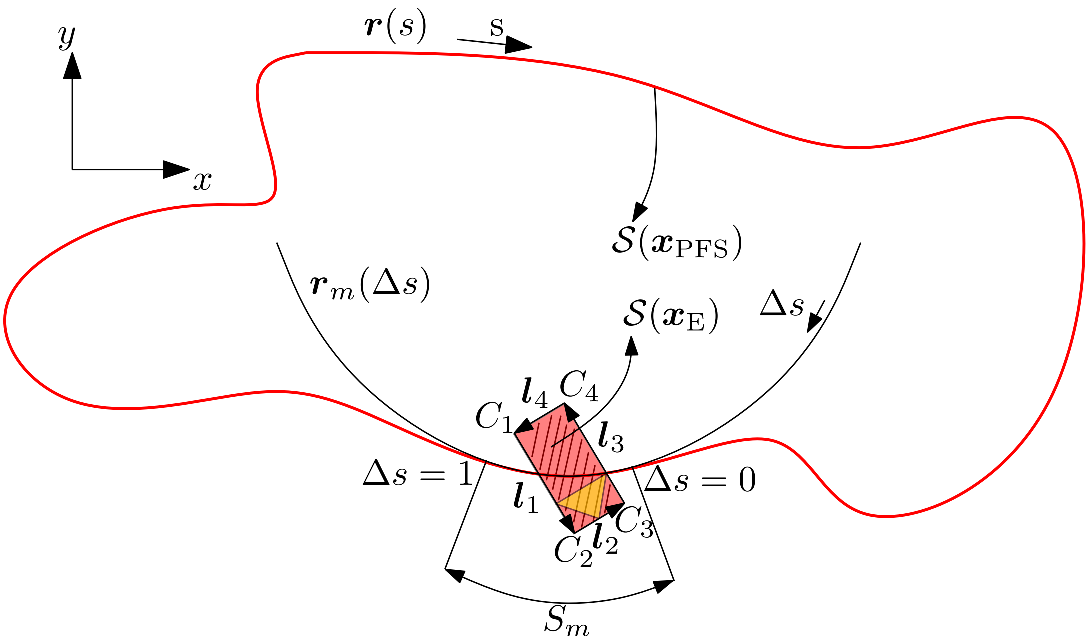


Figure 6.16: Collision check between a B-spline curve and the ego vehicle bounding rectangle.

individual elements need to be combined. Considering all f predicted vehicles and the static environment, the probability of the combined event $C_k(T_\mathrm{P})$: *The ego vehicle collides with at least one vehicle or the static driving environment at least once within the prediction horizon* $\{k{:}k{+}T_P\}$, evaluated at time step k, can be derived as

$$\begin{aligned} p\big(C_k(T_\mathrm{P})\big) &= p\left(\bigcup_{i=1}^{f} C_{\mathrm{V}_i,k}(T_\mathrm{P}) \cup C_{\mathrm{PFS},k}(T_\mathrm{P})\right) \\ &= p\left(\neg\left(\bigcap_{i=1}^{f} \neg C_{\mathrm{V}_i,k}(T_\mathrm{P}) \cap \neg C_{\mathrm{PFS},k}(T_\mathrm{P})\right)\right) \\ &= 1 - p\left(\bigcap_{i=1}^{f} \neg C_{\mathrm{V}_i,k}(T_\mathrm{P}) \cap \neg C_{\mathrm{PFS},k}(T_\mathrm{P})\right) \\ &= 1 - \prod_{i=1}^{f} \Big(1 - p\big(C_{\mathrm{V}_i,k}(T_\mathrm{P})\big)\Big)\Big(1 - p\big(C_{\mathrm{PFS},k}(T_\mathrm{P})\big)\Big), \end{aligned} \tag{6.46}$$

in which the first step follows from de Morgan's laws and the last step from the stochastic independence of individual (not mutually exclusive) collision events. This independence is a consequence of the independence assumption of all vehicle trajectories, so that the probability of the ego vehicle colliding with one object is not influenced by the probability of colliding with another. If this assumption is omitted, $p\big(C_k(T_\mathrm{P})\big)$ must be calculated according to

$$
\begin{aligned}
&p\big(C_k(T_\mathrm{P})\big) = \\
&\int_{\boldsymbol{x}_{\mathrm{PFS},k}} \underbrace{\overbrace{\left(\int_{\boldsymbol{x}_{\mathrm{V}_f,k+T_\mathrm{P}}} \cdots \int_{\boldsymbol{x}_{\mathrm{V}_1,k+T_\mathrm{P}}} \int_{\boldsymbol{x}_{\mathrm{E},k+T_\mathrm{P}}}\right)}^{f+1} \cdots \left(\int_{\boldsymbol{x}_{\mathrm{V}_f,k}} \cdots \int_{\boldsymbol{x}_{\mathrm{V}_1,k}} \int_{\boldsymbol{x}_{\mathrm{E},k}}\right)}_{(T_\mathrm{P}+1)\cdot(f+1)} \\
&I_{C_\mathrm{g}}(\boldsymbol{x}_{\mathrm{E},k:k+T_\mathrm{P}}, \boldsymbol{x}_{\mathrm{V}_1,k:k+T_\mathrm{P}}, \ldots, \boldsymbol{x}_{\mathrm{V}_f,k:k+T_\mathrm{P}}, \boldsymbol{x}_{\mathrm{PFS},k}) \\
&p(\boldsymbol{x}_{\mathrm{E},k:k+T_\mathrm{P}}, \boldsymbol{x}_{\mathrm{V}_1,k:k+T_\mathrm{P}}, \ldots, \boldsymbol{x}_{\mathrm{V}_f,k:k+T_\mathrm{P}}, \boldsymbol{x}_{\mathrm{PFS},k})(\mathrm{d}\boldsymbol{x}_{\mathrm{E},k}\mathrm{d}\boldsymbol{x}_{\mathrm{V}_1,k} \ldots \mathrm{d}\boldsymbol{x}_{\mathrm{V}_f,k}) \\
&\ldots(\mathrm{d}\boldsymbol{x}_{\mathrm{E},k+T_\mathrm{P}}\mathrm{d}\boldsymbol{x}_{\mathrm{V}_1,k+T_\mathrm{P}} \ldots \mathrm{d}\boldsymbol{x}_{\mathrm{V}_f,k+T_\mathrm{P}})\mathrm{d}\boldsymbol{x}_{\mathrm{PFS},k},
\end{aligned}
\tag{6.47}
$$

with general indicator function

$$
\begin{aligned}
&I_{C_\mathrm{g}}(\boldsymbol{x}_{\mathrm{E},k:k+T_\mathrm{P}}, \boldsymbol{x}_{\mathrm{V}_1,k:k+T_\mathrm{P}}, \ldots, \boldsymbol{x}_{\mathrm{V}_f,k:k+T_\mathrm{P}}, \boldsymbol{x}_{\mathrm{PFS},k}) \\
&= \begin{cases} 1, \exists m \in \{k{:}k+T_\mathrm{P}\} \colon \mathcal{S}(\boldsymbol{x}_{\mathrm{E},m}) \bigcap \left(\bigcup_{i=1}^{f} \mathcal{S}(\boldsymbol{x}_{\mathrm{V}_i,m}) \bigcup \mathcal{S}(\boldsymbol{x}_{\mathrm{PFS},k})\right) \neq \emptyset, \\ 0, \text{else}. \end{cases}
\end{aligned}
\tag{6.48}
$$

Criticality Metric TTCCP

We now consider (6.46) as a function of T_P, i.e. the length of the prediction horizon (number of prediction time steps), and define the criticality metric TTCCP as the time until the combined collision probability $p(C_k(T_\mathrm{P}))$ exceeds a *Critical Collision Probability* (CCP), therefore

$$
\mathrm{TTCCP} = \min\big(T_\mathrm{P} \mid p(C_k(T_\mathrm{P})) > \mathrm{CCP}\big) \cdot T, \tag{6.49}
$$

with prediction sampling time T. This can be seen as a generalization of the criticality metric TTC for *arbitrary*, *uncertain*, *multi-object* driving environments. Note that the TTCCP is based on the collision probability over a future *time span* and not on the collision probability of future *time instants*. The former is always a monotonically increasing function over the prediction time, which does not hold for the latter. This is also the reason why a CCP threshold on the collision probability of future time instants is an irrational approach to calculate the TTCCP. This probability can be low in every prediction time step and therefore falsely indicate a safe situation, whereas the collision probability over a time span can reveal an imminent danger by combining the individual criticalities over time. This fact is further illustrated in Fig. 6.17 on an exemplary scenario. Note further that the CCP threshold in (6.49) is a system design parameter, which could be changed for different applications. It can, for example, be lowered for driver warnings even in relatively uncertain, dangerous situations and raised for CAS, in which an autonomous evasive action should only be triggered if there exists a very certain imminent danger.

Monte Carlo Simulation

In a practical implementation, (6.35) and (6.39) within (6.46) are solved via Monte Carlo simulation. Therefore, the expectation of the indicator function in (6.35), i.e. the collision probability, can be estimated by

$$p(C_{\mathrm{V}_i,k}(T_{\mathrm{P}})) = \mathbb{E}(I_C) \approx \frac{1}{N} \sum_{i=1}^{N} I_C(\tilde{\boldsymbol{x}}_i), \tag{6.50}$$

in which $\tilde{\boldsymbol{x}}_i$ is one out of N independent and identically distributed drawn samples from the joint density $p(\boldsymbol{x}_{\mathrm{E},k:k+T_{\mathrm{P}}}, \boldsymbol{x}_{\mathrm{V}_i,k:k+T_{\mathrm{P}}})$ and the unbiased estimate converges almost surely to the true collision probability by the strong law of large numbers with convergence rate of order $\mathcal{O}(\frac{1}{\sqrt{N}})$ [94]. Each sample therefore corresponds to two complete predicted trajectories. Although, following (6.37), the trajectories of the vehicles can be drawn independently of each other, the specification of the Monte Carlo estimate in this more general form is advantageous as it then becomes directly obvious that only a single summation over simultaneously drawn samples from both individual distributions, $p(\boldsymbol{x}_{\mathrm{E},k:k+T_{\mathrm{P}}})$ and $p(\boldsymbol{x}_{\mathrm{V}_i,k:k+T_{\mathrm{P}}})$, is required. This approach is termed Fast Monte Carlo in [138] and further elaborated and justified in [71]. This way, much fewer samples are needed to obtain the same estimation error compared to a naive double summation over individual trajectory samples. Another performance gain is achieved by realizing that samples for the ego vehicle, drawn from $p(\boldsymbol{x}_{\mathrm{E},k:k+T_{\mathrm{P}}})$, can be reused in the evaluation of (6.35) for the different vehicles and in (6.39) for the collision probability with the static environment due to the mentioned independence assumption between vehicle trajectories. Moreover, trajectory samples do not have to be generated for the whole prediction horizon $\{k{:}k+T_{\mathrm{P}}\}$ but only until the first collision is detected because the

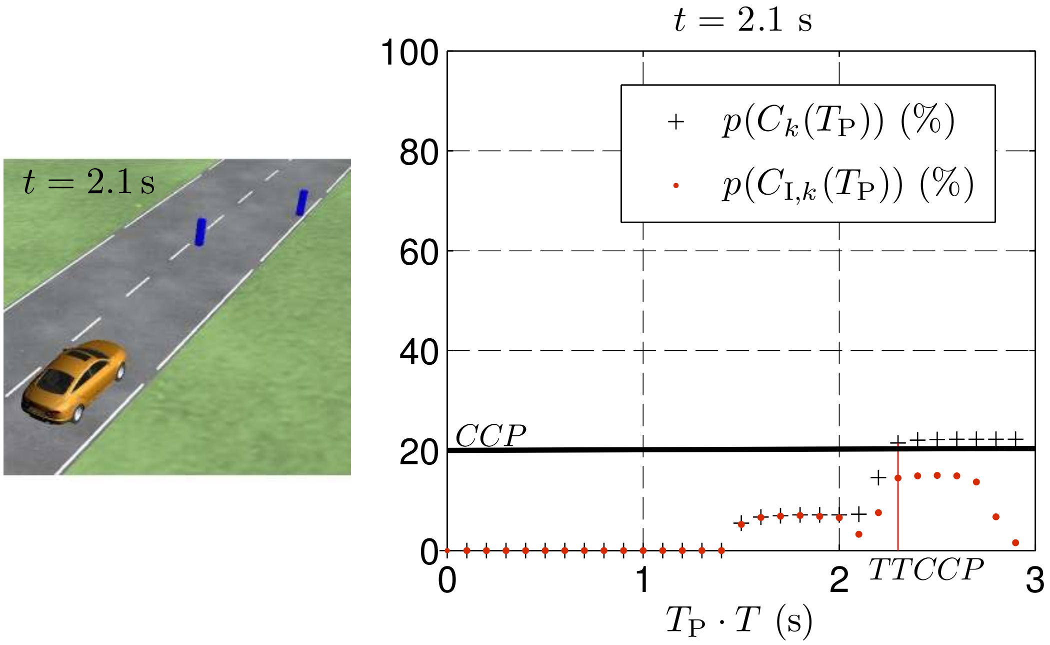


Figure 6.17: Difference between collision probability of future time instants and future time spans. We consider a vehicle moving with constant velocity on a lane with two obstacles (blue) standing left and right in such a way that the vehicle can just get through by moving straight. The obstacles have an additional longitudinal offset slightly larger than the vehicle length. We first have a look at the collision probabilities at future *time instants*, $p\big(C_{\mathrm{I},k}(T_\mathrm{P})\big)$, which are shown in red for a prediction horizon of 3 s in the right diagram. Two plateaus are visible, which stem from predicted collisions with the two obstacles under consideration of vehicle and obstacle extensions. The plateau from the obstacle further away is higher due to the larger variance of the predicted position at greater prediction times. This results in a larger amount of predicted vehicle configurations overlapping with the obstacle. For predicted time instants in-between the plateaus ($T_\mathrm{P} \cdot T \approx 2.1\,\mathrm{s}$), the collision probability decreases as the predicted location is now mostly right between the longitudinal positions of both obstacles and thus no collision occurs. In every predicted time step, $p\big(C_{\mathrm{I},k}(T_\mathrm{P})\big)$ is lower than the CCP fixed at 20 %. Note that the maximal probability over *time instants* would be the same if either only one or both obstacles were present (because simultaneous collisions with both obstacles at a single future time step are impossible in this scenario), which contradicts intuition. In contrast, we now have a look at the collision probability over a future *time span*, $p\big(C_k(T_\mathrm{P})\big)$, shown in black. Its shape is similar to $p\big(C_{\mathrm{I},k}(T_\mathrm{P})\big)$ at first, but does not decrease as all trajectories that have already collided at least once with the first obstacle are still considered. For longer prediction time horizons, the second obstacle now raises the probability $p\big(C_k(T_\mathrm{P})\big)$ above CCP and the situation is regarded as critical with a TTCCP of 2.3 s.

combined collision event probability does (intentionally) not change in the case of subsequent collisions.

All in all, the run-time for the criticality assessment scales linearly with the number of dynamic objects and is – due to the PFS map approach – nearly independent of the complexity of the static environment. In this regard, it only depends on the (normally small) number of geometric primitives contained in the PFS map, and on the number of spline spans that are delimiters to obstacle boundaries.

In the following, results of the proposed criticality assessment system are shown in exemplary simulated driving scenarios.

6.4 Simulation Environment and Results

The complete criticality assessment system has been implemented in C++ and coupled with the simulation environment IPG Carmaker.[20] The developed simulation extensions include the full replication of the environment representation (including occupancy grid mapping, PFS mapping) as a foundation, the BN-based inference for maneuver detection via the junction tree algorithm [143], the probabilistic, maneuver-based predictions, and the criticality assessment via Monte Carlo simulation.

The system has been parametrized by constructing manifold driving situations within the simulation environment. The final parameter set used in all simulations is summarized in Table 6.3, the CPTs of the BN can be found in appendix A. The chosen time-discretization of $T = 0.1\,\mathrm{s}$ allows collision detections between two standard-sized vehicles of length 4.7 m up to a relative velocity of 340 km/h or collisions with a point object up to a velocity of 170 km/h. The criticality assessment itself is likewise performed every 0.1 s and should optimally be executed directly after an environment representation update.[21] The maximum length of the prediction horizon is set to 3 s, which is considered sufficient for timely driver warnings and interventions.

In the following, exemplary driving scenarios are shown to illustrate the capabilities and scope of the designed system. Each of the scenarios emphasizes different aspects and contains a variety of sub-scenarios and maneuvers in different dynamic driving environments.

[20] Thanks to Igor Achieser, Yuanwen Qin, Julien Seitz, and Gerrit Wege for their help in implementing the concept in their theses [1, 202, 230, 266].

[21] Results are shown only every 300 ms or 600 ms for clarity.

Table 6.3: Prediction and criticality assessment parameters.

Model	Parameter	Value
FR	$3\sigma_{\Delta a_{\mathrm{R,lon}}}; 3\sigma_{y_{\mathrm{R}}}; T_{\mathrm{c}}; u; 3\sigma_{\psi_{\mathrm{R}}}$	$0.2\,\mathrm{m/s^2}; \frac{1}{2}(w_{\mathrm{L}}-w_{\mathrm{V}}); 1.5\,\mathrm{s}; \frac{1}{2}w_{\mathrm{L}}; 5\,^{\circ}$
FV	$\tau_{\mathrm{r}}; a_{\mathrm{f,min}}; a_{\mathrm{f,max}}; 3\sigma_{\Delta a_{\mathrm{R,lon}}};$ $3\sigma_{y_{\mathrm{R}}}; T_{\mathrm{c}}; u; 3\sigma_{\psi_{\mathrm{R}}}$	$2\,\mathrm{s}; -3.5\,\mathrm{m/s^2}; 2.5\,\mathrm{m/s^2}; 0.2\,\mathrm{m/s^2};$ $\frac{1}{2}(w_{\mathrm{L}}-w_{\mathrm{V}}); 1.5\,\mathrm{s}; \frac{1}{2}w_{\mathrm{L}}; 5\,^{\circ}$
TB	$d_{\mathrm{r}}; a_{\mathrm{min}}; 3\sigma_{y_{\mathrm{R}}};$ $T_{\mathrm{c}}; u; 3\sigma_{\psi_{\mathrm{R}}}$	$1\,\mathrm{m}; -8\,\mathrm{m/s^2}; \frac{1}{2}(w_{\mathrm{L}}-w_{\mathrm{V}});$ $1.5\,\mathrm{s}; \frac{1}{2}w_{\mathrm{L}}; 5\,^{\circ}$
LC	$3\sigma_{\Delta a_{\mathrm{R,lon}}}; 3\sigma_{y_{\mathrm{R,s}}}$	$0.2\,\mathrm{m/s^2}; \frac{1}{2}(w_{\mathrm{L}}-w_{\mathrm{V}})$
TU	$a_{\mathrm{min}}; a_{\mathrm{c,max}}; 3\sigma_{\Delta a_{\mathrm{R,lon}}};$ $3\sigma_{y_{\mathrm{R}}}; T_{\mathrm{c}}; u; 3\sigma_{\psi_{\mathrm{R}}}$	$-8\,\mathrm{m/s^2}; 4\,\mathrm{m/s^2}; 0.2\,\mathrm{m/s^2};$ $\frac{1}{2}(w_{\mathrm{L}}-w_{\mathrm{V}}); 1.5\,\mathrm{s}; \frac{1}{2}w_{\mathrm{L}}; 5\,^{\circ}$
TR	$3\sigma_{\Delta a}; 3\sigma_{\Delta\omega}$	$0.2\,\mathrm{m/s^2}; 1.5\,^{\circ}/\mathrm{s}$
PFS	$N_{\mathrm{b}}; d$	$70; 3$
—	$w_{\mathrm{L}}; w_{\mathrm{V}}; T; N; \mathrm{CCP}$	$3.5\,\mathrm{m}; 1.5\,\mathrm{m}; 0.1\,\mathrm{s}; 5000; 0.2$

6.4.1 Dangerous Lane Change Scenario

In the first example visualized in Fig. 6.18(a,b,c), the ego vehicle (orange) approaches another vehicle (blue) driving in the same direction on the same lane, decelerates rapidly ($t \approx 4\,\mathrm{s}$) to successfully prevent a collision, and subsequently performs a lane change ($t \approx 6\,\mathrm{s}$) to the left to overtake. On the adjacent lane, however, an oncoming vehicle (red) approaches with which the ego vehicle finally collides at $t = 7.6\,\mathrm{s}$.

The maneuver estimation (Fig. 6.18d) clearly shows the individual driving maneuvers for the ego vehicle – the other two vehicles just follow the road with constant velocity. First, high probabilities (≈ 0.8) are assigned to the *follow vehicle* maneuver as the ego vehicle approaches the other vehicle on the same lane. The remaining probability mass is nearly fully allocated to the *follow road* maneuver, which is reasonable as the evidence does not fully support the *follow vehicle* maneuver class due to the existing velocity difference. The other maneuver probabilities correctly remain close to zero. As soon as the ego vehicle brakes, the *target brake* probability rises. When the braking is finished at $t \approx 5\,\mathrm{s}$, the probability declines to zero again and the *follow vehicle* maneuver again becomes dominant before the lane change. Subsequently, the executed lane change is detected early on, resulting in high *lane change* maneuver probabilities at $t \approx 6\,\mathrm{s}$. Once finished, the *follow road* maneuver probability rises again, but now relates to the left lane, before the collision with the oncoming vehicle takes place.

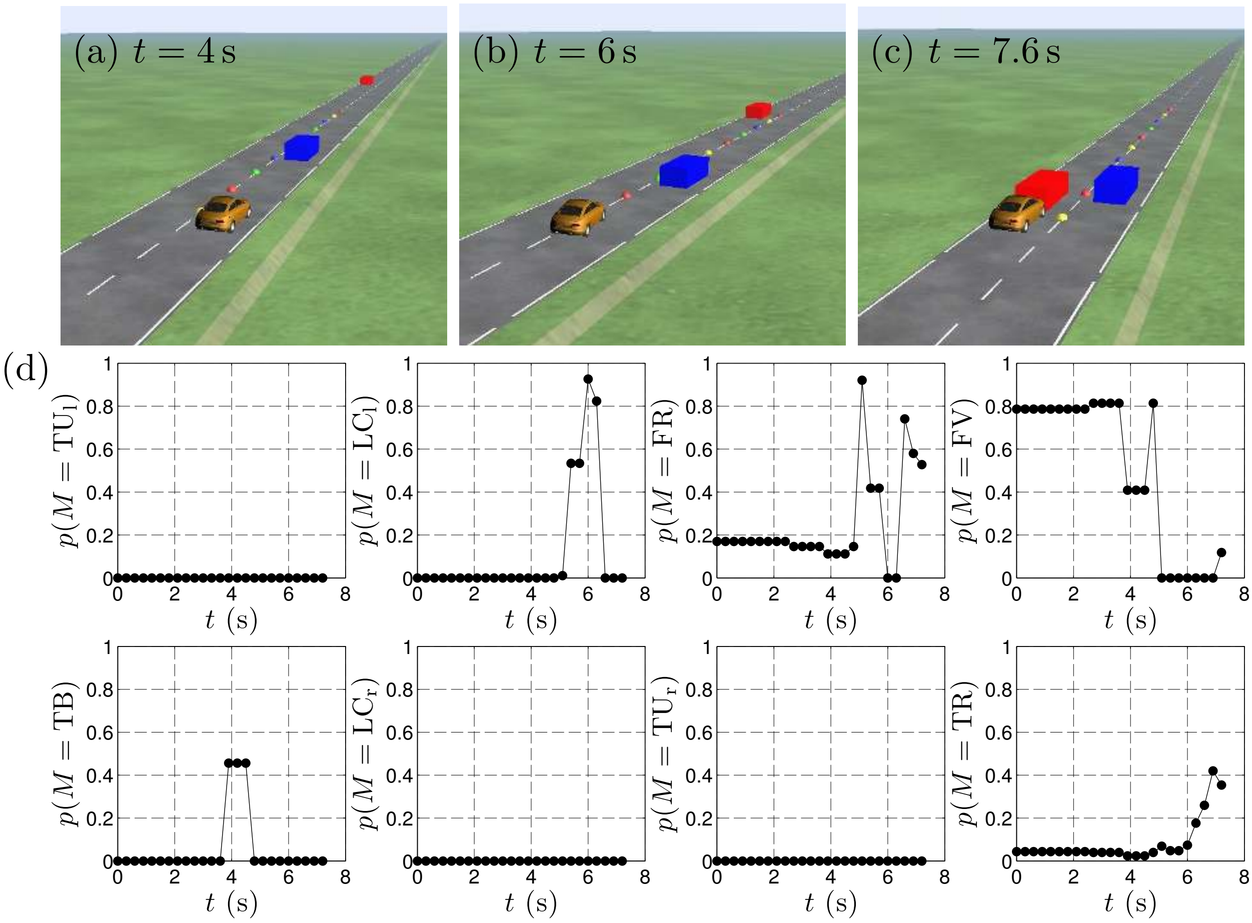


Figure 6.18: Lane change scenario and collision with oncoming vehicle. The traffic scene at different simulation times is shown in (a,b,c), inferred maneuver probabilities in (d).

Fig. 6.19 additionally illustrates the scene from a bird's-eye view and visualizes predicted trajectory samples of the proposed approach (Fig. 6.19a) with corresponding combined collision probabilities over the future 3 s time horizons in contrast to deterministic CTRA (Fig. 6.19b) and CV (Fig. 6.19c) models.[22] The ego vehicle and the two other vehicles' centered rear points are shown with blue and green circles, respectively. It becomes clearly visible that the deterministic prediction models, which do not consider environment knowledge, cannot reasonably perform longer-term predictions as their predicted trajectories are outside the road after a short prediction time. The derived criticality metrics TTC (CV/CTRA)

[22] Combined collision probabilities are only visualized if they exceed 5 %. Deterministic CTRA and CV predictions only allow binary collision decisions as no uncertainties are considered. Therefore, the shown collision probabilities in Fig. 6.19(b,c) are only either 100 % or zero (not shown).

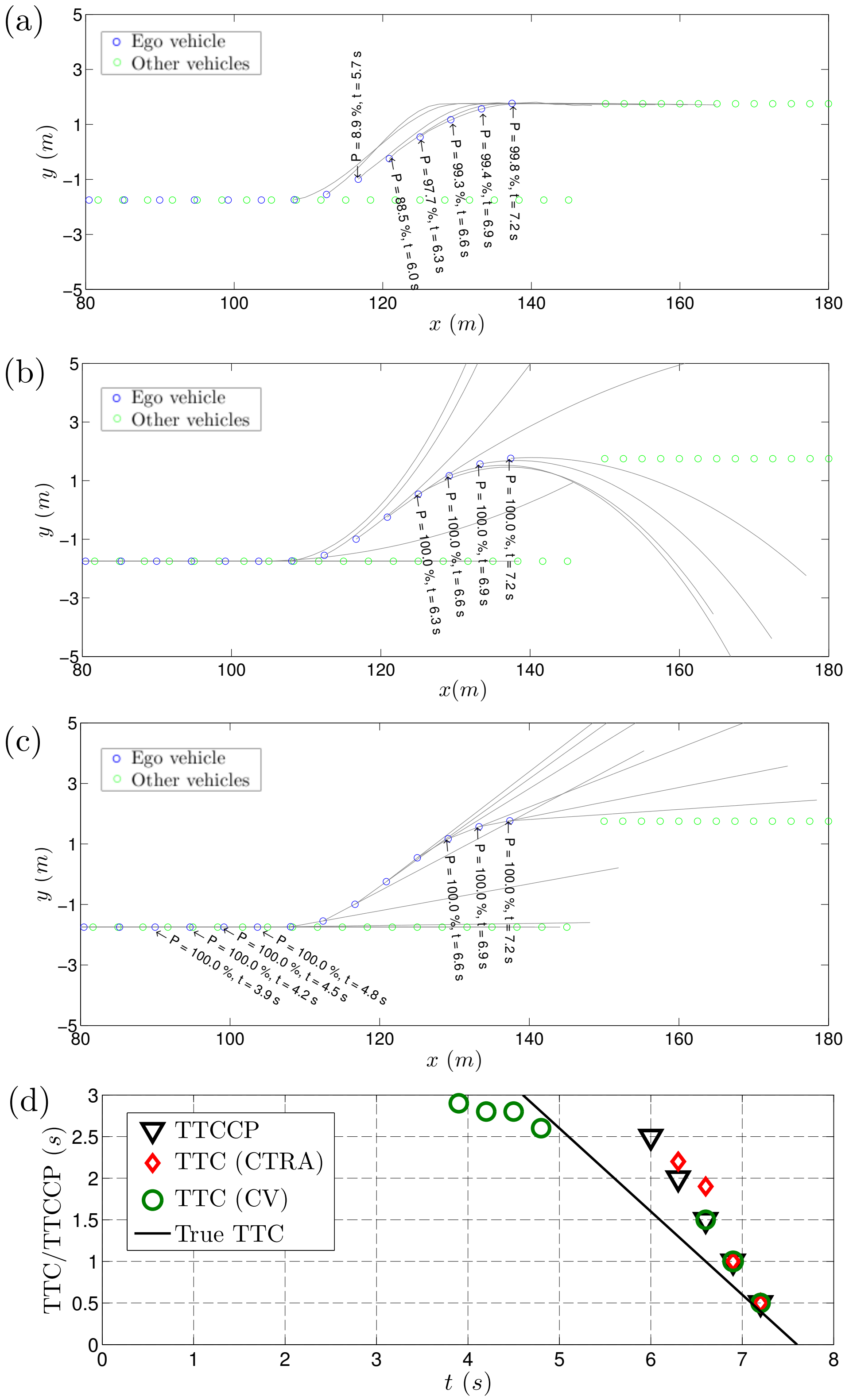


Figure 6.19: Bird's-eye view with predicted trajectories of the proposed approach (only mean lane change predictions are shown for clarity) (a) in contrast to CTRA (b) and CV (c) model predictions with corresponding combined collision probabilities for 3 s prediction horizons and TTC/TTCCP evolution (d).

and TTCCP are plotted in Fig. 6.19d over the simulation scenario time. The TTC calculated via a CV model generates false positive criticality values at $t \approx 4\,\mathrm{s}$ as it does not consider the deceleration of the ego vehicle before performing the lane change. The CTRA-based TTC and the proposed approach do not show this undesirable result. At this stage, the maneuver-based approach mainly predicts via *target brake* and – to a lesser extend – via a *follow vehicle* model. Neither of the two generates colliding trajectories in this situation. Additionally, we see that the true collision can be detected first by the proposed method as the prediction is performed mainly via the *lane change* model during the lane change. Note that the first two TTC (CTRA) values stem from another false collision detection of the ego vehicle with the blue vehicle on the same lane as the predicted trajectories with high turn rate cross the original right lane again – a clearly non-reasonable prediction. With this in mind, the proposed system is able to generate a reasonable criticality value 0.9 s earlier than systems relying on a CTRA-based TTC and still 0.6 s earlier than systems with a CV-based TTC calculation in this scenario. This allows timelier driver warnings – 1.6 s before the imminent collision in this case. Note that the true TTC progression as an optimal result is impossible to match for any prediction method here as, for example, 3 s in advance (at $t = 4.6\,\mathrm{s}$), absolutely no evidence is available that would imply an impending lane change. We further realize that the TTCCP metric reasonably converges to zero the closer we get to the true collision time $t = 7.6\,\mathrm{s}$. Deviations between TTCCP and true TTC before the collision event stem from a discrepancy between the lane change prediction model and the true, executed lane change trajectory, which can never be known beforehand.

6.4.2 Near-Collision Turn Scenario

The second scenario is visualized in Fig. 6.20(a,b,c). Therein, the ego vehicle (orange) approaches an intersection while another crossing vehicle (red) appears from the left. Both vehicles come dangerously close to each other at the intersection and do only barely not collide. The ego vehicle then turns to the left ($t \approx 10\,\mathrm{s}$), follows the lane again after the turn, and finally collides with constant velocity with a standing obstacle at $t = 17.5\,\mathrm{s}$.

We first have a look at the maneuver detection for the ego vehicle shown in Fig. 6.20d. As the vehicle follows the lane at the beginning of the scenario, high probabilities are correctly assigned to the *follow road* maneuver. As soon as diagnostic motion evidence for a lateral motion to

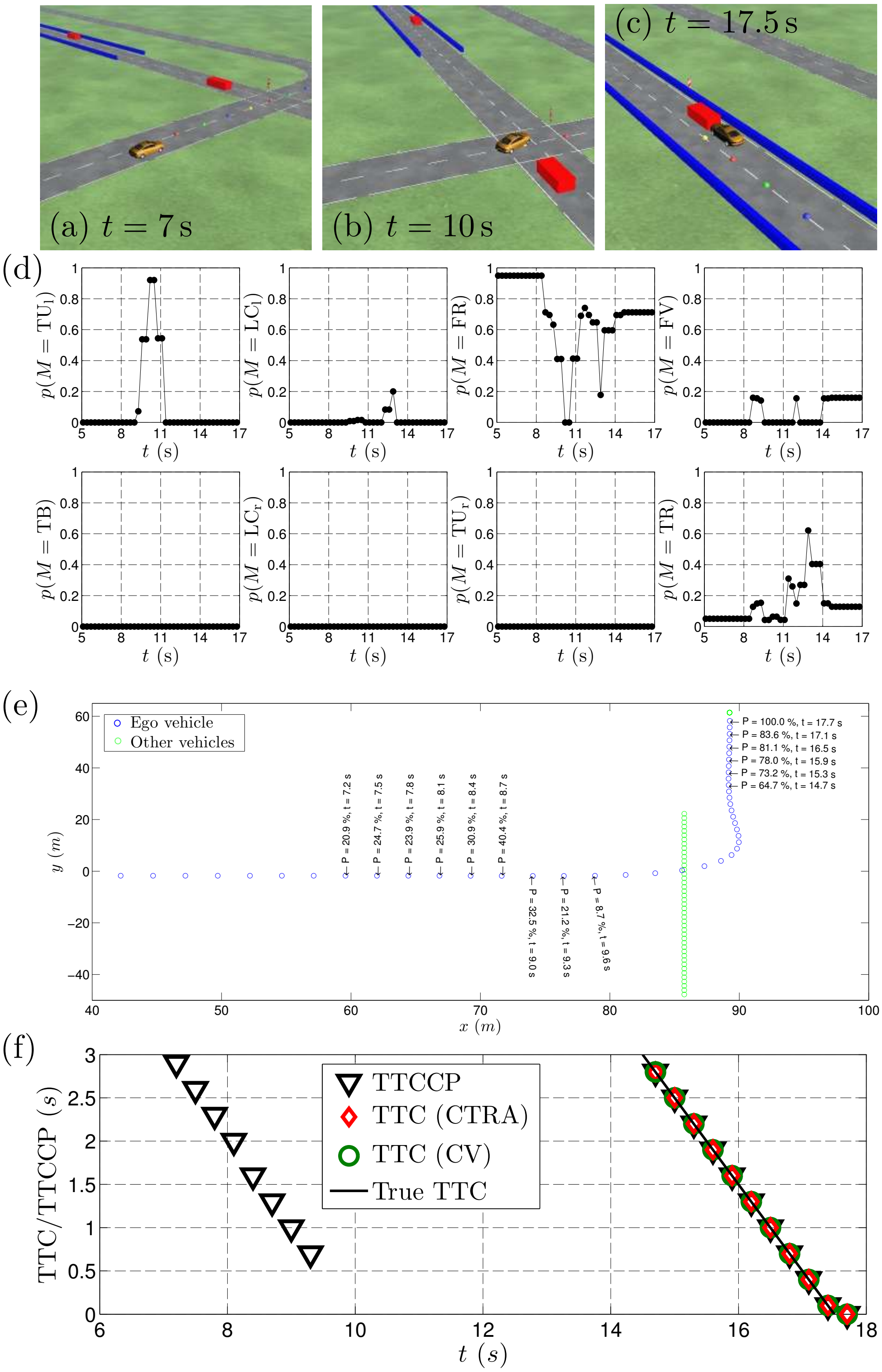


Figure 6.20: Turn scenario (a) with near-collision with cross traffic (b) and subsequent collision with stationary vehicle (c). Maneuver probabilities (d) are shown together with a bird's-eye view with corresponding combined collision probabilities for 3 s prediction horizons (e) and the TTC/TTCCP evolution (f).

the left is available, the *turn left* maneuver is recognized. The probabilities for a lane change to the left remain low during this phase, which is a result of the intercausal reasoning induced by the BN structure. The measurable physical lateral motion evidences could, in principle, also be the consequence of a lane change maneuver but the causal knowledge of an existing turning drastically reduces the probability of a lane change in this situation. This is in accordance with human reasoning, especially as lane changes are normally restricted at intersections. The peak of the *trash* maneuver class after the turn at $t \approx 13\,\mathrm{s}$ stems from a vehicle motion further to the right than necessary to follow the road again, which could also imply that the driver accidentally leaves the road, see also the vehicle path in Fig. 6.20e. The inference result is therefore also understandable in this situation. As soon as the vehicle indeed follows the road again, this ambiguity is resolved and the *follow road* maneuver is clearly dominant again.

Studying Fig. 6.20f, we notice that both the CTRA-based and the CV-based TTC do not provide any indication of the dangerous situation during the intersection approach although every minor acceleration change could lead to a collision. Since the deterministic TTC metric considers current accelerations or velocities as exactly constant over the prediction horizon, there is no predicted collision and therefore no reasonable TTC. Thus, the TTC cannot be used to warn the driver or trigger interventions in this critical near-collision scenario. The TTCCP, in opposition, clearly indicates the danger several seconds ($3\,\mathrm{s}$) in advance by predicting both vehicles (mainly) according to a *follow road* model, in which the future acceleration realizations are considered uncertain. Consequently, a noticeable, non-zero combined collision probability exists as can also be seen in the bird's-eye view in Fig. 6.20e, reflecting the criticality. Eventual alarms triggered by low TTCCP values at $t \approx 9\,\mathrm{s}$ should not be considered false alarms because all available knowledge at this point in time implies a critical situation, which is only resolved when the situation evolves further. As soon as it becomes evident that most probably no collision occurs, the TTCCP vanishes (rises to ∞) at $t = 9.6\,\mathrm{s}$. After the turning, the ego vehicle approaches the standing obstacle with constant velocity and collides at $t = 17.5\,\mathrm{s}$. In this unambiguous case, the TTCCP is identical to the TTC. This is reasonable because the TTCCP is designed as a generalization of the TTC for ambiguous, uncertain environments. In obvious situations like this, both metrics should coincide.

Fig. 6.21 additionally shows the combined collision probabilities $p\big(C_k(T_\mathrm{P})\big)$, evaluated via (6.46), p.173, together with (6.35) and (6.39), as a function of the prediction horizon length $T_{\mathrm{P}} \cdot T$ (in seconds) at two specific simulation times for better understanding how the TTCCP values arise. The TTCCP corresponds to the length of the prediction horizon, at which $p\big(C_k(T_{\mathrm{P}})\big)$ first exceeds the CCP, fixed at 20 % in our case. Fig. 6.21a illustrates the result for the simulation time $t = 8.4\,\mathrm{s}$, shortly before the near-collision at the intersection. At this time, a TTCCP of 1.6 s is estimated, see also the corresponding plot in Fig. 6.20f. This corresponds to a high estimated collision probability within the next 1.6 s, therefore within the time span $t \in [8.4\,\mathrm{s}, 10\,\mathrm{s}]$, which is a reasonable result as the ego vehicle indeed reaches the intersection at $t \approx 10\,\mathrm{s}$, see also Fig. 6.20b. The increase of $p\big(C_k(T_{\mathrm{P}})\big)$ is not completely abrupt due to the uncertainties evolved in the *follow road* models, which are predominantly used for prediction at this time for both vehicles. Moreover, $p\big(C_k(T_{\mathrm{P}})\big)$ only rises to a maximum of approximately 31 % within the complete, 3 s prediction horizon as also reflected in the bird's-eye view visualization in Fig. 6.20e. This is because a collision is not certain as it would only happen if either the ego vehicle accelerated and/or the other vehicle decelerated slightly in this scenario.

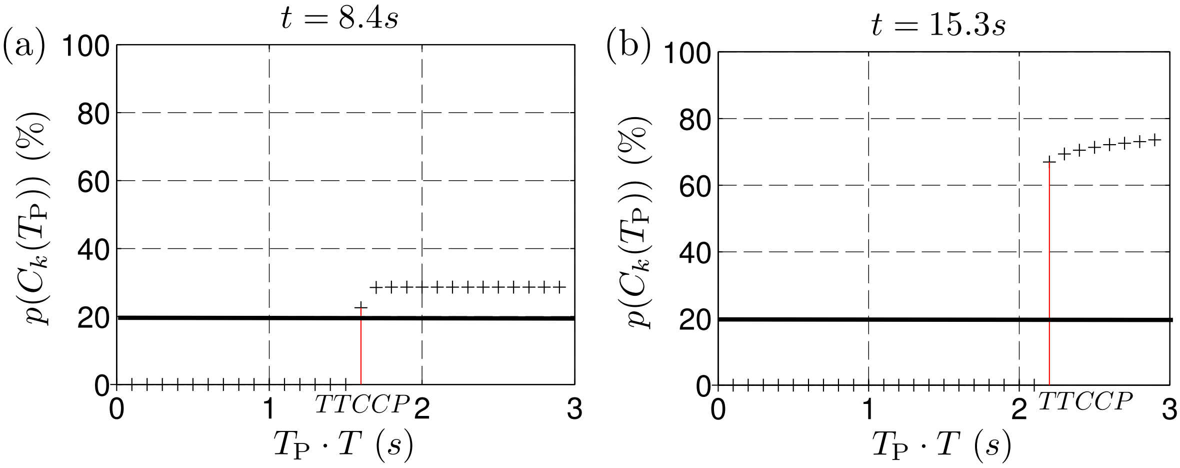


Figure 6.21: Combined collision probabilities over prediction horizon length for the simulation times $t = 8.4\,\mathrm{s}$ (a) and $t = 15.3\,\mathrm{s}$ (b) with resulting TTCCPs for a CCP of 20 %.

Fig. 6.21b, in contrast, shows the situation at the simulation time $t = 15.3\,\mathrm{s}$, shortly before the final collision. The estimated TTCCP lies at 2.2 s, which reflects the true collision time at $t = 17.5\,\mathrm{s}$ correctly, see Fig. 6.20c. Moreover, the combined collision probabilities rise more strongly here because the system is more certain about a future collision as the majority of the predicted ego vehicle configurations overlap with the standing obstacle. The final value of $p\big(C_k(T_\mathrm{P})\big)$ over the 3 s prediction horizon reaches approximately 73 % at this time. This can also be seen in Fig. 6.20e, where it becomes additionally obvious that this value reasonably rises to 100 % several hundred milliseconds before the true collision time because all predicted trajectories finally result in a collision.

As already noted, the CCP value is a system design parameter. It can, for example, be lowered for mere driver warnings even in relatively uncertain dangerous situations, such as in the shown near-collision scenario, and raised for CAS in which an autonomous evasive action shall only be triggered if there exists a very certain imminent danger, such as in the shown unambiguous collision scenario in the end.

6.4.3 Static Environment Collision Scenario

The third scenario, visualized in Fig. 6.22(a,b,c), is devoted to the advantages of the trash maneuver class and the incorporation of PFS maps for criticality assessment. At first, the ego vehicle (orange) follows the road flawlessly. At $t \approx 2.7$ s, however, it deviates from the straight road course to the right, e.g. due to driver inattentiveness, and is near a collision with the right guardrail. Before colliding, the vehicle gets back on track and starts executing a lane change to the left shortly afterwards at $t \approx 7$ s. This lane change is, however, not completed but aborted halfway through at $t \approx 10$ s, and the vehicle moves in-between both lanes for the rest of the scenario. At $t \approx 15$ s, it reaches a narrowing of the road, e.g. a construction side, which gets even more pronounced as soon as also the left lane becomes partially blocked, see Fig. 6.22c. Finally, the vehicle inevitably collides with the road block at $t = 21.9$ s.

At the beginning, the ego vehicle motion along the road is explained by the *follow road* model as shown in Fig. 6.22d. As soon as the vehicle departs from the road course to the right, the motion cannot be reasonably explained by the standard maneuver classes any more. As a consequence, the maneuver detection shows a rise in the *trash* maneuver class probability to nearly 40 % and the abnormal situation becomes evident. Without the trash class, the system would distribute the probability mass between the standard maneuvers, which is clearly inappropriate in this case and would prevent the detection of this critical situation. Since the deviation from the road is not extreme, there is still a relatively high probability of a *follow road* maneuver during this time, which rises back to the high initial value as soon as the driver avoids the collision with the right guardrail and gets back on track at $t \approx 5$ s. The subsequent start of the *lane change*

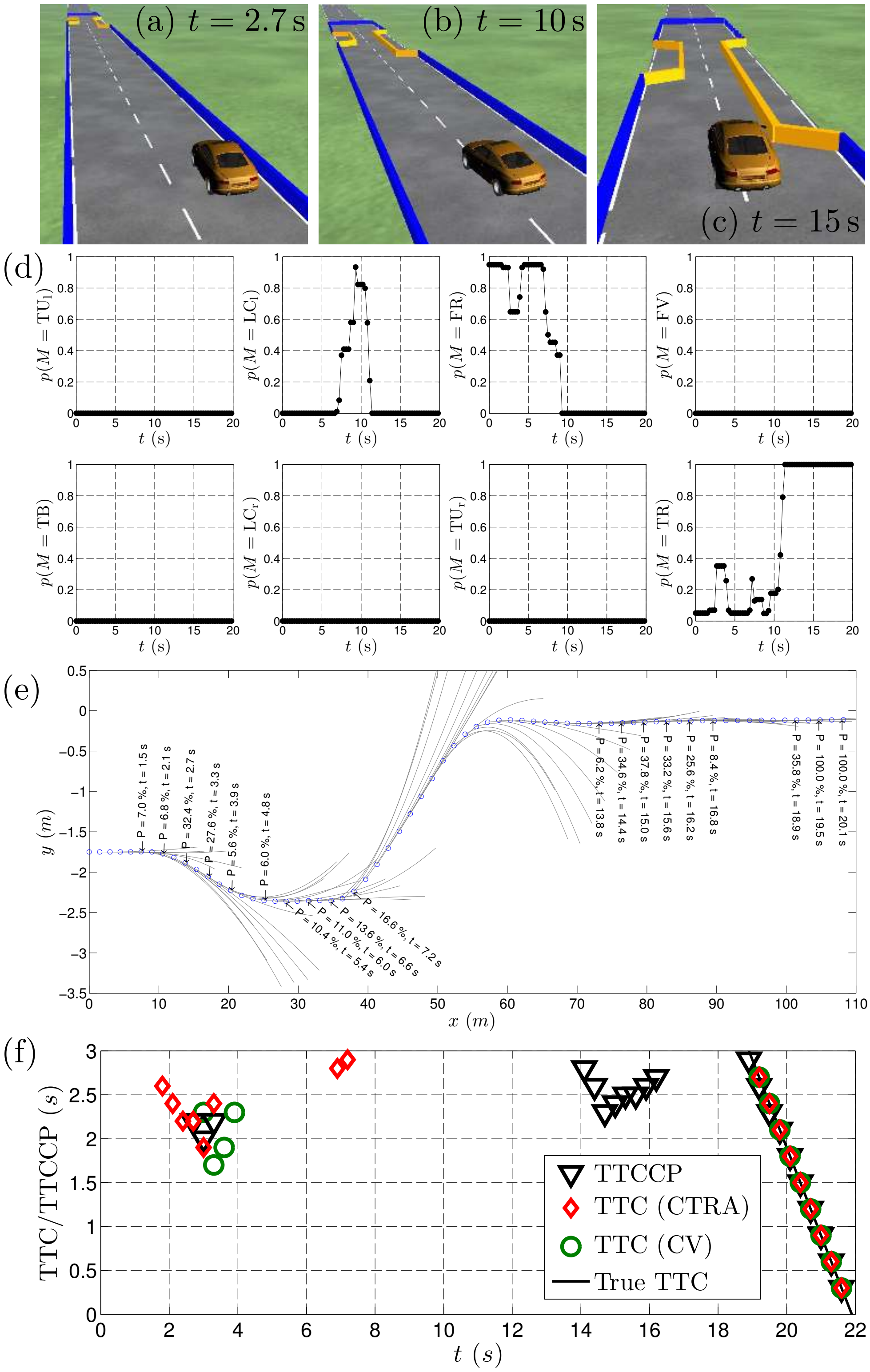


Figure 6.22: Deviation from the road course with near-collision (a) followed by a lane change, which is aborted halfway through (b), and subsequent drive through a narrowing with final collision with road blocking (c). Maneuver probabilities (d) are shown together with a bird's-eye view with combined collision probabilities for 3 s prediction horizons (e) and the TTC/TTCCP evolution (f).

maneuver is then clearly recognized similarly to the first scenario, but as soon as the driver aborts the lane change at $t \approx 10\,\mathrm{s}$ and drives in-between both lanes, the *trash* maneuver class probability rises rapidly and remains high for the rest of the scenario. This is a reasonable inference result because no standard maneuver corresponds to this driver action.

Fig. 6.22e additionally shows the bird's-eye view of the vehicle position along with the mean predicted CTRA trajectories of the trash maneuver class and the corresponding combined collision probabilities over the corresponding 3 s prediction horizons. As long as the driver follows the road at the beginning, the criticality remains near zero due to the bounded variance of the lateral coordinate's Ornstein-Uhlenbeck model so that only a negligible number of predicted trajectories collide with the guardrail. This advantageous property eliminates false warnings in this standard situation. As soon as the vehicle deviates to the right, however, the collision probability rises due to the trash maneuver class's CTRA model predictions, which allows driver warnings or interventions. For better insight, Fig. 6.23a additionally visualizes the combined collision probability as a function of the prediction length at that time ($t = 2.7\,\mathrm{s}$). The consistent rise in collision probability over the prediction horizon exceeds the CCP of 20 % at TTCCP = 2.2 s in the future, which is also reflected in the TTCCP evolution in Fig. 6.22f. The TTCCP around this time is, in value, similar to the deterministic TTC (CTRA) but emerges slightly afterwards because the *trash* maneuver probability has to build up first. Note that the TTC (CV) values are delayed in comparison to TTC (CTRA) and

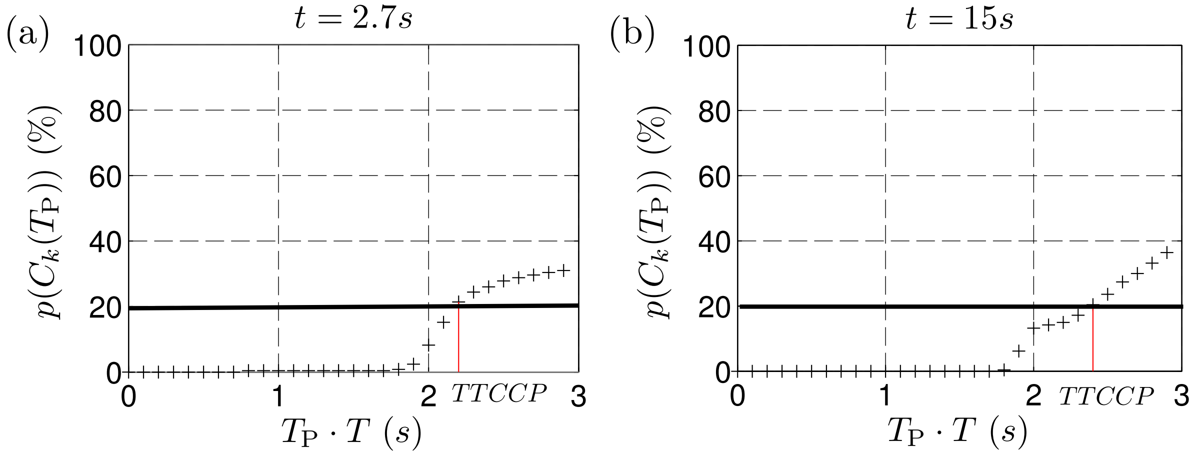


Figure 6.23: Combined collision probabilities over prediction horizon length for the simulation times $t = 2.7\,\mathrm{s}$ (a) and $t = 15\,\mathrm{s}$ (b) with resulting TTCCPs for a CCP of 20 %.

TTCCP as no turn rates are considered in the CV model. Therefore, both the rising criticality and the subsequent collision avoidance by the driver's countersteer are reflected too late in the TTC (CV). At the subsequent start of the lane change, the TTC (CTRA) generates false values at $t \approx 7\,\mathrm{s}$ as the CTRA predictions collide with the left guardrail due to the strong yaw rate at the beginning of the lane change. These false criticality values are suppressed by the proposed approach because of the predominant prediction via the *lane change* model at this time.

Between $t \approx 15\,\mathrm{s}$ and $t \approx 18\,\mathrm{s}$, the vehicle drives through the already mentioned road narrowing. The PFS maps overlaid their corresponding occupancy grid maps (occupied shown in blue, free in green) are additionally depicted in Fig. 6.24(a,b) for these times. The narrowing is clearly visible in the map representations and the situation is considered critical as any minor steering would result in a collision. The TTCCP can represent this criticality, whereas a reasonable TTC does not exist (remains at ∞). A closer look at the collision probability evolution at $t = 15\,\mathrm{s}$ in Fig. 6.23b reveals a first increase of $p\big(C_k(T_\mathrm{P})\big)$ approximately 1.8 s in the future, followed by a plateau and a further increase after 2.3 s prediction time. This evolution is the result of the two subsequent narrowings, first only on the right, later also on the left. The first narrowing does not suffice to exceed the CCP but only the second one provokes a situation criticality high enough to generate a TTCCP value of 2.4 s. This result is in accordance with human reasoning in this situation, see Fig. 6.22c. Note that the TTCCP values remain approximately constant during the narrow road passage because the criticality does not change as long as there is no steering against the boundary. At the end of the narrowing, the TTCCP

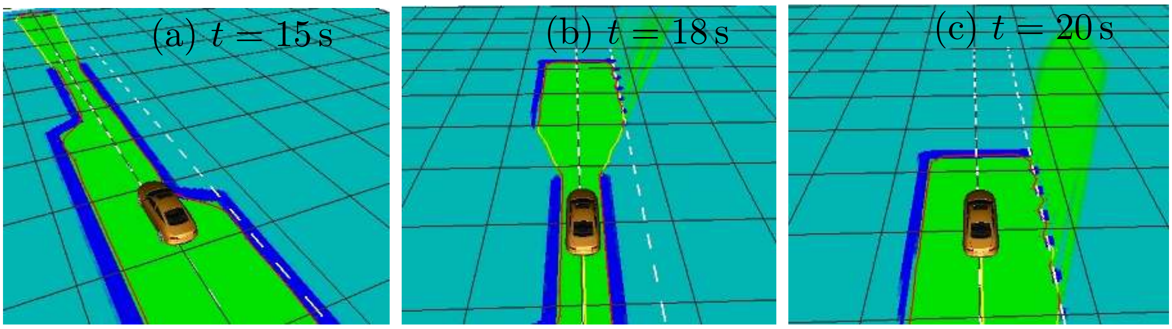


Figure 6.24: PFS maps overlaid corresponding occupancy grid maps for simulation times $t = 15\,\mathrm{s}$ (a), $t = 18\,\mathrm{s}$ (b), and $t = 20\,\mathrm{s}$ (c).

disappears (rises to ∞)[23], see Fig. 6.22f, before finally converging towards the true collision time with the road block at $t = 21.9\,\mathrm{s}$, which is apparent also in the PFS map in Fig. 6.24c. The combined collision probability rises to 100 % here, as all *trash* maneuver class's trajectories collide with the outer B-spline curve at the end of the scenario due to the completely blocked road without any driver reaction.

In summary, the third scenario underlined the value of the additional trash maneuver class for detecting abnormal situations as well as the beneficial embedding of the PFS map-based environment representation into the criticality assessment system.

6.5 Conclusion

An integrated approach to long-term trajectory prediction and criticality assessment has been presented, which considers maneuver and motion uncertainties of all vehicles within the traffic scene as well as arbitrary static environments described by PFS maps.

In a first step, a novel Bayesian network has been designed for the detection of the driving maneuvers *lane change left*, *turn left*, *follow vehicle*, *follow road*, *target brake*, *turn right*, and *lane change right*. The main contributions in the network design process are

- the three layer network structure, in which the hidden maneuver layer is embedded in-between observable causal and diagnostic evidence layers to enable intercausal reasoning, and
- the additional trash maneuver class, which permits the detection of abnormal events that do not correspond to standard driving maneuvers, and the seamless application from structured to completely unstructured environments.

The Bayesian maneuver detection is combined with novel, probabilistic trajectory prediction models designed for each driving maneuver. These consider uncertainties involved in the execution of the different maneuvers and are adapted to the current driving environment. By combining all predictions of all driving maneuvers, the trajectory prediction step yields

[23] In fact, the TTCCP reasonably disappears earlier (at $t \approx 16\,\mathrm{s}$) as fewer and fewer predicted trajectories collide with the lateral boundary the closer the vehicle gets to the end of the narrowing.

a joint distribution of future vehicle configurations over the prediction horizon for each vehicle, which includes all driving maneuver uncertainties.

The prediction results of all objects in the traffic scene are subsequently used – together with a PFS map-based representation of arbitrary static driving environments – to calculate an introduced criticality time metric, the *Time-To-Critical-Collision-Probability* (TTCCP), as a generalization of TTC for *arbitrary*, *uncertain*, *multi-object* driving environments and *longer* prediction horizons. The TTCCP considers the full probabilistic knowledge of the joint distribution of environment configurations over a future time span and is not based on marginal prediction densities, a specific family of distributions, or point estimates as most often done in nowadays systems. Therefore, multimodal, non-Gaussian configuration distributions are taken into account, which naturally arise if different driving maneuver hypothesis and extended objects are considered. Exemplary driving situations have been simulated in a developed simulation environment, which underline the effectiveness and scope of the developed system. Its main contributions are

- the suppression of many false warnings due to maneuver-based predictions,
- the generation of timelier true warnings in comparison to TTC,
- the possibility to generate warnings in critical almost-collision situations in which a (finite) TTC does not even exist,
- the long-term validity as not only physical quantities are considered but also information about the drivers' intentions and the driving environment,
- the sensitivity to exceptional, rarely happening situations due to the additional trash maneuver class,
- the inclusion of different sources of uncertainty, i.e. uncertainty in states, driver intentions, and driving maneuver executions,
- the consideration of multiple extended dynamic objects,
- the inclusion of arbitrary static environments via PFS maps, and
- the theoretically sound Bayesian combination of symbolic, qualitative maneuver descriptions; spatio-temporal, quantitative trajectory predictions; and probabilistic criticality assessment.

Although the proposed method contains novelties in the areas of maneuver detection, trajectory prediction, and criticality assessment, the main contribution might lie in the throughout design considerations of the individual components with the aim of long-term criticality assessment in mind. In summary, the system satisfies the requirements stated in the motivation and provides the answer to the third main question raised in the thesis introduction, i.e. how a system for *long-term*, *multi-object* trajectory prediction and criticality assessment in *arbitrary*, *uncertain* driving environments can be realized.

In the future, the developed criticality assessment system should be put to the test and evaluated in real driving environments, which was out of scope of this thesis. Open research questions in this regard are how to evaluate such systems as it is nontrivial to come up with objective measures for a quantitative system evaluation. The difference between true TTC and estimated TTC or TTCCP progression is, for example, unsuitable because warnings might be perfectly correct and reasonable although no collision happens in the end. Also the comparison of predicted positions or, more general, predicted distributions between different approaches would always favor systems that model the average driver, which in turn provide unreasonable results in exceptional, rarely happening situations that cannot be learned beforehand. It seems that an evaluation can only reasonably be performed by human probands, who pass their opinion of whether a possible warning or intervention is regarded as helpful or rather disturbing or annoying in a given situation and hardly by machine-calculated evaluation measures.

Summarizing the results described in this thesis so far, a novel environment representation and a novel method for long-term trajectory prediction and criticality assessment have been presented. Both are important modules for driver assistance systems but need to be combined with other subsystems, such as trajectory planning, vehicle control, behavior planning, and human machine interfaces, to realize a complete ADAS. The next chapter therefore focuses on the description of the driver assistance system PRORETA 3, which constitutes a novel, integrated approach to collision avoidance and vehicle automation and which relies on the environment representation proposed in this thesis.

7 PRORETA 3: An Integrated Driver Assistance System

The present chapter presents the integrated driver assistance system PRORETA 3, which employs the methods introduced in this thesis. After a short introduction and motivation, an overview of the system including its software architecture, environment representation, trajectory planning, and *Human Machine Interface* (HMI) is given. Afterwards, a variety of driving scenarios that PRORETA 3 can handle are shown to illustrate the capabilities and scope of the system. Results obtained from test persons who experienced the PRORETA 3 assistance concept in a real prototype vehicle are presented subsequently. The chapter closes with a short conclusion.

7.1 Introduction and Motivation

As already stated in the thesis introduction, there currently exist a large number of different ADAS. These, however, mainly operate as independent, individually designed units for very specific traffic scenarios. The ever increasing number of assistance functions not only entails arbitration difficulties but also makes it harder for the driver to understand and handle all functions properly. PRORETA 3 aims at decreasing the complexity for the driver by just providing two easy to understand modes, namely the

- *Safety Corridor* (SC) as a unified active safety function for a large variety of dangerous traffic situations, and the
- *Cooperative Automation* (CA) as a unified driving comfort function via maneuver-based vehicle guidance, which relieves the driver from vehicle stabilization tasks.

Both modes build upon the same holistic environment representation, the same integrated trajectory planning, and the same HMI elements to ensure a consistent and comprehensible ADAS, which has been fully implemented on a real prototype vehicle provided by the Continental AG, see Fig. 7.1.

Figure 7.1: PRORETA 3 prototype vehicle with core research and development team. From left to right: Matthias Pfromm (HMI), Matthias Schreier (Environment Representation and Prediction), Felix Lotz (System Architecture and Behavior Planning), and Eric Bauer (Trajectory Planning).

PRORETA 3 is the successor of PRORETA 1 and 2 [112]. PRORETA 1 provides safety warnings and emergency interventions by active steering and braking with respect to vehicles moving in the same direction, PRORETA 2 concentrates on dangerous overtaking maneuvers. All three PRORETA projects resulted from a cooperation between the TU Darmstadt and the Continental AG. In the following, an overview of PRORETA 3 is given, which the author co-developed with his colleagues, see Fig. 7.1, in the course of this thesis. For more detailed information, the reader is invited to consult the related articles [25, 56, 274, 275] and dissertations [23, 158, 196].

7.2 System Overview

As already stated, PRORETA 3 is based on two functional modes: SC and CA. Within the SC mode, the driver conducts the vehicle in the traditional way while the system continuously monitors the situation in the background. Only if the situation becomes critical, the system will support the driver to avert the danger. For this purpose, the driver is first informed of the danger, then warned, and – if this does not lead to an appropriate driver reaction – the system takes control to follow a safe, accident-avoiding trajectory to prevent the upcoming accident. This safe trajectory considers available static free space, predicted dynamic objects, and situation-specific information such as road lanes or critical traffic rule violations. The warning and intervention cascade is further adapted to the driver attentiveness to give and apply only necessary information and interventions. In contrast to many current active safety systems, the SC mode is not limited to specific scenarios and adopts a unified functional philosophy within its limits.[1]

The CA mode can be activated on top of the SC mode and offers partially automated driving in form of a maneuver-based vehicle guidance concept. Within the CA mode, the driver just commands discrete maneuvers such as lane changes or turns, and the vehicle executes the chosen maneuvers under consideration of traffic rules via automated longitudinal and lateral vehicle control. The driver is thus relieved from error-prone, tedious vehicle stabilization tasks but still remains responsible for supervising maneuver executions and staying vigilant. The CA mode thus not only increases driving comfort but also contributes to driving safety as it lowers the latent danger, e.g. by keeping to speed limits, while simultaneously keeping the driver in the loop. Since the SC is still active in the background, all active safety-related system functions also apply within the CA mode.

In contrast to other research projects about cooperative driving, for an overview see [83], PRORETA 3 offers a fixed level of vehicle automation and standard HMI input devices to create a near-future automation concept as an intermediate step between state-of-the-art ADAS and fully automated driving. The system's software architecture and its main modules are shortly explained in the following.

7.2.1 Software Architecture

Fig. 7.2 gives an overview of the PRORETA 3 software architecture based on [104, 157]. It comprises two hierarchically arranged horizontal layers: the coordination layer (blue) and the behavior layer (orange).[2] Each of these layers consists of three main components: the world model, the

[1] Of course, not all dangerous situation can be addressed as the system is, for example, limited by the sensor setup with finite sensing range, accuracy, and precision.

[2] The two layers can easily be extended by a high-level mission layer, see [104].

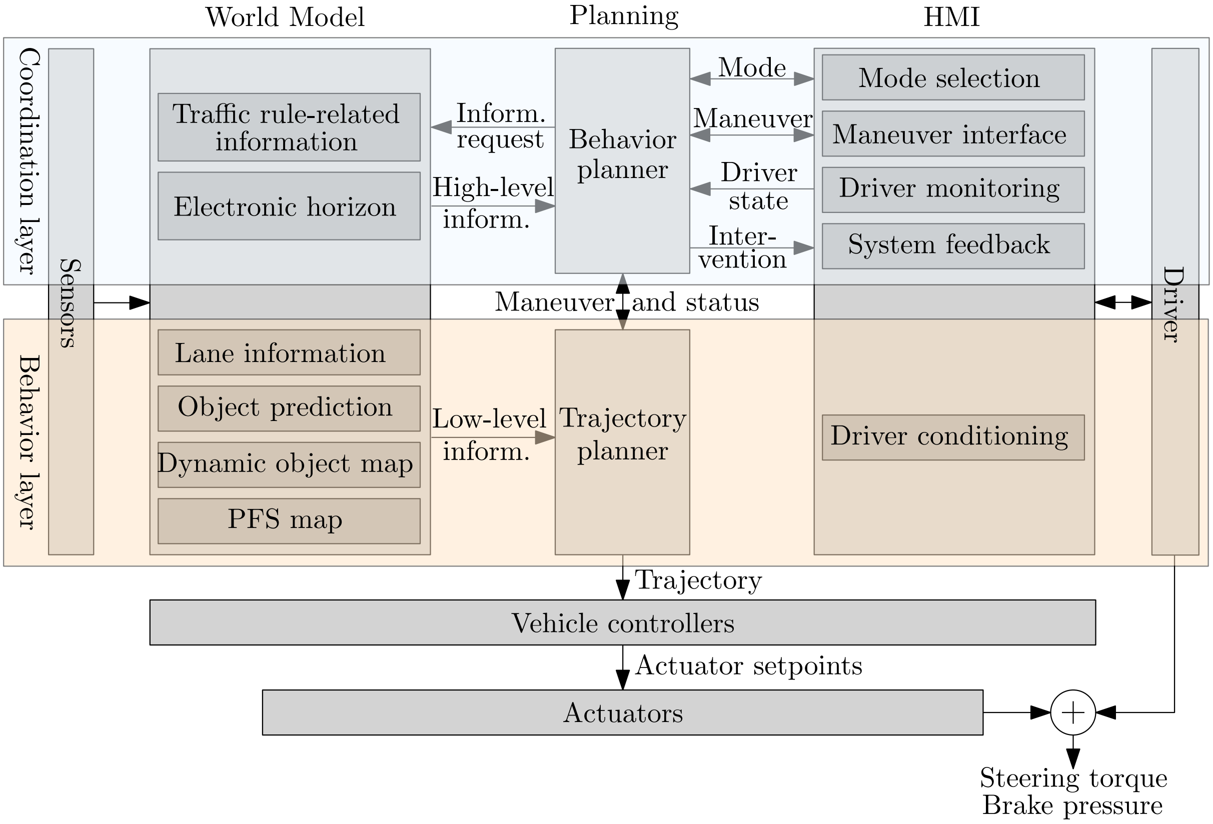


Figure 7.2: Simplified PRORETA 3 software architecture based on [104, 157].

planning modules, and the HMI modules. The *world model* provides the planning modules with information to enable appropriate decisions. The exchanged information grade varies from more abstract in the coordination layer in the form of high-level traffic rule-related information and an electronic horizon, to more detailed low-level metric information in the behavior layer in the form of lane knowledge, dynamic object predictions, dynamic object maps, and PFS maps. The *planning modules* consist of a high-level behavior planner and a low-level trajectory planner. The behavior planner not only receives information from the world model but also from the *HMI modules* in form of the active PRORETA mode, the chosen maneuver, and driver state information.

If the CA mode is enabled by the driver, the behavior planner decides which maneuvers are offered to the driver under consideration of the current static vehicle environment and high-level world model information. The possible maneuvers are forwarded to the HMI, and if a chosen maneuver is safe to be executed, the behavior planner delegates the commanded maneuver to the underlying trajectory planner. The trajectory planner then calculates a maneuver trajectory, which itself is send to underlying longitudinal and lateral vehicle controllers that generate appropriate actuator setpoints. The actuators in turn apply the resulting steering torque and brake pressure values. Since the driver has direct mechanical access to the actuator outputs, the system can always be overruled by the driver in this regard. Additionally, the trajectory planner reports the maneuver status back to the behavior planner, e.g. to which percentage the maneuver is finished.

The architecture remains the same if the SC mode is enabled by the driver. In this case, the behavior planner supervises the driver's focus of attention, which is provided by the HMI via a driver monitoring camera. This driver attention information is compared with the criticality level obtained from the underlying trajectory planner as soon as a potentially critical situation arises. The behavior planner can then adapt the warning and intervention mechanism so that a distracted driver is assisted earlier than an attentive driver to reduce false positive warnings. The concrete warning strategy is determined in the HMI's driver conditioning module, whereas the evasive trajectory is planned by the trajectory planner under consideration of the low-level world model information and subsequently sent to the vehicle controllers.

More detailed information about the software architecture and the behavior planning can be found in [158]. In the following, it is demonstrated how the metric environment representation is coupled with the trajectory planner, and how the HMI components of PRORETA 3 permit the interaction between vehicle and driver.

7.2.2 Environment Representation and Planning

PRORETA 3 is an example for an assistance concept, for which solely sparse object-based environment representations – as used in today's adaptive cruise control or emergency braking systems – are not sufficient as they cannot represent unstructured or only slightly structured urban and rural environments. These include, for example, parked vehicles, trees, houses, or traffic islands as obstacles that have to be taken into account within the SC mode. Consequently, the requirement of a dense representation becomes obvious that is not based on predefined geometric shapes any more. Moreover, free space needs to be represented explicitly for safety-related trajectory planning and transferred efficiently between different computational units. Therefore, the metric environment representation proposed in this thesis is implemented in PRORETA 3 and thus comprises a combination of PFS maps and dynamic object maps.

The sensor setup employed to obtain the representation is illustrated in Fig. 7.3 along with the corresponding sensor fields of view. It consists of a stereo camera, which additionally captures lane markings, traffic signs, and traffic lights, a long-range radar (77 GHz) behind the front bumper as well as two short-range and two blind spot detection radars (24 GHz), which cover the area alongside and behind the PRORETA 3 vehicle. Together with the object predictions and lane information, the introduced environment representation provides the basis for trajectory planning.

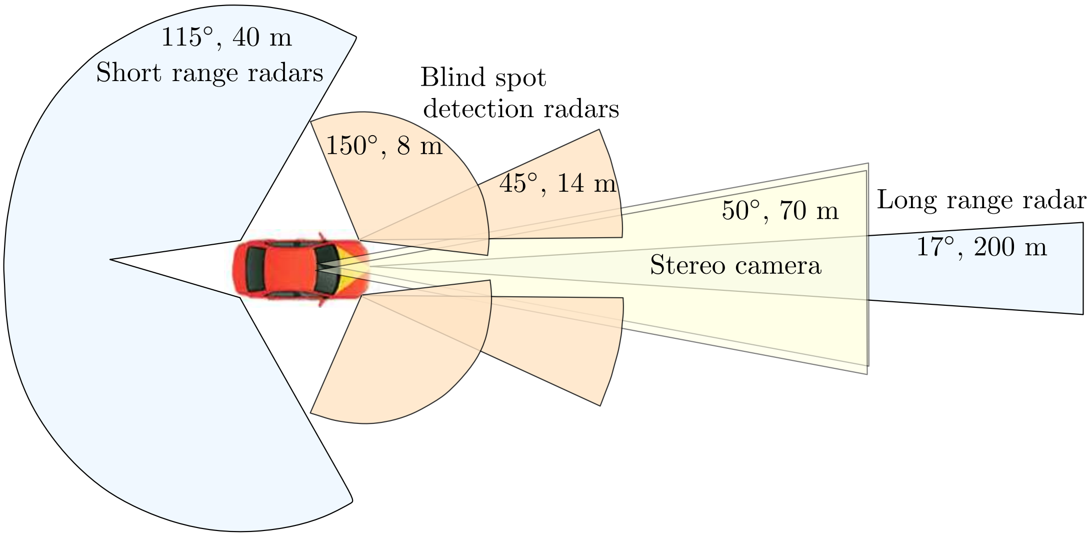


Figure 7.3: PRORETA 3 sensor setup with corresponding fields of view.

For this purpose, the PFS map and the dynamic object map are mapped onto a potential hazard field in a way that high potentials are assigned to areas close to free space boundaries. For the exemplary driving scene, the transition from the environment representation to the potential field is illustrated in Fig. 7.4. It is evident that areas near the free space delimiting B-spline curve and near free space holes receive high potentials, whereas low potentials are assigned to other locations with drivable free space. Dynamic objects are handled by predicting their configuration forward in time, e.g. with the approach introduced in Chapter 6, and setting up a prediction time-dependent potential field. The resulting potential field is part of a cost functional, which is minimized online with respect to a vehicle dynamics model via a model predictive control approach. The resulting trajectory is recalculated in every time step in the background and analyzed with regard to vehicle dynamics-related states, such as the maximum friction coefficient necessary to execute the trajectory. If thresholds are exceeded, an emergency warning or intervention is triggered within

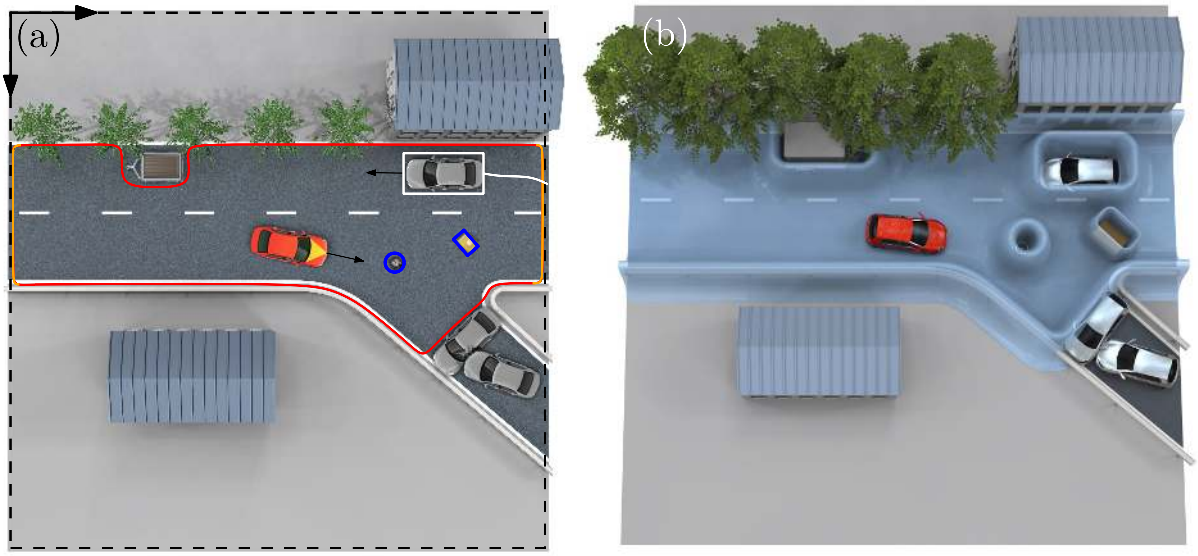


Figure 7.4: Transition from the metric environment representation to trajectory planning. The representation (a) is mapped onto a potential hazard field (b) by assigning high potentials to areas near free space boundaries. The hazard field itself is part of a cost functional from which the safe trajectory is derived via optimization.

the SC mode. Only in the latter case, the calculated trajectory is forwarded to the vehicle controllers to bring the vehicle back to a safe state automatically.

If the trajectory planner receives the request to execute a maneuver from the behavior planner within the mode CA, no decisions are needed, and the calculated trajectory is continuously forwarded to the controllers. To obtain the trajectory within CA, the potential field is varied to account for the desired maneuver, e.g. the minimum of the road potential field is shifted from one lane to the other in case of a desired lane change maneuver. More detailed information about the trajectory planner can be found in [23, 24].

7.2.3 Human Machine Interface

The PRORETA 3 HMI is based on established input devices such as the steering wheel, accelerator pedal, brake pedal, and levers to permit a near-future realization without requiring lengthy driver adaptions. One lever (Fig. 7.5a) is used to change between the PRORETA modes SC and CA. If SC is chosen (CA off), the HMI's task is to provide information, warnings, and action recommendations for the driver if necessary.

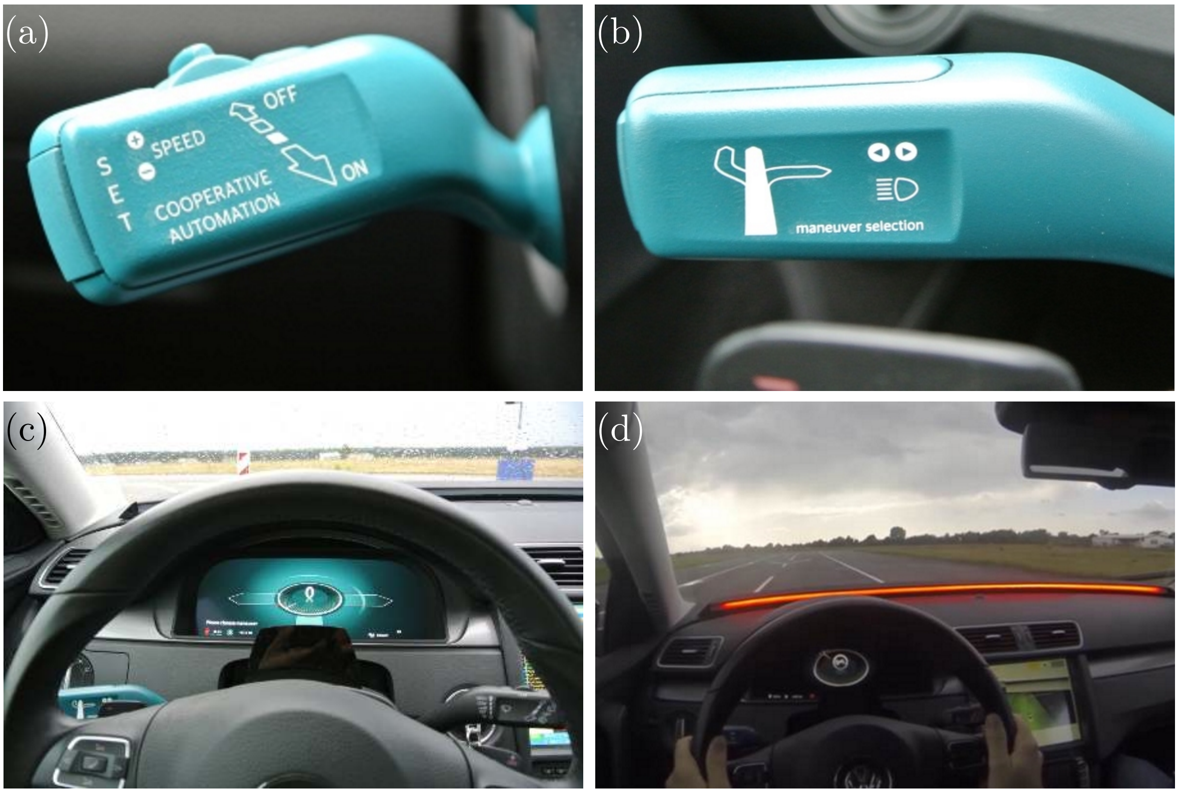


Figure 7.5: PRORETA 3 HMI components. Mode selection lever (a), maneuver selection lever (b), instrument cluster with driver monitoring camera (c), LED stripes (d).

For this purpose, a display in the instrument cluster (7.5c) shows an abstraction of the safety corridor in form of a metaphoric "safety bubble" that dents at the location from where the danger occurs. The stronger the dent, the higher is the criticality of the current driving situation from this direction. Fig. 7.6(a,b,c) exemplarily show the safety bubble evolution during a dangerous approach towards a standing obstacle. The bubble dents only weakly at first as long as the danger is not directly imminent (Fig. 7.6a). The driver is thus just informed and warned, which is additionally amplified by a directional audible warning signal and LED stripes (Fig. 7.5d), which are activated in red at the specific location [197]. An infrared driver monitoring camera mounted behind the steering wheel (Fig. 7.5c) additionally determines the driver's viewing direction to avoid unnecessary and annoying warnings. If the vehicle approaches further without a suitable driver reaction, the bubble dents more strongly (Fig. 7.6b), and an action recommendation is given to the driver. This recommendation is not only transmitted visually in form of a brake symbol in the bubble but also haptically by an *Accelerator Force Feedback Pedal* (AFFP), which

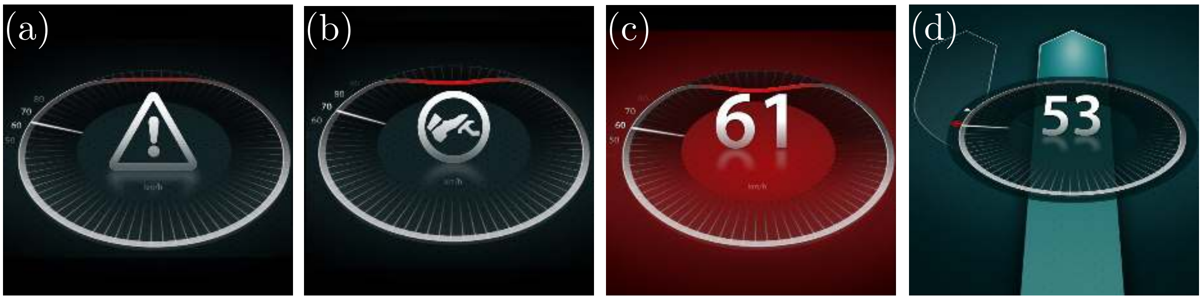


Figure 7.6: PRORETA 3 instrument cluster visualizations. Within the SC mode, the safety bubble dents at the location from where the danger occurs (a,b,c). Within the CA mode, possible maneuver options are offered to the driver and the currently executed maneuver is highlighted in blue (d).

presses against the driver's foot to initiate a correct reaction, i.e. braking in this case. If all warnings and action recommendations go unnoticed, an autonomous emergency maneuver is triggered finally (Fig. 7.6c).

If the driver actives the CA mode, the LED stripes switch their color to blue to ensure mode awareness. Additionally, the instrument cluster changes from the safety bubble to a maneuver representation, see Fig. 7.6d. The display now depicts the different possible driving maneuvers, such as lane changes or turns, along with the currently executed maneuver highlighted in blue. The driver can activate the corresponding maneuver by pushing the maneuver selection lever (Fig. 7.5b) – a modified indicator lever – up or down. The desired speed can be adjusted with the mode selection lever similar to current adaptive cruise control systems. More detailed information about the HMI can be found in [196].

7.3 Driving Scenarios and Results

PRORETA 3 has been presented to the public in September 2014 within a two-day final event, which took place at the August-Euler-Airfield in Griesheim, Germany. The participants experienced PRORETA 3 during test drives, presentations, and demonstrator units, and were encouraged to drive the system for themselves on a test track on the driver's seat. Various test scenarios, see Table 7.1, were set up to permit a comprehensive and authentic system experience. Some impressions of the scenarios are further depicted in Fig. 7.7. Before the results obtained from the test persons are summarized, the scenarios are first explained shortly based on [274] in the following, beginning with the SC mode scenarios.

Table 7.1: PRORETA 3 test scenarios.

Mode	Scenario
SC	Reaction to suddenly appearing dynamic obstacle
SC	Reaction to static obstacle with driver distraction detection
SC	Speed adaption and warning in road bends
SC	Collision avoidance in construction sites
SC	Prevention of unintentional lane departures
SC	Prevention of wrong-way driving
SC	Reaction to red traffic light disregard
CA	Automated lane following with speed adaption
CA	Automated lane change maneuvers
CA	Lane merging scenario
CA	Intersection scenario with traffic signs
CA	Automated driving through construction sites
CA	Intersection scenario with traffic lights

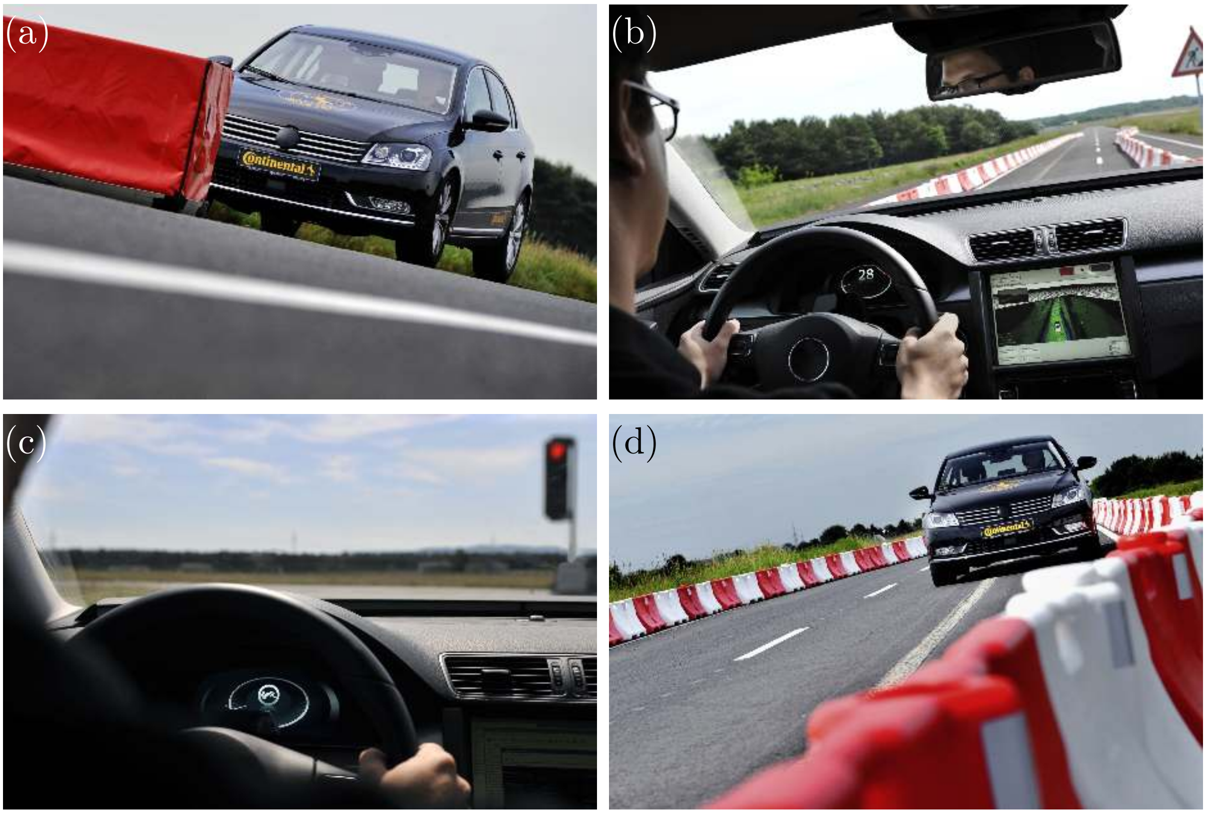


Figure 7.7: Impressions of some PRORETA 3 test scenarios. Emergency intervention with respect to suddenly appearing dynamic obstacle (a), collision avoidance in construction sites (b), reaction to red traffic light disregard (d), automated driving through construction sites (d).

Safety Corridor Scenarios

1. *Reaction to suddenly appearing dynamic obstacle*: The driver is informed to follow the lane without further instructions, only a static obstacle is visible in great distance. Suddenly, a dynamic obstacle is shot into the way from the side, see Fig. 7.7a. If the driver does not brake at all or only too slightly, the PRORETA 3 system performs a full brake execution at the last possible moment in time to prevent the impending collision.

2. *Reaction to static obstacle with driver distraction detection*: The driver is informed to drive towards a static obstacle. Several seconds before the collision, the LED stripes direct the driver's attention to the direction of the hazardous obstacle and additional audible warning signals are triggered. If the driver is distracted and does not look in the direction of the obstacle, an additional "light comet" is activated. If the vehicle approaches further, the safety bubble dents more strongly at the front. If driver reactions are still absent, the AFFP is pressed against the driver's foot, and a strong emergency braking maneuver is finally activated.

3. *Speed adaption and warning in road bends*: In the case that a driver is approaching a road bend with dangerously high speed, red LED stripes are activated as a warning along with an acoustic signal and the AFFP. The vehicle speed is additionally reduced smoothly so that the driver can drive safely through the road bend.

4. *Collision avoidance in construction sites*: The driver is informed to steer against construction walls left and right, see Fig. 7.7b. The safety bubble dents at the corresponding side of the higher criticality of an imminent collision with the static environment structure, determined via the PFS map-based free space detection algorithm. An additional evasive steering torque is applied to prevent the collision.

5. *Prevention of unintentional lane departures*: The driver is informed to leave the lane illegitimately without activating the indicator. The vehicle notices the unintentional lane departure and smoothly directs the vehicle back towards the middle and the orientation of the lane. This leads to more safety compared to a "simple" reactive additional steering torque.

6. *Prevention of wrong-way driving*: If the driver accidentally drives into a one-way street the wrong way, a warning via LED stripes, acoustic signals, and the AFFP is triggered, and an additional visualization of the corresponding wrong-way icon is shown in the instrument cluster. In case of no driver reaction, a smooth braking to a standstill is applied automatically.

7. *Reaction to red traffic light disregard*: A similar warning and intervention strategy is applied if red traffic lights are disregarded or ignored, see Fig. 7.7c. The difference is that the intervention is not as smooth but much more rigorous – similar to an emergency braking with respect to real obstacles.

Besides these 7 SC mode scenarios, the following 6 CA mode scenarios were part of the final event.

Cooperative Automation Scenarios

1. *Automated lane following with speed adaption*: As soon as the driver activates the CA mode via the mode selection lever, the vehicle is permanently centered in the middle of the lane. The controlled speed is the minimum of i) the speed limit indicated by traffic signs, ii) the speed limit determined from the road bend, and iii) the driver's desired speed. The driver can change the latter by pushing the mode selection lever up or down.

2. *Automated lane change maneuvers*: If a lane change maneuver is possible, the driver is informed via the instrument cluster and can select the maneuver by pushing the maneuver selection lever up or down, analogous to the activation of the right or left indicator in a conventional vehicle. The vehicle then performs an automated lane change in the desired direction.

3. *Lane merging scenario*: In this scenario, the left lane is merged into the right lane. If the vehicle is approaching on the left lane and the driver does not select a lane change, the vehicle decelerates smoothly to a standstill. Afterwards, the driver has to choose a lane change so that the vehicle can accelerate and execute the maneuver.

4. *Intersection scenario with traffic signs*: The vehicle receives the information about the intersection approach by a digital map via an

electronic horizon and informs the driver about possible turning directions in the display. The driver can then choose the desired turning maneuver. The *turn right* maneuver is, for example, chosen by pushing the maneuver selection lever up, just like activating the right indicator in a conventional vehicle. The vehicle then performs the turn by itself. If the driver does not choose any maneuver, the behavior planner activates a so-called *default maneuver* such as going straight, which is determined by the specific intersection layout and right-of-way rules.

5. *Automated driving through construction sites*: In analogy to the forth SC scenario, the automated system can handle construction sites and narrow road sections even without lane markings due to its flexible free space detection and trajectory planning approach. This is demonstrated within a 150 m long stretch of road with obstacle boundaries to the left and right, see Fig. 7.7d.

6. *Intersection scenario with traffic lights*: The system is also able to handle simple intersections controlled by traffic lights. During the intersection approach, the camera searches for traffic lights and their state. This information is then processed by the behavior planner, which decides on the upcoming vehicle behavior. Depending on the traffic light state and the desired maneuver, the PRORETA 3 prototype is able to stop at a red traffic light and to continue after the signal switches back to green. Within this process, the driver can change the delegated maneuver as long as the vehicle is at a standstill.

Now that all scenarios have been described, the main results obtained from PRORETA 3 test persons are summarized in the following.

Results

84 participants experienced the PRORETA 3 system in total during the final event. Each of them came in touch with the system for the first time and performed a test drive consisting of two laps. A first lap involved the SC scenarios described previously, a second lap focused on the mentioned CA scenarios. Each test drive took approximately 20 minutes. Afterwards, the test persons answered a questionnaire anonymously. On a 6 point grading scale (from 1: excellent; to 6: fail), the SC mode received 1.92 ± 0.93 points, the CA mode 1.87 ± 0.99 points, and the entire

PRORETA 3 system 1.74 ± 0.86 points [274].[3] Thus, the results of both modes and of the complete system do not differ significantly and underline the throughout positive system experience.

Additionally, the test persons gave feedback on how strongly they agree on several semantic statements that are related to perceived workload, perceived system reliability, perceived safety, driving comfort, and system understandability, see Fig. 7.8. Both modes were found to ease the burden on the driver, work reliably, evoke a feeling of safety, increase driving comfort, and are easy to understand. Especially the CA mode notably increases driving comfort by relieving the driver from tedious vehicle stabilization tasks. More detailed evaluation results of the PRORETA 3 system can be found in [196].

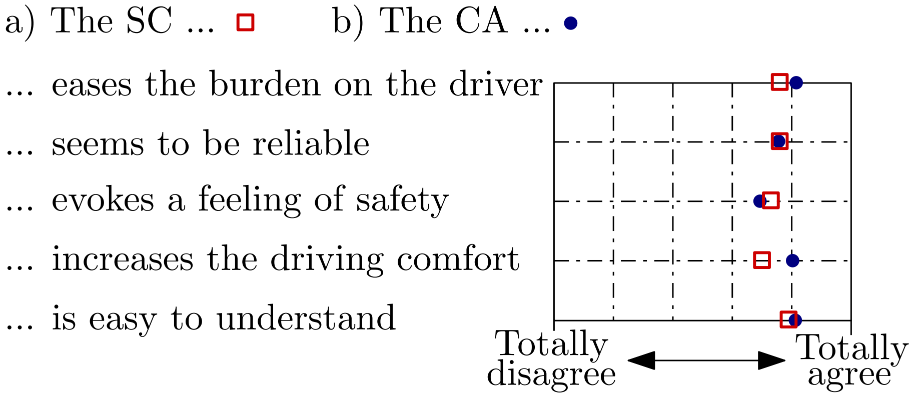


Figure 7.8: Results of the subjective assessment of the PRORETA 3 modes SC and CA based on [274]. Mean values of 84 test persons.

7.4 Conclusion

In conclusion, PRORETA 3 constitutes an integrated ADAS that is not limited to a narrow scope of driving scenarios but permits safety and comfort in a large variety of different traffic situations. The system is based on a hierarchical software architecture, which links the world model, the planning modules, and the HMI modules in a consistent way. To cope with the large amount of possible scenarios, a generic environment representation, which has been introduced in this thesis, serves as the basis for a generic trajectory planning approach. While standard sensor and HMI components facilitate a near-future realization without requiring lengthy vehicle

[3] Thanks to Stefan Cieler for the evaluation.

or driver adaptions, the restriction to just two modes – *Safety Corridor* and *Cooperative Automation* – reduces the perceived system complexity for the driver and makes it easy to understand.

On a test track, the PRORETA 3 vehicle successfully handled numerous hazardous situations within the SC mode including suddenly appearing dynamic obstacles, fast approaches towards strong road bends, near-collisions with construction sites, unintentional lane departures, dangerous wrong-way driving, and disregards of red traffic lights. Within the CA mode, the system relieves the driver from tedious vehicle stabilization tasks and realizes a maneuver-based vehicle guidance concept. Therein, the PRORETA 3 vehicle follows lanes automatically with adequate velocity, and the driver can command lane changes and turns, which are then executed by the vehicle. The test scenarios involved standard lanes, merged lanes as well as different intersections with traffic signs and traffic lights. In addition, the system provides lateral vehicle guidance with respect to elevated objects, which was demonstrated in a construction site scenario.

The final system has been experienced by 84 test persons, who evaluated PRORETA 3 positively with an average grade of 1.74 on a six grade scale. The system including all environment representation, planning, and HMI components ran stable and robust throughout the complete two-day final event with non-stop test runs.

In summary, the maneuver-based driving option within PRORETA 3 in form of the CA mode provides an intermediate step between current ADAS and fully autonomous driving, whereas the additional safety obtained by the SC mode makes a valuable contribution to realize the *Vision Zero*. The following, final chapter summarizes the main results of this thesis and highlights possible future research directions.

8 Summary and Outlook

The key motivation of this thesis was to make a contribution to safe and accident-free driving. Since most accidents are caused by human error, adequate assistance of the driver is an obvious step to advance towards this goal before fully autonomous driving finally becomes a reality. For proper assistance, it is necessary that the vehicle perceives the driving environment, builds an internal model of the world, and extrapolates the current traffic situation into the near-future to provide adequate warnings, action recommendations, and safety interventions. In this context, the three key questions raised in this thesis were i) how to represent the driving environment appropriately, ii) how to obtain such a representation, and iii) how to predict the future traffic scene evolution for proper criticality assessment.

At the beginning of this thesis, different metric environment representations known from the mobile robotics domain have been compared and discussed with respect to their suitability for future ADAS in Chapter 2. Based on the found shortcomings, a novel representation has been derived, which consists of *dynamic object maps* for the representation of dynamic entities in combination with *Parametric Free Space* (PFS) *maps* for encoding relevant static parts of the environment. The newly introduced PFS maps are a continuous bird's-eye view of the local static environment around the ego vehicle, which model the world by a combination of a closed curve and geometric primitives. An important difference to other parametric maps is that not objects are described explicitly but the absence of objects in form of relevant *free space*. The main advantages of the proposed representation are its compactness that allows the transmission within very limited bandwidth; its generality to represent a large variety of driving environments; the suppression of irrelevant free space information for simplified trajectory planning and interpretation; the explicit free space representation vital for safety-related evasive trajectory planning; and the consistent separation between static and dynamic entities for simplified situation interpretation.

Since Bayesian inference plays a major role to realize the proposed representation, all relevant theoretical foundations have subsequently been presented in Chapter 3. These include the description of the employed filter algorithms as special cases of the Bayesian optimal filter as well as the two related filter applications *occupancy grid mapping* and *target tracking*. In this context, an extension of the IMM-PDAF in form of the so-called IMM-UK-PDAF has been presented, which combines Interacting Multiple Model Filters with Unscented Kalman Filters and Probabilistic Data Association Filters. The IMM-UK-PDAF provides an approximate solution for recursively estimating states and mode probabilities of maneuvering targets in clutter, each of which is described by a jump Markov *nonlinear* system, and is used within the generation process of the environment representation.

The following two chapters have been devoted to this generation process. Due to the widespread usage of grid map-based representations, a method has been presented therein that obtains both the *dynamic object map* and the *PFS map* in a consistent way by first building a grid map in dynamic driving environments as an intermediate step. Traditional mapping algorithms are, however, not suitable for this purpose as they implicitly assume a static world. Therefore, a novel method for grid mapping in *dynamic* road environments has been introduced together with a discussion of related work on this topic in Chapter 4. Dynamic cell hypothesis are first generated based on consecutive conventional grid maps, density-based clustering, and appropriate merging of newly free and newly occupied areas. The resulting dynamic object reconstructions are subsequently tracked by IMM-UK-PDAFs with different motion models not merely to recursively estimate their states in clutter but rather to classify between real dynamic objects and false reconstructions via an introduced random motion model. Tracks of real dynamic objects are cleared from the grid and integrated into the dynamic object map. The result is a dynamic object map with estimated state vectors of extended objects along with an optimized occupancy grid that contains only the static driving environment.

The proposed dynamic mapping approach has been implemented on an experimental vehicle and evaluated in real traffic environments. On track level, the system's motion classification capabilities showed persuasive precision, recall, and accuracy values as well as reliable decisions within short observation times. The cell level evaluation revealed superior performance compared to traditional occupancy grid mapping filters that are used in most state-of-the-art ADAS and intelligent vehicles research projects. The key aspects of the approach are the use of newly available free space to detect and reconstruct dynamic object hypothesis even in close vicinity to static environment structures; the robust IMM-UK-PDAF-based motion consistency check, which additionally allows to filter out dynamic mapping errors even if the motion detector fails due to the prediction models; and the straightforward combination with existing mapping algorithms and different environment sensors. The resulting optimized, static grid map provides the basis for the generation of compact PFS maps.

This PFS map generation process has been described in detail in the following Chapter 5 together with a discussion of related work on free space detection and estimation. Methodically, the optimized grid is treated as an image, and methods well-known from the image analysis domain are employed to extract relevant free space boundaries. For this purpose, the grid is filtered, segmented, and different morphological operations are performed. Extracted boundary cells of the so-found, arbitrarily shaped free space segment are treated as measurements for a continuous, dynamic B-Spline free space contour tracking. This tracking step provides additional robustness and offers a straightforward way to incorporate spatio-temporal smoothness constraints. Experimental results obtained from real-world driving scenarios highlighted the generality of the compact PFS map representation and the effective suppression of irrelevant information contained in common occupancy grids. By parallel programming techniques and various algorithmic optimizations, the complete metric environment representation including both maps is extracted in less than 25 ms online on an experimental vehicle and is thus directly applicable in practice.

Up to this point, the thesis was centered on the first two questions, i.e. how to represent the environment adequately and how to generate this representation. The following Chapter 6 transitioned to the question of how to predict and assess the future traffic scene evolution. In this regard, an in-depth review of related work on short-term and long-term trajectory prediction, situation prediction, and criticality assessment has been carried out first. Based on the limitations of previous works, a novel method for probabilistic, long-term *trajectory prediction* and *criticality assessment* in arbitrary driving environments has then been introduced, which uses the proposed environment representation as a foundation. Therein, a distribution of high-level driving maneuvers is inferred for each vehicle in the traffic scene via Bayesian inference initially. For this purpose, the domain has been modeled in a novel Bayesian network with both causal and diagnostic evidences as well as an additional trash maneuver class, which allows the detection of irrational driving behavior and the seamless application from highly-structured to non-structured environments. Subsequently, maneuver-based, probabilistic trajectory prediction models have been designed and employed to predict each vehicle's configuration forward in time. Random elements in these models consider the uncertainty within the future driving maneuver execution of human drivers. Finally, the criticality time metric *Time-To-Critical-Collision-Probability* (TTCCP) has been introduced as a generalization of TTC for *arbitrary*, *uncertain*, *multi-object* driving environments and *longer* prediction horizons. The TTCCP considers the full probabilistic knowledge of the joint distribution of environment configurations over a future time span and is not based on marginal prediction densities, a specific family of distributions, or point estimates as most often done in nowadays systems. All uncertain maneuver-based predictions are taken into account. Additionally, arbitrarily shaped static environments described by the introduced PFS maps are considered in this process.

A simulation environment has been developed to test the approach. It includes a full replica of the environment representation including occupancy grid mapping and PFS mapping as a foundation, the Bayesian network-based maneuver estimation, the probabilistic predictions as realizations of stochastic processes, and all TTCCP-related Monte Carlo estimations. Exemplary driving situations underlined the system's capability to suppress false warnings, to generate timelier true warnings in comparison to TTC, and to generate warnings in critical almost-collision situations in which a (finite) TTC does not even exist. Although the approach contains novelties in the specific areas *maneuver detection*, *trajectory prediction*, and *criticality assessment*, the main contribution might lie in the throughout design considerations of the individual components with the aim of long-term criticality assessment in mind along with a theoretically sound, Bayesian combination of symbolic, qualitative maneuver descriptions; spatio-temporal, quantitative trajectory predictions; and probabilistic criticality assessment. Due to the same underlying Bayesian theory, the contributions obtained in this thesis harmonize naturally with each other and can easily be combined.

All contributions originated in the context of developing the holistic driver assistance system PRORETA 3, which has been presented in the subsequent Chapter 7. PRORETA 3 constitutes a novel, integrated approach to collision avoidance and vehicle automation, which aims at decreasing the system complexity for the driver. This is achieved by providing just two easy to understand modes, namely the i) *Safety Corridor* as a unified active safety function for a large variety of dangerous traffic situations, and the ii) *Cooperative Automation* as a unified driving comfort function via maneuver-based vehicle guidance, which relieves the driver from vehicle stabilization tasks. The system's main components have been explained, and different driving scenarios were presented along with evaluations results obtained from test persons. In summary, the maneuver-based driving option within PRORETA 3 provides an intermediate step between current ADAS and fully autonomous driving, whereas the integral safety aspect makes a valuable contribution to realize the *Vision Zero*.

However, many challenging problems still remain to be solved. These include not just legal or testing aspects as often highlighted but also substantial robotics-, image analysis-, and artificial intelligence-related issues. As far as the low-level environment representation is concerned, accurate state estimates and correct lane assignments even for maneuvering objects in great distances and strong road bends, in combination with precise ego vehicle localization without prerecorded maps, are vital and pose a great challenge for today's sensor hardware and perception algorithms. On a higher-level, dense semantic mapping in form of automatic online labeling of manifold scene classes such as main roads, dirt tracks, sidewalks, traffic isles, curb stones, or guardrails is required to provide a broader description of the traffic scene, and is far from being solved currently. Related to this, the online generation not only of metric but also of topological maps is an important aspect for future work. Especially in complex intersections, this requires the challenging online detection and classification of all possible driving paths. On top of this, automated scene understanding systems in complex environments that take metric and topological maps, traffic rules, vehicle interactions, pedestrian interactions, occlusions, perception uncertainty, weather conditions, etc. into account, need to be developed further. In the context of traffic situation prediction, the joint inference of the real driving situation and the drivers' situation awareness could pave the way for better predictions. If it is possible to infer which elements of the driving situation a driver failed to notice, e.g. via a driver monitoring camera or indirectly via detecting the absence of adequate behavior, then predictions could be based on the drivers' possibly perceived subsets of situation elements. Especially vehicle-to-vehicle communications accompanied with the availability of all vehicles' internal states will offer many possibilities in this regard. Last but not least, the question of how to evaluate prediction and criticality assessment systems properly should be treated in detail in the future. These challenges, along with many more, ensure that research and development in the area of intelligent vehicles and ADAS will still have their raison d'être for a very long time.

A Bayesian Network Parameters

Bayesian network parameters for the network illustrated in Fig. A.1 are given in form of i) the range of values of all nodes in Table A.1, ii) the prior distributions of all root nodes in Table A.2, and iii) the CPTs of all conditional distributions in Table A.3–A.14. Negative time values, e.g. for TLC or $\mathrm{TTO}_{\mathrm{fro}}$, correspond to a motion away from the line or object under consideration. As the network is parametrized symmetrical with respect to left and right motion, the corresponding indices within the CPTs are neglected for better readability. *Don't care* (d.c.) entries are employed if the state value does not affect the corresponding entry in the CPT. In normal operation, all root nodes except $\mathrm{TR}_{\mathrm{l,lon,r}}$ are evidence nodes and therefore measurable. Thus, the prior distributions of these root nodes do normally not affect the maneuver detection and become relevant only if evidences cannot be extracted or measured, e.g. in the case of a sensor defect.

Table A.1: States of evidence variables with corresponding range of values.

Variable	State 0	State 1	State 2	State 3
$\mathrm{LE}_{\mathrm{l/r/c}}$	false	true	–	–
$\mathrm{TLC}_{\mathrm{l/r}}$	$< 0\,\mathrm{s}$	$[0\,\mathrm{s}, 2\,\mathrm{s}[$	$[2\,\mathrm{s}, 4\,\mathrm{s}]$	$> 4\,\mathrm{s}$
$\mathrm{TTU}_{\mathrm{l/r}}$	$< 2\,\mathrm{s}$	$[2\,\mathrm{s}, 5\,\mathrm{s}]$	$> 5\,\mathrm{s}$	–
$\mathrm{TE}_{\mathrm{l/r}}$	false	true	–	–
v_{rel}	$< -6\,\mathrm{m/s}$	$[-6\,\mathrm{m/s}, 6\,\mathrm{m/s}]$	$> 6\,\mathrm{m/s}$	–
$\mathrm{OE}_{\mathrm{fro}}$	false	true	–	–
$\mathrm{TTO}_{\mathrm{fro}}$	$< 0\,\mathrm{s}$	$[0\,\mathrm{s}, 5\,\mathrm{s}]$	$> 5\,\mathrm{s}$	–
ψ_{R}	$< -0.04\,\mathrm{rad}$	$[-0.04\,\mathrm{rad}, 0.04\,\mathrm{rad}]$	$> 0.04\,\mathrm{rad}$	–
$a_{\mathrm{R,lat}}$	$< -0.2\,\mathrm{m/s^2}$	$[-0.2\,\mathrm{m/s^2}, 0.2\,\mathrm{m/s^2}]$	$> 0.2\,\mathrm{m/s^2}$	–
$v_{\mathrm{R,lat}}$	$< -0.2\,\mathrm{m/s}$	$[-0.2\,\mathrm{m/s}, 0.2\,\mathrm{m/s}]$	$> 0.2\,\mathrm{m/s}$	–
$a_{\mathrm{R,lon}}$	$< -1\,\mathrm{m/s^2}$	$[-1\,\mathrm{m/s^2}, 1\,\mathrm{m/s^2}]$	$> 1\,\mathrm{m/s^2}$	–

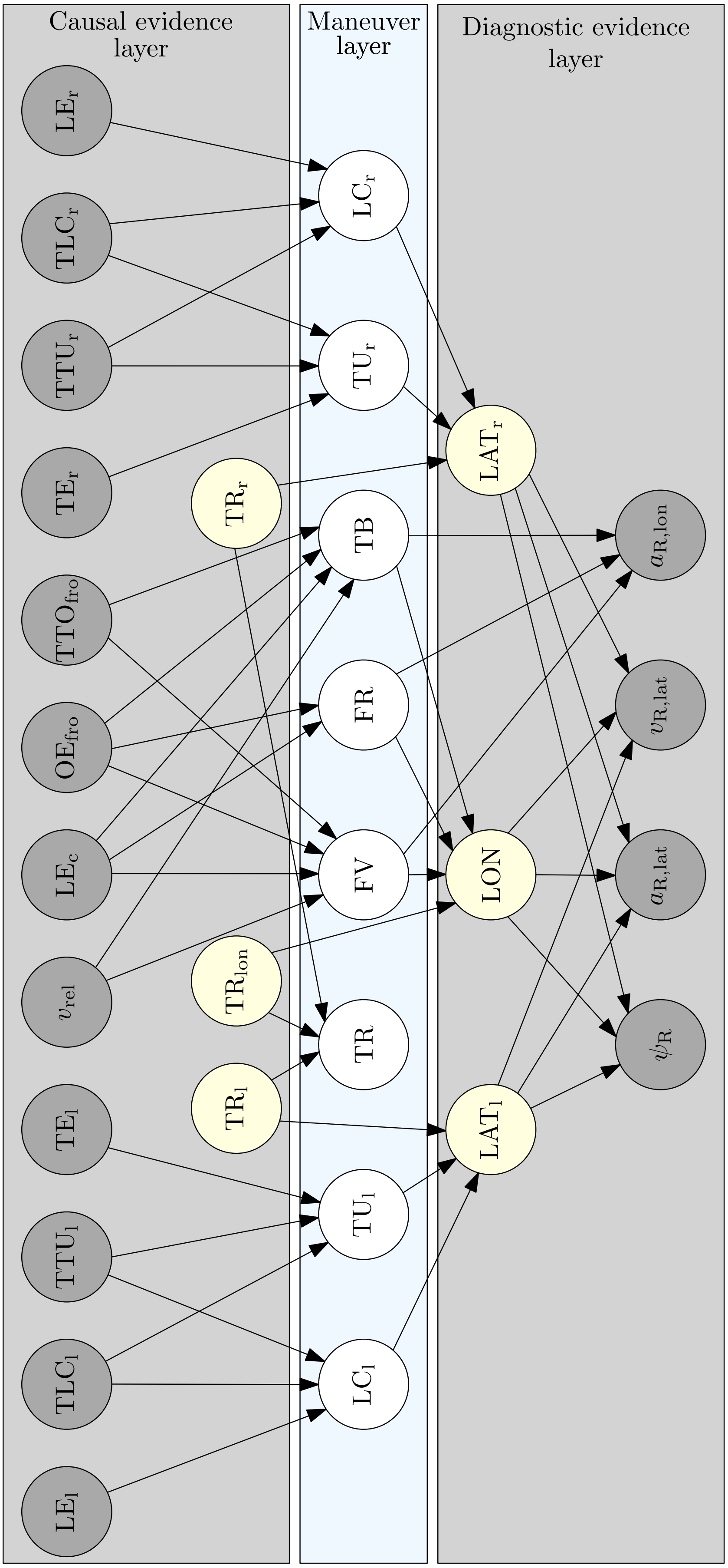


Figure A.1: Bayesian network for maneuver detection. Abbreviations are explained in Table 6.2, p.153.

Table A.2: Prior distributions of all root nodes.

Variable	State 0	State 1	State 2	State 3
$\mathrm{LE}_{\mathrm{l/r/c}}$	0.99	0.01	–	–
$\mathrm{TLC}_{\mathrm{l/r}}$	0.97	0.01	0.01	0.01
$\mathrm{TTU}_{\mathrm{l/r}}$	0.01	0.01	0.98	–
$\mathrm{TE}_{\mathrm{l/r}}$	0.99	0.01	–	–
v_{rel}	0.49	0.02	0.49	–
$\mathrm{OE}_{\mathrm{fro}}$	0.99	0.01	–	–
$\mathrm{TTO}_{\mathrm{fro}}$	0.98	0.01	0.01	–
$\mathrm{TR}_{\mathrm{l/lon/r}}$	0.9	0.1	–	–

Table A.3: CPT of $p(\mathrm{LC}|\mathrm{LE},\mathrm{TTU},\mathrm{TLC})$.

LE		1												0
TTU		0				1				2				d.c.
TLC		0	1	2	3	0	1	2	3	0	1	2	3	d.c.
LC	0	1	0.99	0.99	1	1	0.7	0.9	0.99	1	0.5	0.9	0.99	1
	1	0	0.01	0.01	0	0	0.3	0.1	0.01	0	0.5	0.1	0.01	0

Table A.4: CPT of $p(\mathrm{TU}|\mathrm{TE},\mathrm{TTU},\mathrm{TLC})$.

TE		1												0
TTU		0				1				2				d.c.
TLC		0	1	2	3	0	1	2	3	0	1	2	3	d.c.
TU	0	1	0.4	0.9	0.95	1	0.7	0.9	0.99	1	0.99	0.99	1	1
	1	0	0.6	0.1	0.05	0	0.3	0.1	0.01	0	0.01	0.01	0	0

Table A.5: CPT of $p(\mathrm{TR}|\mathrm{TR}_{\mathrm{l}}, \mathrm{TR}_{\mathrm{r}}, \mathrm{TR}_{\mathrm{lon}})$.

TR_{l}		0				1			
TR_{r}		0		1		0		1	
$\mathrm{TR}_{\mathrm{lon}}$		0	1	0	1	0	1	0	1
TR	0	1	0.7	0.7	0.49	0.7	0.49	0.49	0.34
	1	0	0.3	0.3	0.51	0.3	0.51	0.51	0.66

Table A.6: CPT of $p(\mathrm{FV}|\mathrm{LE}_{\mathrm{c}}, \mathrm{OE}_{\mathrm{fro}}, v_{\mathrm{rel}}, \mathrm{TTO}_{\mathrm{fro}})$.

LE_{c}		1									0	d.c.
$\mathrm{OE}_{\mathrm{fro}}$		1									d.c.	0
v_{rel}		0			1			2			d.c.	d.c.
$\mathrm{TTO}_{\mathrm{fro}}$		0	1	2	0	1	2	0	1	2	d.c.	d.c.
FV	0	0.98	0.98	0.98	0.5	0.2	0.4	0.98	0.98	0.98	1	1
	1	0.02	0.02	0.02	0.5	0.8	0.6	0.02	0.02	0.02	0	0

Table A.7: CPT of $p(\mathrm{FR}|\mathrm{LE}_\mathrm{c}, \mathrm{OE}_\mathrm{fro})$.

LE_c		0		1	
OE_fro		0	1	0	1
FR	0	1	1	0.2	0.8
	1	0	0	0.8	0.2

Table A.8: CPT of $p(\mathrm{TB}|\mathrm{LE}_\mathrm{c}, \mathrm{OE}_\mathrm{fro}, v_\mathrm{rel}, \mathrm{TTO}_\mathrm{fro})$.

LE_c		1									0	d.c.
OE_fro		1									d.c.	0
v_rel		0			1			2			d.c.	d.c.
$\mathrm{TTO}_\mathrm{fro}$		0	1	2	0	1	2	0	1	2	d.c.	d.c.
TB	0	0.99	0.99	0.99	0.99	0.5	0.9	0.99	0.5	0.9	1	1
	1	0.01	0.01	0.01	0.01	0.5	0.1	0.01	0.5	0.1	0	0

Table A.9: CPT of $p(\mathrm{LAT}|\mathrm{LC}, \mathrm{TU}, \mathrm{TR})$.

LC		0				1			
TU		0		1		0		1	
TR		0	1	0	1	0	1	0	1
LAT	0	1	0	0	0	0	0	0	0
	1	0	1	1	1	1	1	1	1

Table A.10: CPT of $p(\mathrm{LON}|\mathrm{TR}, \mathrm{FV}, \mathrm{FR}, \mathrm{TB})$.

TR		0	1	d.c.	d.c.	d.c.
FV		0	d.c.	1	d.c.	d.c.
FR		0	d.c.	d.c.	1	d.c.
TB		0	d.c.	d.c.	d.c.	1
LON	0	1	0	0	0	0
	1	0	1	1	1	1

Table A.11: CPT of $p(\psi_\mathrm{R}|\mathrm{LAT}_\mathrm{l}, \mathrm{LON}, \mathrm{LAT}_\mathrm{r})$.

LAT_l		0				1			
LON		0		1		0		1	
LAT_r		0	1	0	1	0	1	0	1
ψ_R	0	0.41	0.6	0.31	0.48	0.01	0.4	0.01	0.33
	1	0.18	0.39	0.38	0.51	0.39	0.2	0.51	0.34
	2	0.41	0.01	0.31	0.01	0.6	0.4	0.48	0.33

Table A.12: CPT of $p(a_\mathrm{R,lat}|\mathrm{LAT}_\mathrm{l}, \mathrm{LON}, \mathrm{LAT}_\mathrm{r})$.

LAT_l		0				1			
LON		0		1		0		1	
LAT_r		0	1	0	1	0	1	0	1
$a_\mathrm{R,lat}$	0	0.33	0.61	0.32	0.64	0.01	0.4	0.14	0.5
	1	0.34	0.38	0.36	0.22	0.38	0.2	0.22	0
	2	0.33	0.01	0.32	0.14	0.61	0.4	0.64	0.5

Table A.13: CPT of $p(v_{\mathrm{R,lat}}|\mathrm{LAT_l}, \mathrm{LON}, \mathrm{LAT_r})$.

$\mathrm{LAT_l}$		0				1			
LON		0		1		0		1	
$\mathrm{LAT_r}$		0	1	0	1	0	1	0	1
$v_{\mathrm{R,lat}}$	0	0.33	1	0	0.7	0	0.01	0	0.1
	1	0.34	0	1	0.3	0	0.98	0.3	0.8
	2	0.33	0	0	0	1	0.01	0.7	0.1

Table A.14: CPT of $p(a_{\mathrm{R,lon}}|\mathrm{FV}, \mathrm{FR}, \mathrm{TB})$.

FV		0				1			
FR		0		1		0		1	
TB		0	1	0	1	0	1	0	1
$a_{\mathrm{R,lon}}$	0	0.33	1	0.33	1	0.1	1	0.2	1
	1	0.34	0	0.34	0	0.8	0	0.6	0
	2	0.33	0	0.33	0	0.1	0	0.2	0

B Publications and Supervisions

B.1 List of Publications by the Author

B.1.1 Journal and Book Chapter Publications

- E. Bauer, F. Lotz, M. Pfromm, M. Schreier, S. Cieler, A. Eckert, A. Hohm, S. Lüke, P. Rieth, B. Abendroth, V. Willert, J. Adamy, R. Bruder, U. Konigorski, and H. Winner. PRORETA 3: An Integrated Approach to Collision Avoidance and Vehicle Automation. *at – Automatisierungstechnik*, 60(12):755–765, Dec. 2012.

- M. Schreier. Quaternion-Based Adaptive Attitude Control Schemes for Quadrotor Systems. *International Journal of Mechatronics and Automation*, 3(4):217–229, Dec. 2013.

- M. Schreier, V. Willert, and J. Adamy. Compact Representation of Dynamic Driving Environments for ADAS by Parametric Free Space and Dynamic Object Maps. *IEEE Transactions on Intelligent Transportation Systems*, PP(99):1–18, Sept. 2015.

- M. Schreier, V. Willert, and J. Adamy. An Integrated Approach to Maneuver-Based Trajectory Prediction and Criticality Assessment in Arbitrary Road Environments. *IEEE Transactions on Intelligent Transportation Systems*, accepted.

- H. Winner, F. Lotz, E. Bauer, U. Konigorski, M. Schreier, J. Adamy, M. Pfromm, R. Bruder, S. Lüke, and S. Cieler. PRORETA 3: Comprehensive Driver Assistance by Safety Corridor and Cooperative Automation. In *Handbook of Driver Assistance Systems. Basic Information, Components and Systems for Active Safety and Comfort*, chapter 58, pages 1449–1469. Springer, Switzerland, 2016.

B.1.2 Conference Publications

- M. Schreier. Modeling and Adaptive Control of a Quadrotor. In *Proc. of the IEEE International Conference on Mechatronics and Automation*, pages 383–390, Chengdu, China, Aug. 2012.
- M. Schreier and E. Bauer. Projekt PRORETA 3 – Ein integrales Fahrerassistenzsystem zur Kollisionsvermeidung und Fahrzeugautomatisierung. In *12. Kooperationsforum Fahrerassistenzsysteme*, Aschaffenburg, Germany, May 2015.
- M. Schreier, E. Bauer, J. Adamy, and U. Konigorski. Das Fahrerassistenzsystem PRORETA 3: Umgebungsrepräsentation und Trajektorienplanung. In *48. Regelungstechnisches Kolloquium*, Boppard, Germany, Feb. 2014.
- M. Schreier and V. Willert. Robust Free Space Detection in Occupancy Grid Maps by Methods of Image Analysis and Dynamic B-Spline Contour Tracking. In *Proc. of the IEEE International Conference on Intelligent Transportation Systems*, pages 514–521, Anchorage, AK, USA, Sept. 2012.
- M. Schreier, V. Willert, and J. Adamy. From Grid Maps to Parametric Free Space Maps – A Highly Compact, Generic Environment Representation for ADAS. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 938–944, Gold Coast, Australia, June 2013.
- M. Schreier, V. Willert, and J. Adamy. Bayesian, Maneuver-Based, Long-Term Trajectory Prediction and Criticality Assessment for Driver Assistance Systems. In *Proc. of the IEEE International Conference on Intelligent Transportation Systems*, pages 334–341, Qingdao, China, Oct. 2014.
- M. Schreier, V. Willert, and J. Adamy. Grid Mapping in Dynamic Road Environments: Classification of Dynamic Cell Hypothesis via Tracking. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 3995–4002, Hong Kong, China, June 2014.
- H. Winner, F. Lotz, E. Bauer, U. Konigorski, M. Schreier, J. Adamy, M. Pfromm, R. Bruder, S. Lüke, and S. Cieler. PRORETA 3 – An Integrated ADAS Concept – Comprehensive Driver Assistance by Safety Corridor and Cooperative Automation. In *1. Internationale ATZ-Fachtagung Fahrerassistenzsysteme*, Frankfurt a. M., Germany, Apr. 2015.

B.2 List of Supervisions by the Author

- I. Achieser. Trajektorienprädiktion zur Kritikalitätsbewertung von Fahrsituationen. Bachelor's Thesis, Oct. 2012.
- S. Bernhard. Modellbasierte Schätzung der Schwerpunktlage eines Fahrzeugs. Master's thesis, Jan. 2014.
- K. S. Buhle. Anpassung eines Fahrsimulatorszenarios für die Evaluation von Fahrerassistenzsystemen am Beispiel Conduct-by-Wire. Bachelor's Thesis, Nov. 2012.
- A. El Monjid. Erkennung von Fahrmanövern für Fahrerassistenzsysteme. Diploma Thesis, Aug. 2012.
- P. Keßler. Modellbasierte Schätzung der Referenzgeschwindigkeit eines Fahrzeugs. Master's thesis, Mar. 2013.
- S. Licitar. Methoden der Situationsanalyse im Automobilumfeld. Studienarbeit, Sept. 2011.
- T. Lingenau. Entwicklung eines Querreglers auf Basis des Einspurmodells mittels Zustandsregelung. Diploma Thesis, Nov. 2011.
- M. Ma. Implementierung und Evaluierung eines Bewertungsalgorithmus für Kompressionsverfahren von Occupancy Grids auf hochparallelen Recheneinheiten (GPUs) unter Verwendung aktueller Programmierschnittstellen. Master's thesis, Feb. 2014.
- W. Mahmud. Kategorisierung und Bewertung von Transitionen zwischen Mensch und Automation. Studienarbeit, Feb. 2014.
- S. Merten. Modellierung einer Gridmap für eine Simulationsumgebung für Fahrerassistenzsysteme. Bachelor's Thesis, Mar. 2012.
- F. C. Papst. Verkehrsknotendetektion aus parametrischen Freiraum- und Gridkarten. Diploma Thesis, Nov. 2013.
- Y. Qin. Optimierung eines Systems zur Kritikalitätsbewertung von Fahrsituationen. Master's thesis, Jan. 2014.
- J. Schönichen. Integration von Informationen digitaler Karten in ein Fusions-Framework zur Verbesserung der Umfeldwahrnehmung für Fahrerassistenz-Konzepte. Master's thesis, Nov. 2012.

- J. Seitz. Parametrische Freiraumkarten in der Kritikalitätsbewertung von Fahrsituationen. Bachelor's Thesis, Sept. 2013.
- S. Waz. Haptikprüfung eines kapazitiven Touchpads. Diploma Thesis, Nov. 2012.
- G. Wege. Probabilistische Langzeit-Trajektorienprädiktion für Fahrerassistenzsysteme. Diploma Thesis, May 2013.
- R. Zou. Freiraumbestimmung auf Basis von Occupancy-Gridmaps. Master's thesis, Apr. 2012.

Bibliography

[1] I. Achieser. Trajektorienprädiktion zur Kritikalitätsbewertung von Fahrsituationen. Bachelor's Thesis, Technische Universität Darmstadt, Darmstadt, Germany, Oct. 2012.

[2] J. Adamy. *Nichtlineare Systeme und Regelungen*. Springer, Berlin, Germany, second, revised and extended edition, 2014.

[3] G. Agamennoni, J. I. Nieto, and E. M. Nebot. A Bayesian Approach for Driving Behavior Inference. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 595–600, Baden-Baden, Germany, June 2011.

[4] M. Althoff, D. Althoff, D. Wollherr, and M. Buss. Safety Verification of Autonomous Vehicles for Coordinated Evasive Maneuvers. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 1078–1083, San Diego, CA, USA, June 2010.

[5] M. Althoff, O. Stursberg, and M. Buss. Model-Based Probabilistic Collision Detection in Autonomous Driving. *IEEE Transactions on Intelligent Transportation Systems*, 10(2):299–310, June 2009.

[6] B. D. O. Anderson and J. B. Moore. *Optimal Filtering*. Information and System Sciences. Prentice-Hall, Englewood Cliffs, NJ, USA, 1979.

[7] G. S. Aoude, J. Joseph, N. Roy, and J. P. How. Mobile Agent Trajectory Prediction using Bayesian Nonparametric Reachability Trees. In *Proc. of the AIAA Infotech@Aerospace*, pages 1–17, St. Louis, MO, USA, Mar. 2011.

[8] N. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, Feb. 2002.

[9] M. Asif, M. R. Arshad, M. Y. I. Zia, and A. Yahya. An Implementation of Active Contour and Kalman Filter for Road Tracking. *IAENG International Journal of Applied Mathematics*, 37(2):71–77, Nov. 2007.

[10] J. Aue, M. R. Schmid, T. Graf, and J. Effertz. Improved Object Tracking from Detailed Shape Estimation Using Object Local Grid Maps with Stereo. In *Proc. of the IEEE International Conference on Intelligent Transportation Systems*, pages 330–335, The Hague, The Netherlands, Oct. 2013.

[11] H. Badino, U. Franke, and R. Mester. Free Space Computation Using Stochastic Occupancy Grids and Dynamic Programming. In *Workshop on Dynamical Vision ICCV*, Rio de Janeiro, Brazil, Oct. 2007.

[12] H. Badino, U. Franke, and D. Pfeiffer. The Stixel World – A Compact Medium Level Representation of the 3D-World. In *DAGM Symposium on Pattern Recognition*, Jena, Germany, Sept. 2009.

[13] T. Bailey and H. Durrant-Whyte. Simultaneous Localization and Mapping. *IEEE Robotics & Automation Magazine*, 13(3):108–117, Sept. 2006.

[14] Y. Bar-Shalom, K. C. Chang, and H. A. P. Blom. Automatic Track Formation in Clutter with a Recursive Algorithm. In *Proc. of the 28th Conference on Decision and Control*, pages 1402–1408, Tampa, FL, USA, Dec. 1989.

[15] Y. Bar-Shalom, F. Daum, and J. Huang. The Probabilistic Data Association Filter. *IEEE Control Systems Magazine*, 29(6):82–100, Dec. 2009.

[16] Y. Bar-Shalom and T. E. Fortmann. *Tracking and Data Association*, volume 179 of *Mathematics in Science and Engineering.* Academic Press, Orlando, FL, USA, 1988.

[17] Y. Bar-Shalom and X.-R. Li. *Multitarget-Multisensor Tracking: Principles and Techniques.* Yaakov Bar-Shalom, Storrs, CT, USA, 1995.

[18] Y. Bar-Shalom, X.-R. Li, and T. Kirubarajan. *Estimation with Applications to Tracking and Navigation. Theory, Algorithms and Software.* Wiley-Interscience, New York City, NY, USA, 2001.

[19] Y. Bar-Shalom and E. Tse. Tracking in a Cluttered Environment with Probabilistic Data Association. *Automatica*, 11(5):451–460, Sept. 1975.

[20] C. Barrios, H. Himberg, Y. Motai, and A. Sadek. Multiple Model Framework of Adaptive Extended Kalman Filtering for Predicting Vehicle Location. In *Proc. of the IEEE International Conference on Intelligent Transportation Systems*, pages 1053–1059, Toronto, Canada, Sept. 2006.

[21] A. Barth. *Vehicle Tracking and Motion Estimation Based on Stereo Vision Sequences.* PhD thesis, Rheinische Friedrich-Wilhelms-Universität, Bonn, Germany, Sept. 2010.

[22] A. Barth and U. Franke. Where will the Oncoming Vehicle be the Next Second. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 1068–1073, Eindhoven, The Netherlands, June 2008.

[23] E. Bauer. *Modellprädiktive integrale Trajektorienplanung für Fahrerassistenzsysteme.* PhD thesis, Technische Universität Darmstadt, Darmstadt, Germany, in preparation.

[24] E. Bauer and U. Konigorski. Ein modellprädiktiver Querplanungsansatz zur Kollisionsvermeidung. In *AUTOREG*, Baden-Baden, Germany, June 2013.

[25] E. Bauer, F. Lotz, M. Pfromm, M. Schreier, S. Cieler, A. Eckert, A. Hohm, S. Lüke, P. Rieth, B. Abendroth, V. Willert, J. Adamy, R. Bruder, U. Konigorski, and H. Winner. PRORETA 3: An Integrated Approach to Collision Avoidance and Vehicle Automation. *at – Automatisierungstechnik*, 60(12):755–765, Dec. 2012.

[26] T. Bayes. An Essay Towards Solving a Problem in the Doctrine of Chances. *Philosophical Transactions*, 53:370–418, Jan. 1763.

[27] J. O. Berger. *Statistical Decision Theory and Bayesian Analysis.* Springer, New York City, NY, USA, second edition, 1985.

[28] H. Berndt and K. Dietmayer. Driver Intention Inference with Vehicle Onboard Sensors. In *IEEE International Conference on Vehicular Electronics and Safety*, pages 102–107, Pune, India, Nov. 2009.

[29] A. Berthelot, A. Tamke, T. Dang, and G. Breuel. Stochastic Situation Assessment in Advanced Driver Assistance System for Complex Multi-Objects Traffic Situations. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1180–1185, Vilamoura, Algarve, Portugal, Oct. 2012.

[30] C. M. Bishop. *Pattern Recognition and Machine Learning.* Springer, New York City, NY, USA, 2006.

[31] R. Biswas, B. Limketkai, S. Sanner, and S. Thrun. Towards Object Mapping in Non-Stationary Environments With Mobile Robots. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, pages 1014–1019, Lausanne, Switzerland, Oct. 2002.

[32] S. Biswas and B. C. Lovell. *Bézier and Splines in Image Processing and Machine Vision.* Springer, London, UK, 2008.

[33] S. Blackman. Multiple Hypothesis Tracking for Multiple Target Tracking. *IEEE Aerospace and Electronic Systems Magazine*, 19(1):5–18, Jan. 2004.

[34] S. Blackman and R. Popoli. *Design and Analysis of Modern Tracking Systems.* Artech House, Boston, MA, USA, 1999.

[35] A. Blake and M. Isard. *Active Contours.* Springer, London, UK, 1998.

[36] H. A. P. Blom and Y. Bar-Shalom. The Interacting Multiple Model Algorithm for Systems with Markovian Switching Coefficients. *IEEE Transactions on Automatic Control*, 33(8):780–783, Aug. 1988.

[37] H. A. P. Blom and E. A. Bloem. Probabilistic Data Association Avoiding Track Coalescence. *IEEE Transactions on Automatic Control*, 45(45):247–259, Feb. 2000.

[38] H. A. P. Blom and E. A. Bloem. Combining IMM and JPDA for Tracking Multiple Maneuvering Targets in Clutter. In *Proc. of the 5th IEEE International Conference on Information Fusion*, volume 1, pages 705–712, Annapolis, MD, USA, July 2002.

[39] S. Bonnin, T. H. Weisswange, F. Kummert, and J. Schmüdderich. General Behavior Prediction by a Combination of Scenario-Specific

Models. *IEEE Transactions on Intelligent Transportation Systems*, 15(4):1478–1488, Aug. 2014.

[40] E. Bouzouraa. *Belegungskartenbasierte Umfeldwahrnehmung in Kombination mit objektbasierten Ansätzen für Fahrerassistenzsysteme.* PhD thesis, Technische Universität München, München, Germany, 2011.

[41] M. E. Bouzouraa. Modellbasierte Freiraumverfolgung für Fahrerassistenzsysteme. In *8. Workshop Fahrerassistenzsysteme*, pages 67–75, Walting, Germany, Sept. 2012.

[42] M. E. Bouzouraa and U. Hofmann. Fusion of Occupancy Grid Mapping and Model Based Object Tracking for Driver Assistance Systems Using Laser and Radar Sensors. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 294–300, San Diego, CA, USA, June 2010.

[43] J. E. Bresenham. Algorithm for Computer Control of a Digital Plotter. *IBM Systems Journal*, 4(1):25–30, Jan. 1965.

[44] M. Brännström, E. Coelingh, and J. Sjöberg. Model-Based Threat Assessment for Avoiding Arbitrary Vehicle Collisions. *IEEE Transactions on Intelligent Transportation Systems*, 11(3):658–669, Sept. 2010.

[45] A. Broadhurst, S. Baker, and T. Kanade. Monte Carlo Road Safety Reasoning. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 319–324, Las Vegas, NV, USA, June 2005.

[46] K. H. Brodersen, C. S. Ong, K. E. Stephan, and J. M. Buhmann. The Balanced Accuracy and its Posterior Distribution. In *International Conference on Pattern Recognition*, pages 3121–3124, Istanbul, Turkey, Aug. 2010.

[47] A. Broggi, S. Cattani, M. Patander, M. Sabbatelli, and P. Zani. A full-3D Voxel-based Dynamic Obstacle Detection for Urban Scenario Using Stereo Vision. In *Proc. of the IEEE International Conference on Intelligent Transportation Systems*, pages 71–76, The Hague, The Netherlands, Oct. 2013.

[48] A. Broggi, A. Zelinsky, M. Parent, and C. E. Thorpe. Intelligent Vehicles. In *Handbook of Robotics*, chapter 51, pages 1175–1198. Springer, Berlin, Germany, 2008.

[49] W. Burgard and M. Hebert. World Modeling. In *Handbook of Robotics*, chapter 36, pages 853–869. Springer, Berlin, Germany, 2008.

[50] D. Caveney. Numerical Integration for Future Vehicle Path Prediction. In *Proc. of the American Control Conference*, pages 3906–3912, New York City, NY, USA, July 2007.

[51] P. Cerri and P. Grisleri. Free Space Detection on Highways Using Time Correlation Between Stabilized Sub-Pixel Precision IPM Images. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 2223–2228, Barcelona, Spain, Apr. 2005.

[52] S. Challa and K. Koks. Bayesian and Dempster-Shafer Fusion. *Sadhana*, 29:145–176, Apr. 2004.

[53] S. Challa, M. R. Morelande, D. Mušicki, and R. J. Evans. *Fundamentals of Object Tracking*. Cambridge University Press, Cambridge, England, 2011.

[54] C. Chen, C. Tay, C. Laugier, and K. Mekhnacha. Dynamic Environment Modeling with Gridmap: A Multiple-Object Tracking Application. In *Proc. of the International Conference on Control, Automation, Robotics and Vision*, pages 1–6, Singapore, Dec. 2006.

[55] H. Christensen and G. Hager. Sensing and Estimation. In *Springer Handbook of Robotics*, chapter 4, pages 87–107. Springer, Berlin, Germany, 2008.

[56] S. Cieler, U. Konigorski, S. Lüke, and H. Winner. Projekt PRORETA 3 – Sicherheit und Automation mit Assistenzsystemen. *ATZ – Automobiltechnische Zeitschrift*, pages 20–27, Oct. 2014.

[57] C. Coué, C. Pradalier, C. Laugier, T. Fraichard, and P. Bessière. Bayesian Occupancy Filtering for Multitarget Tracking: An Automotive Application. *The International Journal of Robotics Research*, 25(1):19–30, Jan. 2006.

[58] I. Dagli, M. Brost, and G. Breuel. Action Recognition and Prediction for Driver Assistance Systems Using Dynamic Belief Networks. In *Agent Technologies, Infrastructures, Tools, and Applications for E-Services*, pages 179–194. Springer, Berlin, Germany, 2003.

[59] L. Dagum and R. Menon. Open MP: An Industry-Standard API for Shared-Memory Programming. *IEEE Computational Science & Engineering*, 5(1):46–55, Jan. 1998.

[60] F. Damerow and J. Eggert. Predictive Risk Maps. In *Proc. of the IEEE International Conference on Intelligent Transportation Systems*, pages 703–710, Qingdao, China, Oct. 2014.

[61] R. Danescu and S. Nedevschi. A Particle-Based Solution for Modeling and Tracking Dynamic Digital Elevation Maps. *IEEE Transactions on Intelligent Transportation Systems*, 15(3):1002–1015, June 2014.

[62] R. Danescu, F. Oniga, and S. Nedevschi. Modeling and Tracking the Driving Environment with a Particle-Based Occupancy Grid. *IEEE Transactions on Intelligent Transportation Systems*, 12(4):1331–1342, Dec. 2011.

[63] R. Danescu, C. Pantilie, F. Oniga, and S. Nedevschi. Particle Grid Tracking System Stereovision Based Obstacle Perception in Driving Environments. *IEEE Intelligent Transportation Systems Magazine*, 4(1):6–20, Jan. 2012.

[64] M. Darms. *Eine Basis-Systemarchitektur zur Sensordatenfusion von Umfeldsensoren für Fahrerassistenzsysteme.* Number 653 in VDI Fortschrittsberichte (series 12). VDI, Düsseldorf, Germany, 2007.

[65] M. Darms, M. Komar, and S. Lüke. Map Based Road Boundary Estimation. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 609–614, San Diego, CA, USA, June 2010.

[66] C. de Boor. *A Practical Guide to Splines*, volume 27 of *Applied Mathematical Sciences*. Springer, New York, USA, revised edition, 1978.

[67] K. Dietmayer, A. Kirchner, and N. Kämpchen. Fusionsarchitekturen zur Umfeldwahrnehmung für zukünftige Fahrerassistenzsysteme. In *Fahrerassistenzsysteme mit maschineller Wahrnehmung*, chapter 4, pages 59–88. Springer, Berlin, Germany, 2005.

[68] A. Doucet, N. de Freitas, K. Murphy, and S. Russell. Rao-Blackwellised Particle Filtering for Dynamic Bayesian Networks. In *Proc. of the 16th Conference on Uncertainty in Artificial Intelligence*, pages 176–183, Stanford, CA, USA, June 2000.

[69] A. Doucet, S. Godsill, and C. Andrieu. On Sequential Monte Carlo Sampling Methods for Bayesian Filtering. *Statistics and Computing*, 10(3):197–208, July 2000.

[70] K. Dörner, W. Schwertberger, and E. Hipp. Bahnführungsassistenz für Nutzfahrzeuge. In *Handbuch Fahrerassistenzsysteme. Grundlagen, Komponenten und Systeme für aktive Sicherheit und Komfort*, chapter 53, pages 1009–1027. Springer Vieweg, Wiesbaden, Germany, third, revised and extended edition, 2015.

[71] N. E. Du Toit and J. W. Burdich. Probabilistic Collision Checking with Chance Constraints. *IEEE Transactions on Robotics*, 27(4):809–815, Aug. 2011.

[72] H. Durrant-Whyte and T. Bailey. Simultaneous Localization and Mapping. *IEEE Robotics & Automation Magazine*, 13(2):99–110, June 2006.

[73] J. Effertz. *Autonome Fahrzeugführung in urbaner Umgebung durch Kombination objekt- und kartenbasierter Umfeldmodelle.* PhD thesis, Technische Universität Carolo-Wilhelmina, Braunschweig, Germany, July 2009.

[74] J. Eggert. Predictive Risk Estimation for Intelligent ADAS Functions. In *Proc. of the IEEE International Conference on Intelligent Transportation Systems*, pages 711–718, Qingdao, China, Oct. 2014.

[75] A. Eidehall and L. Petersson. Statistical Thread Assessment for General Road Scenes Using Monte Carlo Sampling. *IEEE Transactions on Intelligent Transportation Systems*, 9(1):137–147, Mar. 2008.

[76] A. Elfes. Sonar-Based Real-World Mapping and Navigation. *Journal of Robotics and Automation*, 3(3):249–265, June 1987.

[77] A. Elfes. Using Occupancy Grids for Mobile Robot Perception and Navigation. *Computer*, 22(6):46–57, 1989.

[78] M. R. Endsley. Toward a Theory of Situation Awareness in Dynamic Systems. *Human Factors*, 37(1):32–64, Mar. 1995.

[79] A. Ess, K. Schindler, B. Leibe, and L. Van Gool. Object Detection and Tracking for Autonomous Navigation in Dynamic Environments. *International Journal of Robotics Research*, 29(14):1707–1725, Dec. 2010.

[80] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proc. of the International Conference on Knowledge Discovery and Data Mining*, pages 226–231, Portland, OR, USA, Aug. 1996.

[81] J. Farrell and M. Livstone. Exact Calculation of Discrete-Time Process Noise Statistics for Hybrid Continuous/Discrete Time Applications. In *Proc. of the 32nd Conference on Decision and Control*, pages 857–858, San Antonio, TX, USA, Dec. 1993.

[82] D. Ferguson, M. Darms, C. Urmson, and S. Kolski. Detection, Prediction, and Avoidance of Dynamic Obstacles in Urban Environments. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 1149–1154, Eindhoven, The Netherlands, June 2008.

[83] F. Flemisch, H. Winner, R. Bruder, and K. Bengler. Kooperative Fahrzeugführung. In *Handbuch Fahrerassistenzsysteme. Grundlagen, Komponenten und Systeme für aktive Sicherheit und Komfort*, chapter 58, pages 1103–1110. Springer Vieweg, Wiesbaden, Germany, third, revised and extended edition, 2015.

[84] T. E. Fortmann, Y. Bar-Shalom, and M. Scheffe. Sonar Tracking of Multiple Targets Using Joint Probabilistic Data Association. *IEEE Journal of Oceanic Engineering*, 8(3):173–184, July 1983.

[85] J. Fritsch, T. Kühnl, and F. Kummert. Monocular Road Terrain Detection by Combining Visual and Spatial Information. *IEEE Transactions on Intelligent Transportation Systems*, 15(4):1586–1596, Aug. 2014.

[86] R. Garcia, O. Aycard, T.-D. Vu, and M. Ahrholdt. High Level Sensor Data Fusion for Automotive Applications Using Occupancy Grids. In *Proc. of the Internatinal Conference on Control, Automation, Robotics and Vision*, pages 530–535, Hanoi, Vietnam, Dec. 2008.

[87] A. Gelb. *Applied Optimal Estimation*. The MIT Press, Cambridge, MA, USA, 1974.

[88] T. Gindele, R. Brechtel, and R. Dillmann. A Probabilistic Model for Estimating Driver Behaviors and Vehicle Trajectories in Traffic

Environments. In *Proc. of the IEEE International Conference on Intelligent Transportation Systems*, pages 1625–1631, Madeira Island, Portugal, Sept. 2010.

[89] R. Gopalan, T. Hong, M. Shneier, and R. Chellappa. A Learning Approach Towards Detection and Tracking of Lane Markings. *IEEE Transactions on Intelligent Transportation Systems*, 13(3):1088–1098, Sept. 2012.

[90] N. J. Gordon, D. J. Salmond, and A. F. M. Smith. Novel Approach to Nonlinear/Non-Gaussian Bayesian State Estimation. *Radar and Signal Processing*, 140(2):107–113, Apr. 1993.

[91] R. Grewe. *Optimierung der Repräsentation von Occupancy-Grids für Fahrerassistenzsysteme.* Number 780 in VDI Fortschrittsberichte (series 12). VDI, Düsseldorf, Germany, 2014.

[92] R. Grewe, A. Hohm, S. Hegemann, S. Lueke, and H. Winner. Towards a Generic and Efficient Environment Model for ADAS. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 316–321, Alcalá de Henares, Spain, June 2012.

[93] F. Gustafsson and A. J. Isaksson. Best Choice of Coordinate System for Tracking Coordinated Turn. In *Proc. of the Conference on Decision and Control*, pages 3145–3150, Kobe, Japan, Dec. 1996.

[94] J. M. Hammersley and D. C. Handscomb. *Monte Carlo Methods.* Methuen & Co Ltd, London, UK, 1964.

[95] F. Han, Y. Tan, and J. Eledath. Preceding Vehicle Trajectory Prediction by Multi-Cue Integration. In *Proc. of the Conference on Machine Vision Applications*, pages 575–578, Tokyo, Japan, May 2007.

[96] M. Hebert, C. Caillas, E. Krotkov, I. S. Kweon, and T. Kanade. Terrain Mapping for a Roving Planetary Explorer. In *Proc. IEEE Int. Conf. Robotics and Automation*, pages 997–1002, Scottsdale, AZ, USA, May 1989.

[97] I. Hensel. *Probabilistisch-logische Inferenz relationaler Situationsbeschreibungen aus Verkehrsbildfolgen.* PhD thesis, Karlsruhe Institute of Technology, Karlsruhe, Germany, Oct. 2013.

[98] C. Hermes, J. Einhaus, M. Hahn, C. Wöhler, and F. Kummert. Vehicle Tracking and Motion Prediction in Complex Urban Scenarios. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 26–33, San Diego, CA, USA, June 2010.

[99] C. Hermes, C. Wöhler, K. Schenk, and F. Kummert. Long-Term Vehicle Motion Prediction. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 652–657, Xi'an, China, June 2009.

[100] D. Hähnel, D. Schulz, and W. Burgard. Map Building with Mobile Robots in Populated Environments. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, pages 496–501, Lausanne, Switzerland, Oct. 2002.

[101] J. Hillenbrand, A. M. Spieker, and K. Kroschel. A Multilevel Collision Mitigation Approach – Its Situation Assessment, Decision Making, and Performance Tradeoffs. *IEEE Transactions on Intelligent Transportation Systems*, 7(4):528–540, Dec. 2006.

[102] T. Hülnhagen, I. Dengler, A. Tamke, T. Dang, and G. Breuel. Maneuver Recognition Using Probabilistic Finite-State Machines and Fuzzy Logic. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 65–70, San Diego, CA, USA, June 2010.

[103] M. Hülsen, J. M. Zöllner, and C. Weiss. Traffic Intersection Situation Description Ontology for Advanced Driver Assistance. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 993–999, Baden-Baden, Germany, June 2011.

[104] A. Hohm, F. Lotz, O. Fochler, S. Lüke, and H. Winner. Automated Driving in Real Traffic: From Current Technical Approaches Towards Architectural Perspectives. Technical report, SAE Technical Paper 2014-01-0159, 2014.

[105] A. Hohm, C. Wojek, B. Schiele, and H. Winner. Multi-Level Sensorfusion and Computer-Vision Algorithms within a Driver Assistance System for Avoiding Overtaking Accidents. In *FISITA World Automotive Congress*, pages 1–14, Munich, Germany, Sept. 2008.

[106] C. Høilund, T. B. Moeslund, and C. B. Madsen. Free Space Computation from Stochastic Occupancy Grids Based on Iconic Kalman Filtered Disparity Maps. In *Proc. of the International Conference on*

Computer Vision Theory and Applications, pages 164–167, Angers, France, May 2010.

[107] F. Homm, N. Kämpchen, J. Ota, and D. Burschka. Efficient Occupancy Grid Computation on the GPU with Lidar and Radar for Road Boundary Detection. In *Proc. of the IEEE Intelligent Vehicles Symposium*, San Diego, CA, USA, June 2010.

[108] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard. OctoMap: An Efficient Probabilistic 3D Mapping Framework Based on Octrees. *Autonomous Robots*, 34(3):189–206, Feb. 2013.

[109] A. Houénou, P. Bonnifait, and V. Cherfaoui. Risk Assessment for Collision Avoidance Systems. In *Proc. of the IEEE International Conference on Intelligent Transportation Systems*, pages 386–391, Qingdao, China, Oct. 2014.

[110] A. Houles and Y. Bar-Shalom. Multisensor Tracking of a Maneuvering Target in Clutter. *IEEE Transactions on Aerospace and Electronic Systems*, 25(2):176–189, Mar. 1989.

[111] B. Hummel. *Description Logic for Scene Understanding at the Example of Urban Road Intersections*. PhD thesis, Karlsruhe Institute of Technology, Karlsruhe, Germany, Apr. 2009.

[112] R. Isermann, R. Mannale, and K. Schmitt. Collision-Avoidance Systems PRORETA: Situation Analysis and Intervention Control. *Control Engineering Practice*, 20(11):1236–1246, Nov. 2012.

[113] R. Isermann and M. Münchhof. *Identification of Dynamic Systems. An Introduction with Applications*. Springer, Berlin, Germany, third edition, Sept. 2010.

[114] K. Ito and K. Xiong. Gaussian Filters for Nonlinear Filtering Problems. *IEEE Transactions on Automatic Control*, 45(5):910–927, May 2000.

[115] F. V. Jensen and T. D. Nielsen. *Bayesian Networks and Decision Graphs*. Information Science and Statistics. Springer, Berlin, Germany, second edition, 2007.

[116] B. Jähne. *Digitale Bildverarbeitung*. Springer, Berlin, Germany, sixth, revised and extended edition, 2005.

[117] S. J. Julier. The Scaled Unscented Transformation. In *Proc. of the IEEE American Control Conference*, volume 6, pages 4555–4559, Anchorage, AK, USA, May 2002.

[118] S. J. Julier and J. K. Uhlmann. A New Extension of the Kalman Filter to Nonlinear Systems. In *Proc. of SPIE3068: Signal Processing, Sensor Fusion, and Target Recognition VI*, volume 3068, pages 182–193, Orlando, FL, USA, July 1997.

[119] S. J. Julier and J. K. Uhlmann. Unscented Filtering and Nonlinear Estimation. *Proc. of the IEEE*, 92(3):401–422, Mar. 2004.

[120] R. E. Kálmán. A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME–Journal of Basic Engineering*, 82:35–45, 1960.

[121] J. Kang and M. Chung. Stereo-Vision Based Free Space and Obstacle Detection with Structural and Traversability Analysis Using Probabilistic Volume Polar Grid Map. In *Proc. of the IEEE International Conference on Robotics, Automation and Mechatronics*, pages 245–251, Qingdao, China, Sept. 2011.

[122] D. Kasper, G. Weidl, T. Dang, G. Breuel, A. Tamke, A. Wedel, and W. Rosenstiel. Object-Oriented Bayesian Network for Detection of Lane Change Maneuvers. *IEEE Intelligent Transportation Systems Magazine*, 4(1):1–10, Jan. 2012.

[123] R. Kastner. *Robust Path Identification for Driver Assistance.* Number 746 in VDI Fortschrittsberichte (series 12). VDI, Düsseldorf, Germany, 2011.

[124] E. Käfer. *Situationsklassifikation und Bewegungsprognose in Verkehrssituationen mit mehreren Fahrzeugen.* PhD thesis, Bielefeld University, Bielefeld, Germany, Feb. 2013.

[125] E. Käfer, C. Hermes, C. Wöhler, H. Ritter, and F. Kummert. Recognition of Situation Classes at Road Intersections. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 3960–3965, Anchorage, AK, USA, May 2010.

[126] T. Kühnl. *Road Terrain Detection for Advanced Driver Assistance Systems.* PhD thesis, Bielefeld University, Bielefeld, Germany, May 2013.

[127] T. Kirubarajan and Y. Bar-Shalom. Probabilistic Data Association Techniques for Target Tracking in Clutter. *Proc. of the IEEE*, 92(3):536–557, Mar. 2004.

[128] S. Klingelschmitt, M. Platho, H.-M. Groß, V. Willert, and J. Eggert. Combining Behavior and Situation Information for Reliably Estimating Multiple Intentions. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 388–393, Dearborn, MI, USA, June 2014.

[129] N. Kämpchen, B. Schiele, and K. Dietmayer. Situation Assessment of an Autonomous Emergency Brake for Arbitrary Vehicle-to-Vehicle Collision Scenarios. *IEEE Transactions on Intelligent Transportation Systems*, 10(4):678–687, Dec. 2009.

[130] D. Koller and N. Friedman. *Probabilistic Graphical Models. Principles and Techniques.* The MIT Press, Cambridge, MA, USA, 2009.

[131] M. Konrad, M. Szczot, and K. Dietmayer. Road Course Estimation in Occupancy Grids. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 412 – 417, San Diego, CA, USA, June 2010.

[132] M. Konrad, M. Szczot, F. Schüle, and K. Dietmayer. Generic Grid Mapping for Road Course Estimation. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 851–856, Baden-Baden, Germany, June 2011.

[133] K. B. Korb and A. E. Nicholson. *Bayesian Artificial Intelligence.* Computer Science and Data Analysis. Chapman & Hall/CRC Press, Boca Raton, FL, USA, 2004.

[134] G. Kraetzschmar, G. P. Gassull, and K. Uhl. Probabilistic Quadtrees for Variable-Resolution Mapping of Large Environments. In *Proc. of the 5th IFAC/EURON Symposium on Intelligent Autonomous Vehicles*, Lisbon, Portugal, July 2004.

[135] S. Kubota, T. Nakano, and Y. Okamoto. A Global Optimization Algorithm for Real-Time On-Board Stereo Obstacle Detection Systems. In *Proc. of the IEEE Intelligent Vehicles Symposium*, Istanbul, Turkey, June 2007.

[136] Z. Kulpa. Area and Perimeter Measurement of Blobs in Discrete Binary Images. *Computer Graphics and Image Processing*, 6(5):434–451, Oct. 1977.

[137] M. Kurdej, J. Moras, V. Cherfaoui, and P. Bonnifait. Map-Aided Evidential Grids for Driving Scene Understanding. *IEEE Intelligent Transportation Systems Magazine*, 7(1):30–41, Jan. 2015.

[138] A. Lambert, D. Gruyer, and G. S. Pierre. A Fast Monte Carlo Algorithm for Collision Probability Estimation. In *Proc. of the International Conference on Control, Automation, Robotics and Vision*, pages 406–411, Hanoi, Vietnam, Dec. 2008.

[139] P.-S. Laplace. *Essai philosophique sur les probabilités.* Courcier, Paris, France, 1814.

[140] H. Lategahn, W. Derendarz, T. Graf, B. Kitt, and J. Effertz. Occupancy Grid Computation from Dense Stereo and Sparse Structure and Motion Points for Automotive Applications. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 819–824, San Diego, CA, USA, June 2010.

[141] H. Lategahn, T. Graf, C. Hasberg, B. Kitt, and J. Effertz. Mapping in Dynamic Environments Using Stereo Vision. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 150–156, Baden-Baden, Germany, June 2011.

[142] C. Laugier, I. E. Paromtchik, M. Perrollaz, M. Yong, J.-D. Yoder, C. Tay, K. Mekhnacha, and A. Nègre. Probabilistic Analysis of Dynamic Scenes and Collision Risks Assessment to Improve Driving Safety. *IEEE Intelligent Transportation Systems Magazine*, 3(4):4–19, 2011.

[143] S. L. Lauritzen and D. J. Spiegelhalter. Local Computations with Probabilities on Graphical Structures and Their Application to Expert Systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, 50(2):157–224, 1988.

[144] A. Lawitzky, D. Althoff, C. F. Passenberg, G. Tanzmeier, D. Wollherr, and M. Buss. Interactive Scene Prediction for Automotive Applications. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 1028–1033, Gold Coast, Australia, June 2013.

[145] J. Leal. *Stochastic Environment Representation.* PhD thesis, University of Sydney, Sydney, Australia, Jan. 2003.

[146] T. Lefebvre, H. Bruyninckx, and J. D. Schutter. Kalman Filters for Nonlinear Systems: a Comparison of Performance. *International Journal of Control*, 77(7):639–653, May 2004.

[147] S. Lefèvre, C. Laugier, and J. Ibañez-Guzmán. Exploiting Map Information for Driver Intention Estimation at Road Intersections. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 583–588, Baden-Baden, Germany, June 2011.

[148] S. Lefèvre, C. Laugier, and J. Ibañez-Guzmán. Risk Assessment at Road Intersections: Comparing Intention and Expectation. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 165–171, Alcalá de Henares, Spain, June 2012.

[149] S. Lefèvre, D. Vasquez, and C. Laugier. A Survey on Motion Prediction and Risk Assessment for Intelligent Vehicles. *ROBOMECH Journal*, 1(1):1–14, July 2014.

[150] X.-R. Li and Y. Bar-Shalom. Multiple-Model Estimation with Variable Structure. *IEEE Transactions on Automatic Control*, 41(4):478–493, Apr. 1996.

[151] X.-R. Li and V. P. Jilkov. Survey of Maneuvering Target Tracking. Part IV: Decision-Based Methods. In *Proc. of the SPIE Conference on Signal and Data Processing of Small Targets*, pages 511–534, Orlando, FL, USA, Apr. 2002.

[152] X.-R. Li and V. P. Jilkov. Survey of Maneuvering Target Tracking. Part I: Dynamic Models. *IEEE Transactions on Aerospace and Electronic Systems*, 39(4):1333–1364, Oct. 2003.

[153] X.-R. Li and V. P. Jilkov. Survey of Maneuvering Target Tracking. Part V: Multiple-Model Methods. *IEEE Transactions on Aerospace and Electronic Systems*, 41(4):1255–1321, Oct. 2005.

[154] F. Liu. *Objektverfolgung durch Fusion von Radar- und Monokameradaten auf Merkmalsebene für zukünftige Fahrerassistenzsysteme.* PhD thesis, Karlsruhe Institute of Technology, Karlsruhe, Germany, Dec. 2009.

[155] S. Lüke, D. Waldbauer, M. Darms, and M. Komar. An Assistance System for Construction Sites. In *6th IFAC Symposium Advances in Automotive Control*, pages 761–767, Munich, Germany, July 2010.

[156] H. Loose and U. Franke. B-Spline-Based Road Model for 3D Lane Recognition. In *Proc. of the IEEE International Conference on Intelligent Transportation Systems*, pages 91–98, Madeira Island, Portugal, Sept. 2010.

[157] F. Lotz. System Architectures for Automated Vehicle Guidance Concepts. In *Automotive Systems Engineering*, chapter 3, pages 39–61. Springer, Berlin, Germany, 2013.

[158] F. Lotz. *Entwicklung einer Referenzarchitektur für die assistierte und automatisierte Fahrzeugführung mit Fahrereinbindung.* PhD thesis, Technische Universität Darmstadt, Darmstadt, Germany, in preparation.

[159] B. D. Lucas and T. Kanade. An Iterative Image Registration Technique with an Application to Stereo Vision. In *Proc. of the International Joint Conference on Artificial Intelligence*, pages 674–679, Vancouver, BC, USA, Aug. 1981.

[160] C. Lundquist and T. B. Schön. Estimation of the Free Space in Front of a Moving Vehicle. In *Proc. of the SAE World Congress*, Detroit, MI, USA, Apr. 2009.

[161] J. Lunze. *Regelungstechnik 2. Mehrgrößensysteme, Digitale Regelung.* Springer, Berlin, Germany, sixth, revised edition, 2010.

[162] P. Lytrivis, G. Thomaidis, and A. Amditis. Cooperative Path Prediction in Vehicular Environments. In *Proc. of the IEEE International Conference on Intelligent Transportation Systems*, pages 803–808, Beijing, China, Oct. 2008.

[163] R. Madhavan and C. Schlenoff. The Effect of Process Models on Short-term Prediction of Moving Objects for Unmanned Ground Vehicles . In *Proc. of the IEEE International Conference on Intelligent Transportation Systems*, pages 471–476, Washington, D.C., USA, Oct. 2004.

[164] P. C. Mahalanobis. On the Generalized Distance in Statistics. *Proc. of the National Institute of Sciences of India*, 2(1):49–55, Apr. 1936.

[165] R. Mahler. PHD Filters of Higher Order in Target Number. *IEEE Transactions on Aerospace and Electronic Systems*, 43(4):1523–1543, Oct. 2007.

[166] R. Mahler. "Statistics 102" for Multisouce-Multitarget Detection and Tracking. *IEEE International Journal on Selected Topics in Signal Processing*, 7(3):376–389, June 2013.

[167] E. Mazor, A. Averbuch, Y. Bar-Shalom, and J. Dayan. Interacting Multiple Model Methods in Target Tracking: A Survey. *IEEE Transactions on Aerospace and Electronic Systems*, 34(1):103–123, Jan. 1998.

[168] F. Meyer. Topographic Distance and Watershed Lines. *Signal Processing*, 38(1):113–125, July 1994.

[169] D. Meyer-Delius, C. Plagemann, and W. Burgard. Probabilistic Situation Recognition for Vehicular Traffic Scenes. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 459–464, Kobe, Japan, May 2009.

[170] M. Mählisch. *Filtersynthese zur simultanen Minimierung von Existenz-, Assoziations- und Zustandsunsicherheiten in der Fahrzeugumfelderfassung mit heterogenen Sensordaten*. PhD thesis, Ulm University, Ulm, Germany, Aug. 2009.

[171] T. P. Michalke. *Task-Dependent Scene Interpretation in Driver Assistance*. Number 718 in VDI Fortschrittsberichte (series 12). VDI, Düsseldorf, Germany, 2010.

[172] T. P. Michalke, C. Gläser, L. Bürkle, and F. Niewels. The Narrow Road Assistant – Next Generation Advanced Driver Assistance in Inner-City. In *Proc. of the IEEE International Conference on Intelligent Transportation Systems*, pages 2173–2180, The Hague, The Netherlands, Oct. 2013.

[173] N. C. Mitsou and C. S. Tzafestas. An Introduction to the Problem of Mapping in Dynamic Environments. In *Tools in Artificial Intelligence*, chapter 17, pages 295–314. InTech, Rijeka, Croatia, Aug. 2008.

[174] J. Miura, Y. Negishi, and Y. Shirai. Mobile Robot Map Generation by Integrating Omnidirectional Stereo and Laser Range Finder. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 250–255, Lausanne, Switzerland, Oct. 2002.

[175] H. P. Moravec. Robot Spatial Perception by Stereoscopic Vision and 3D Evidence Grids. CMU-RI-TR-96-34, Robotics Institute Carnegie Mellon University, Pittsburgh, PA, USA, Sept. 1996.

[176] B. Morris, A. Doshi, and M. Trivedi. Lane Change Intent Prediction for Driver Assistance: On Road Design and Evaluation. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 858–901, Baden-Baden, Germany, June 2011.

[177] M. Munz. *Generisches Sensorfusionsframework zur gleichzeitigen Zustands- und Existenzschätzung für die Fahrzeugumfelderkennung.* PhD thesis, Ulm University, Ulm, Germany, July 2011.

[178] K. P. Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning.* PhD thesis, University of Berkeley, Berkeley, CA, USA, July 2002.

[179] D. Murray and J. J. Little. Using Real-Time Stereo Vision for Mobile Robot Navigation. *Autonomous Robots*, 8(2):161–171, Apr. 2000.

[180] D. Mušicki. Track Score and Target Existence. In *Proc. of the International Conference on Information Fusion*, pages 1–7, Florence, Italy, July 2006.

[181] H.-H. Nagel, W. Enkelmann, and G. Struck. FhG-Co-Driver: From Map-Guided Automatic Driving by Machine Vision to a Cooperative Driver Support. *Mathematical and Computer Modelling*, 22(4-7):185–212, Aug. 1995.

[182] R. E. Neapolitan. *Learning Bayesian Networks.* Artificial Intelligence. Prentice-Hall, Chicago, IL, USA, 2003.

[183] T.-N. Nguyen, M.-M. Meinecke, M. Tornow, and B. Michaelis. Optimized Grid-Based Environment Perception in Advanced Driver Assistance Systems. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 425–430, Xi'an, Shaanxi, China, June 2009.

[184] T.-N. Nguyen, B. Michaelis, A. Al-Hamadi, M. Tornow, and M.-M. Meinecke. Stereo-Camera-Based Urban Environment Perception Using Occupancy Grid and Object Tracking. *IEEE Transactions on Intelligent Transportation Systems*, 13(1):154–165, Mar. 2012.

[185] J.-M. Nigro, S. Loriette-Rougegrez, and M. Rombaut. Driving Situation Recognition with Uncertainty Management and Rule-Based Systems. *Engineering Applications of Artificial Intelligence*, 15(3-4):217–228, June 2002.

[186] D. Nuss, M. Stuebler, and K. Dietmayer. Consistent Environment Modeling by use of Occupancy Grid Maps, Digital Road Maps, and Multi-Object Tracking. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 1371–1377, Dearborn, MI, USA, June 2014.

[187] B. Øksendal. *Stochastic Differential Equations. An Introduction with Applications.* Springer, Berlin, Germany, sixth edition, 2013.

[188] F. Oniga and S. Nedevschi. Processing Dense Stereo Data Using Elevation Maps: Road Surface, Traffic Isle, and Obstacle Detection. *IEEE Transactions on Vehicular Technology*, 59(3):1172–1182, Mar. 2010.

[189] F. C. Papst. Verkehrsknotendetektion aus parametrischen Freiraum- und Gridkarten. Diploma Thesis, Technische Universität Darmstadt, Darmstadt, Germany, Nov. 2013.

[190] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann, San Francisco, CA, USA, second, revised edition, 1988.

[191] D. Petrich, T. Dang, G. Breuel, and C. Stiller. Assessing Map-Based Maneuver Hypothesis Using Probabilistic Methods and Evidence Theory. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 995–1000, Qingdao, China, Oct. 2014.

[192] D. Petrich, T. Dang, D. Kasper, G. Breuel, and C. Stiller. Map-Based Long Term Motion Prediction for Vehicles in Traffic Environments. In *Proc. of the IEEE International Conference on Intelligent Transportation Systems*, pages 2166–2172, The Hague, The Netherlands, Oct. 2013.

[193] A. Petrovskaya and S. Thrun. Model Based Vehicle Detection and Tracking for Autonomous Urban Driving. *Autonomous Robots*, 26(2-3):123–139, Apr. 2009.

[194] D. Pfeiffer and U. Franke. Modeling Dynamic 3D Environments by Means of the Stixel World. *IEEE Intelligent Transportation Systems Magazine*, 3(3):24–36, 2011.

[195] D. Pfeiffer and U. Franke. Towards a Global Optimal Multi-Layer Stixel Representation of Dense 3D Data. In *Proc. of the British Machine Vision Conference*, pages 1–12, Dundee, Scotland, Aug. 2011.

[196] M. Pfromm. *Menschbezogener Umgang mit Systemstörungen bei teilautomatisierter Fahrzeugführung.* PhD thesis, Technische Universität Darmstadt, Darmstadt, Germany, in preparation.

[197] M. Pfromm, S. Cieler, and R. Bruder. Driver Assistance via Optical Information with Spatial Reference. In *Proc. of the IEEE International Conference on Intelligent Transportation Systems*, pages 2006–2011, The Hague, The Netherlands, Oct. 2013.

[198] M. Platho, H.-M. Groß, and J. Eggert. Learning Driving Situations and Behavior Models from Data. In *Proc. of the IEEE International Conference on Intelligent Transportation Systems*, pages 276–281, The Hague, The Netherlands, Oct. 2013.

[199] A. Polychronopoulos, M. Tsogas, A. Amditis, U. Scheunert, L. Andreone, and F. Tango. Dynamic Situation and Threat Assessment for Collision Warning Systems: The EUCLIDE Approach. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 636–641, Parma, Italy, June 2004.

[200] A. Polychronopoulos, M. Tsogas, A. J. Amditis, and L. Andreone. Sensor Fusion for Predicting Vehicles' Path for Collision Avoidance Systems. *IEEE Transactions on Intelligent Transportation Systems*, 8(3):549–562, Sept. 2007.

[201] Á. C. Prat. *Sensordatenfusion und Bildverarbeitung zur Objekt- und Gefahrenerkennung.* PhD thesis, Technische Universität Carolo-Wilhelmina, Braunschweig, Germany, Nov. 2010.

[202] Y. Qin. Optimierung eines Systems zur Kritikalitätsbewertung von Fahrsituationen. Master's thesis, Technische Universität Darmstadt, Darmstadt, Germany, Jan. 2014.

[203] L. R. Rabiner. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Proc. of the IEEE*, 77(2):257–286, Feb. 1989.

[204] R. Rajamani. *Vehicle Dynamics and Control.* Springer, New York City, NY, USA, second edition, 2012.

[205] M. Reichel, M. E. Bouzouraa, A. Siegel, K.-H. Siedersberger, and M. Maurer. Erweiterte Umfelderkennung und Nutzung einer Ausweichanalyse als Grundlage einer aktiven Gefahrenbremsung. In *AAET 2010 – Automatisierungssyteme, Assistenzsysteme und eingebettete Systeme für Transportmittel*, pages 150–169, Braunschweig, Germany, Feb. 2010.

[206] D. B. Reid. An Algorithm for Tracking Multiple Targets. *IEEE Transactions on Automatic Control*, 24(6):843–854, Dec. 1979.

[207] M. Ribo and A. Pinz. A Comparison of Three Uncertainty Calculi for Building Sonar-Based Occupancy Grids. *Robotics and Autonomous Systems*, 35(3-4):201–209, June 2001.

[208] B. Ristic, S. Arulampalam, and N. Gordon. *Beyond the Kalman Filter. Particle Filter for Tracking Applications.* Artech House, Boston, MA, USA, 2004.

[209] P. L. Rosin. Measuring Rectangularity. *Machine Vision and Applications*, 11(4):191–196, Dec. 1999.

[210] S. Russell and P. Norvig. *Artificial Intelligence. A Modern Approach.* Prentice-Hall, Upper Saddle River, NJ, USA, third edition, 2010.

[211] C. Schlenoff, R. Madhavan, and Z. Kootbally. PRIDE: A Hierarchical, Integrated Prediction Framework for Autonomous On-Road Driving. In *Proc. of the IEEE International Confererence on Robotics and Automation*, pages 2348–2353, Orlando, FL, USA, May 2006.

[212] M. Schmid, S. Ates, J. Dickmann, F. v. Hundelshausen, and H.-J. Wuensche. Parking Space Detection with Hierarchical Dynamic Occupancy Grids. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 254–259, Baden-Baden, Germany, June 2011.

[213] M. R. Schmid, M. Maehlisch, J. Dickmann, and H.-J. Wuensche. Dynamic Level of Detail 3D Occupancy Grids for Automotive Use. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 269–274, San Diego, CA, USA, June 2010.

[214] K. Schmitt. *Situationsanalyse für ein Fahrerassistenzsystem zur Vermeidung von Überholunfällen auf Landstraßen.* PhD thesis, Technische Universität Darmstadt, Darmstadt, Germany, Sept. 2012.

[215] K. Schmitt, R. Mannale, and R. Isermann. Situation Analysis, Warnings and Emergency Braking for Collision Avoidance in Overtaking Situations – PRORETA 2. In *IFAC Symposium Advances in Automotive Control*, pages 744–749, Munich, Germany, July 2010.

[216] J. Schneider, A. Wilde, and K. Naab. Probabilistic Approach for Modeling and Identifying Driving Situations. In *IEEE Intelligent Vehicles Symposium*, pages 343–348, Eindhoven, The Netherlands, June 2008.

[217] M. Schreier. Modeling and Adaptive Control of a Quadrotor. In *Proc. of the IEEE International Conference on Mechatronics and Automation*, pages 383–390, Chengdu, China, Aug. 2012.

[218] M. Schreier. Quaternion-Based Adaptive Attitude Control Schemes for Quadrotor Systems. *International Journal of Mechatronics and Automation*, 3(4):217–229, Dec. 2013.

[219] M. Schreier and E. Bauer. Projekt PRORETA 3 – Ein integrales Fahrerassistenzsystem zur Kollisionsvermeidung und Fahrzeugautomatisierung. In *12. Kooperationsforum Fahrerassistenzsysteme*, Aschaffenburg, Germany, May 2015.

[220] M. Schreier, E. Bauer, J. Adamy, and U. Konigorski. Das Fahrerassistenzsystem PRORETA 3: Umgebungsrepräsentation und Trajektorienplanung. In *48. Regelungstechnisches Kolloquium*, Boppard, Germany, Feb. 2014.

[221] M. Schreier and V. Willert. Robust Free Space Detection in Occupancy Grid Maps by Methods of Image Analysis and Dynamic B-Spline Contour Tracking. In *Proc. of the IEEE International Conference on Intelligent Transportation Systems*, pages 514–521, Anchorage, AK, USA, Sept. 2012.

[222] M. Schreier, V. Willert, and J. Adamy. An Integrated Approach to Maneuver-Based Trajectory Prediction and Criticality Assessment in Arbitrary Road Environments. *IEEE Transactions on Intelligent Transportation Systems*, accepted.

[223] M. Schreier, V. Willert, and J. Adamy. From Grid Maps to Parametric Free Space Maps – A Highly Compact, Generic Environment Representation for ADAS. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 938–944, Gold Coast, Australia, June 2013.

[224] M. Schreier, V. Willert, and J. Adamy. Bayesian, Maneuver-Based, Long-Term Trajectory Prediction and Criticality Assessment for Driver Assistance Systems. In *Proc. of the IEEE International Conference on Intelligent Transportation Systems*, pages 334–341, Qingdao, China, Oct. 2014.

[225] M. Schreier, V. Willert, and J. Adamy. Grid Mapping in Dynamic Road Environments: Classification of Dynamic Cell Hypothesis via Tracking. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 3995–4002, Hong Kong, China, June 2014.

[226] M. Schreier, V. Willert, and J. Adamy. Compact Representation of Dynamic Driving Environments for ADAS by Parametric Free Space and Dynamic Object Maps. *IEEE Transactions on Intelligent Transportation Systems*, PP(99):1–18, Sept. 2015.

[227] R. Schubert, E. Richter, and G. Wanielik. Comparison and Evaluation of Advanced Motion Models for Vehicle Tracking. In *Proc. of the IEEE International Conference on Information Fusion*, pages 1–6, Cologne, Germany, June 2008.

[228] R. Schubert, K. Schulze, and G. Wanielik. Situation Assessment for Automatic Lane-Change Maneuvers. *IEEE Transactions on Intelligent Transportation Systems*, 11(3):607–616, Sept. 2010.

[229] K. Schueler, T. Weiherer, E. Bouzouraa, and U. Hofmann. 360 Degree Multi Sensor Fusion for Static and Dynamic Obstacles. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 692–697, Alcalá de Henares, Spain, June 2012.

[230] J. Seitz. Parametrische Freiraumkarten in der Kritikalitätsbewertung von Fahrsituationen. Bachelor's Thesis, Technische Universität Darmstadt, Darmstadt, Germany, Sept. 2013.

[231] R. Sferco, Y. Page, J.-Y. Le Coz, and P. Fay. Potential Effectiveness of Electronic Stability Programs (ESP) – What European Field Studies Tell Us. In *Proc. of the International Technical Conference*

on the Enhanced Safety of Vehicles, pages 1–10, Amsterdam, The Netherlands, June 2001.

[232] D. Simon. *Optimal State Estimation: Kalman, H_∞, and Nonlinear Approaches.* John Wiley & Sons, Hoboken, NJ, USA, 2006.

[233] S. Sivaraman and M. M. Trivedi. Looking at Vehicles on the Road: A Survey of Vision-Based Vehicle Detection, Tracking, and Behavior Analysis. *IEEE Transactions on Intelligent Transportation Systems*, 14(4):1773–1795, Dec. 2013.

[234] P. Soille. *Morphological Image Analysis. Principles and Applications.* Springer, Berlin, Germany, second edition, 2004.

[235] M. Sonka, V. Hlavac, and R. Boyle. *Image Processing, Analysis, and Machine Vision.* Brooks/Cole Publishing Company, Pacific Grove, CA, USA, second edition, 1999.

[236] J. Sörstedt, L. Svensson, F. Sandblom, and L. Hammarstrand. A New Vehicle Motion Model for Improved Predictions and Situations Assessment. *IEEE Transactions on Intelligent Transportation Systems*, 12(4):1209–1219, Dec. 2011.

[237] C. Stachniss and W. Burgard. Mobile Robot Mapping and Localization in Non-Static Environments. In *Proc. of the National Conference on Artificial Intelligence*, pages 1324–1329, Pittsburgh, PA, USA, July 2005.

[238] Statistisches Bundesamt. *Verkehr. Verkehrsunfälle.* Wiesbaden, Germany, Nov. 2014.

[239] U. Stählin. *Eingriffsentscheidung für ein Fahrerassistenzsystem zur Unfallvermeidung.* PhD thesis, Technische Universität Darmstadt, Darmstadt, Germany, June 2008.

[240] D. Stüker. *Heterogene Sensordatenfusion zur robusten Objektverfolgung im automobilen Straßenverkehr.* PhD thesis, Carl von Ossietzky-Universität, Oldenburg, Germany, Feb. 2004.

[241] S. Suzuki and K. Abe. Topological Structural Analysis of Digitized Binary Images by Border Following. *Computer Vision, Graphics, and Image Processing*, 30(1):32–46, Apr. 1985.

[242] A. Tamke, T. Dang, and G. Breuel. A Flexible Method for Criticality Assessment in Driver Assistance Systems. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 697–702, Baden-Baden, Germany, June 2011.

[243] G. Tanzmeister, M. Friedl, A. Lawitzky, D. Wollherr, and M. Buss. Road Course Estimation in Unknown, Structured Environments. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 630–635, Gold Coast, Australia, June 2013.

[244] G. Tanzmeister, J. Thomas, D. Wollherr, and M. Buss. Grid-based Mapping and Tracking in Dynamic Environments Using a Uniform Evidential Environment Representation. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 6090–6095, Hong Kong, China, June 2014.

[245] S. Thrun. Robotic Mapping: A Survey. In *Exploring Artificial Intelligence in the New Millennium*, pages 1–35. Morgan Kaufmann, 2002.

[246] S. Thrun. Learning Occupancy Grid Maps with Forward Sensor Models. *Autonomous Robots*, 15(2):111–127, Sept. 2003.

[247] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics.* Intelligent Robotics and Autonomous Agents. The MIT Press, Cambridge, MA, USA, 2006.

[248] J. R. Treat. Tri-level Study of the Causes of Traffic Accidents: An Overview of Final Results. In *Proc. of the Conference of the American Association for Automotive Medicine*, pages 391–403, Lake Bluff, IL, USA, Sept. 1977.

[249] R. Triebel, P. Pfaff, and W. Burgard. Multi-Level Surface Maps for Outdoor Terrain Mapping and Loop Closing. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2276–2282, Beijing, China, Oct. 2006.

[250] M. Tsogas, X. Dai, G. Thomaidis, P. Lytrivis, and A. Amditis. Detection of Maneuvers Using Evidence Theory. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 126–131, Eindhoven, The Netherlands, June 2008.

[251] J. K. Tugnait. Tracking of Multiple Maneuvering Targets in Clutter Using Multiple Sensors, IMM, and JPDA Coupled Filtering. *IEEE Transactions on Aerospace and Electronic Systems*, 40(1):320–330, Jan. 2004.

[252] R. Van der Merwe. The Square-Root Unscented Kalman Filter for State and Parameter-Estimation. In *Proc. of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 6, pages 3461–3464, Salt Lake City, UT, USA, May 2001.

[253] R. Van der Merwe. *Sigma-Point Kalman Filters for Probabilistic Inference in Dynamic State-Space Models.* PhD thesis, OGI School of Science & Engineering, Hillsboro, OR, USA, Apr. 2004.

[254] D. Vasquez. *Incremental Learning for Motion Prediction of Pedestrians and Vehicles*, volume 64 of *Tracts in Advanced Robotics.* Springer, Berlin, 2010.

[255] D. Vasquez and T. Fraichard. Motion Prediction for Moving Objects: A Statistical Approach. In *Proc. of the IEEE International Conference on Robotics and Automation*, volume 4, pages 3931–3936, New Orleans, LA, USA, Apr. 2004.

[256] D. Vasquez, T. Fraichard, and C. Laugier. Incremental Learning of Statistical Motion Patterns With Growing Hidden Markov Models. *IEEE Transactions on Intelligent Transportation Systems*, 10(3):403–416, Sept. 2009.

[257] A. Vatavu and S. Nedevschi. Real-Time Modeling of Dynamic Environments in Traffic Scenarios Using a Stereo-Vision System. In *Proc. of the IEEE International Conference on Intelligent Transportation Systems*, pages 722–727, Anchorage, USA, Sept. 2012.

[258] Vägverket. *Safe Traffic. Vision Zero on the Move.* Swedish Road Administration, Borlänge, Sweden, Mar. 2006.

[259] T.-D. Vu and O. Aycard. Laser-based Detection and Tracking Moving Objects Using Data-Driven Markov Chain Monte Carlo. In *Proc. of the IEEE International Conference on Robotics and Automation*, pages 3800–3806, Kobe, Japan, May 2009.

[260] T.-D. Vu, O. Aycard, and N. Appenrodt. Online Localization and Mapping with Moving Object Tracking in Dynamic Outdoor Environments. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 190–195, Istanbul, Turkey, June 2007.

[261] C.-C. Wang. *Simultaneous Localization, Mapping and Moving Object Tracking*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, USA, Apr. 2004.

[262] C.-C. Wang and C. Thorpe. Simultaneous Localization and Mapping with Detection and Tracking of Moving Objects. In *Proc. of the IEEE International Conference on Robotics and Automation*, volume 3, pages 2918–2924, Washington, D.C., USA, May 2002.

[263] C.-C. Wang, C. Thorpe, S. Thrun, M. Hebert, and H. Durrant-Whyte. Simultaneous Localization, Mapping and Moving Object Tracking. *The International Journal of Robotics Research*, 26(9):889–916, Sept. 2007.

[264] Y. Wang, E. K. Teoh, and D. Shen. Lane Detection and Tracking using B-Snake. *Image and Vision Computing*, 22(4):269–280, Apr. 2004.

[265] A. Wedel, H. Badino, C. Rabe, H. Loose, U. Franke, and D. Cremers. B-Spline Modeling of Road Surfaces with an Application to Free-Space Estimation. *IEEE Transactions on Intelligent Transportation Systems*, 10(4):572–583, Dec. 2009.

[266] G. Wege. Probabilistische Langzeit-Trajektorienprädiktion für Fahrerassistenzsysteme. Diploma Thesis, Technische Universität Darmstadt, Darmstadt, Germany, May 2013.

[267] T. Weiherer, E. Bouzouraa, and U. Hofmann. An Interval Based Representation of Occupancy Information for Driver Assistance Systems. In *Proc. of the IEEE International Conference on Intelligent Transportation Systems*, pages 21–27, The Hague, The Netherlands, Oct. 2013.

[268] K. Weiss, N. Kämpchen, and A. Kirchner. Multiple-Model Tracking for the Detection of Lane Change Maneuvers. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 937–942, Parma, Italy, June 2004.

[269] T. Weiss, B. Schiele, and K. Dietmayer. Robust Driving Path Detection in Urban and Highway Scenarios Using a Laser Scanner and Online Occupancy Grids. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 184–189, Istanbul, Turkey, June 2007.

[270] J. Wiest, M. Höffken, U. Kreßel, and K. Dietmayer. Probabilistic Trajectory Prediction with Gaussian Mixture Models. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 141–146, Alcalá de Henares, Spain, June 2012.

[271] V. Willert and J. Eggert. A Stochastic Dynamical System for Optical Flow Estimation. In *Proc. of the 12th IEEE International Conference on Computer Vision, 4th Workshop on Dynamical Vision*, pages 711–718, Kyoto, Japan, Sept. 2009.

[272] H. Winner. Grundlagen von Frontalkollisionsschutzsystemen. In *Handbuch Fahrerassistenzsysteme. Grundlagen, Komponenten und Systeme für aktive Sicherheit und Komfort*, chapter 47, pages 893–912. Springer Vieweg, Wiesbaden, Germany, third, revised and extended edition, 2015.

[273] H. Winner, S. Hakuli, F. Lotz, and C. Singer. *Handbuch Fahrerassistenzsysteme. Grundlagen, Komponenten und Systeme für aktive Sicherheit und Komfort.* Springer Vieweg, Wiesbaden, Germany, third, revised and extended edition, 2015.

[274] H. Winner, F. Lotz, E. Bauer, U. Konigorski, M. Schreier, J. Adamy, M. Pfromm, R. Bruder, S. Lüke, and S. Cieler. PRORETA 3 – An Integrated ADAS Concept – Comprehensive Driver Assistance by Safety Corridor and Cooperative Automation. In *1. Internationale ATZ-Fachtagung Fahrerassistenzsysteme*, Frankfurt a. M., Germany, Apr. 2015.

[275] H. Winner, F. Lotz, E. Bauer, U. Konigorski, M. Schreier, J. Adamy, M. Pfromm, R. Bruder, S. Lüke, and S. Cieler. PRORETA 3: Comprehensive Driver Assistance by Safety Corridor and Cooperative Automation. In *Handbook of Driver Assistance Systems. Basic Information, Components and Systems for Active Safety and Comfort*, chapter 58, pages 1449–1469. Springer, Switzerland, 2016.

[276] H. Winner and M. Schopper. Adaptive Cruise Control. In *Handbuch Fahrerassistenzsysteme. Grundlagen, Komponenten und Systeme für*

aktive Sicherheit und Komfort, chapter 46, pages 851–891. Springer Vieweg, Wiesbaden, Germany, third, revised and extended edition, 2015.

[277] D. F. Wolf and G. S. Sukhatme. Mobile Robot Simultaneous Localization and Mapping in Dynamic Environments. *Autonomous Robots*, 19(1):53–65, July 2005.

[278] D. Zhang and G. Lu. Segmentation of Moving Objects in Image Sequence: A Review. *Circuits, Systems and Signal Process*, 20(2):143–183, Mar. 2001.

Curriculum Vitae

Personal Data	
Name	Matthias Schreier
Date of Birth	30. December 1985
Place of Birth	Schweinfurt, Germany
Nationality	German

Education	
1996 – 2005	Alexander-von-Humboldt Gymnasium Schweinfurt Degree: Abitur (1,3)
2005 – 2008	Berufsakademie Mosbach Studies: Mechatronics Degree: Dipl.-Ing. (BA) (1,2)
2007	Semester abroad in Tours, France (SKF France – Department Sensor Integration)
2008 – 2010	Leibniz Universität Hannover Studies: Mechatronics Specialization 1: System Dynamics and Control Specialization 2: Automation and Robotics Degree: M.Sc. with distinction (1,0)

Professional Experience	
2011 – 2015	Wissenschaftlicher Mitarbeiter Technische Universität Darmstadt Institute of Automatic Control and Mechatronics Control Methods and Robotics Lab Driver assistance system development PRORETA 3 in cooperation with the Continental AG Degree: Dr.-Ing. (summa cum laude)
2013 – 2014	Project organization PRORETA 3 with final event