Technische Universität München

TUM School of Engineering and Design

# A Generic Framework for Motion Prediction in Autonomous Driving

Phillip Jonathan Karle, M.Sc.

Vollständiger Abdruck der von der TUM School of Engineering and Design der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

genehmigten Dissertation.

| | |
|---|---|
| Vorsitz: | Prof. Dr.-Ing. Boris Lohmann |
| Prüfende der Dissertation: | 1. Prof. Dr.-Ing. Markus Lienkamp |
| | 2. Prof. Dr.-Ing. Steven Peters |

Die Dissertation wurde am 09.04.2024 bei der Technischen Universität München eingereicht und durch die TUM School of Engineering and Design am 11.06.2024 angenommen.

*Meinen Eltern und meiner Schwester*

# Acknowledgments

This dissertation was created during my work as a research associate at the Institute of Automotive Technology at the Technical University of Munich from 2020 to 2024 within two research projects. These are the projects ’Scenario Understanding and Motion Prediction for Autonomous Commercial Vehicle Applications’ in cooperation with MAN Truck & Bus SE and ’Data-Enabled Autonomous Driving’ in cooperation with the Bavarian Research Foundation.

First and foremost, I would like to thank my doctoral supervisor Prof. Dr.-Ing. Markus Lienkamp. Besides the support and supervision of my research, I would especially like to express my gratitude for his trust in his doctoral candidates and the personal opportunities he gave me to realize myself at the institute. I wish to extend my thanks to Prof. Dr.-Ing. Steven Peters for taking on the role of the second examiner and for the constructive and open exchange of ideas.

Besides, my thanks go to my project partners at MAN and the Bavarian Research Foundation for the chance to work on these innovative projects. Also, I want to express my thanks to the Indy Autonomous Challenge for providing and organizing the race cars, tracks, and events in which part of my research took place. In addition, I would like to thank the consortium around the research vehicle *EDGAR* (DFG-number INST 95/1653-1 FUGG) for providing the research platform comprising the vehicle, the data center, and the simulation environment, to which I could contribute with my research. I want to take this opportunity to thank Prof. Dr.-Ing. Matthias Althoff, who was instrumental in shaping this consortium, in addition to Prof. Dr.-Ing. Markus Lienkamp.

I am also thankful to my colleagues at the Institute of Automotive Technology for the constructive exchange about my research and the supportive working atmosphere. Special thanks go to the teams of TUM autonomous motorsport and around EDGAR, which I was privileged to be part of. It was an intense but unforgettable time that we could experience together. It was great to deploy our research in real-world projects and achieve great milestones as a team. In addition, I would like to thank my proofreaders Dr.-Ing. Alexander Wischnewski, Maximilian Geißlinger, Nico Uhlemann, and Daniel Esser for their critical and constructive feedback and discussion about my work.

Finally, I would like to thank my family. You have always provided me with support and encouragement, not only as a PhD student but throughout my entire degree program, also in the phases of my studies that were not the easiest. I am immensely grateful that you have supported me along the way.

Garching, April 2024

Phillip Karle

# Contents

# List of Abbreviations

| | |
|---|---|
| AC@CES | Autonomous Challenge at CES |
| AIRL | Adversarial Inverse Reinforcement Learning |
| AM | Attention Mechanism |
| AMOTA | Average Multi-Object Tracking Accuracy |
| AV | Autonomous Vehicle |
| BC | Behavior Cloning |
| CA | Constant Acceleration |
| CCA | Constant Curvature and Acceleration |
| CCV | Constant Curvature and Velocity |
| CNN | Convolutional Neural Network |
| CTRA | Constant Turn-Rate and Acceleration |
| CTRV | Constant Turn-Rate and Velocity |
| CV | Constant Velocity |
| DBN | Dynamic Bayesian Network |
| DG_LSTM | Proximity-Dependent Graph-LSTM |
| DTW | Dynamic Time Warping |
| EKF | Extended Kalman Filter |
| FN | False-Negative |
| FoV | Field of View |
| FP | False-Positive |
| G_SEL | Selector Model |
| GAIL | Generative Adversarial Imitation Learning |
| GAN | Generative Adversarial Network |
| GCN | Graph Convolution Network |
| GNN | Graph Neural Network |
| GT | Game Theory |
| HiL | Hardware-in-the-Loop |
| HMM | Hidden Markov Model |
| IAC | Indy Autonomous Challenge |
| IDM | Intelligent Driver Model |
| IL | Imitation Learning |
| IMM | Interacting Multiple Model |
| IMS | Indianapolis Motor Speedway |
| IoU | Intersection over Union |
| IRL | Inverse Reinforcement Learning |
| KF | Kalman Filter |
| L | LiDAR |
| L_LSTM | Linear LSTM |

| | |
|---|---|
| LVMS | Las Vegas Motor Speedway |
| mAP | mean Average Precision |
| MSE | Mean Squared Error |
| NLL | Negative Log-Likelihood |
| NN | neural network |
| ODD | Operational Design Domain |
| PF | Particle Filter |
| POMDP | partially observable Markov Decision Process |
| PoV | Point of View |
| R | RADAR |
| RMSE | Root Mean Squared Error |
| RNN | Recurrent Neural Network |
| SOT | Social Tensor |
| SSL | Self-Supervised Learning |
| SVM | Support Vector Machine |
| tGNN | target-based Graph Neural Network |
| TP | True-Positive |
| UID | Unique Identifier |
| UKF | Unscented Kalman Filter |

# Formula Symbols

| Formula Symbols | Unit | Description |
|---|---|---|
| $a$ | $\mathrm{m\,s^{-2}}$ | Acceleration |
| $\alpha$ | - | Learning decay step size |
| $\lvert B\rvert$ | - | Batch size |
| $\mathcal{C}$ | - | Set of base curves |
| $\Gamma$ | - | Aggregation function |
| $c$ | - | Base curve |
| $c_o$ | - | Object class |
| $\gamma$ | - | Learning decay factor |
| $\Delta$ | - | Relative tolerance of selection |
| $\delta$ | $\mathrm{m}$ | Proximity threshold |
| $d_{\mathrm{b,d}}$ | $\mathrm{m}$ | Discretization length of boundary |
| $d_{\mathrm{b,l}}$ | $\mathrm{m}$ | Sampling length along boundary |
| $d_{\mathrm{obj}}$ | $\mathrm{m}$ | Object distance |
| $d_{\mathrm{OBF}}$ | $\mathrm{m}$ | Out-of-bounds filter |
| $d_{\mathrm{map}}$ | $\mathrm{m}$ | Map size |
| $d_{\mathrm{MRG}}$ | $\mathrm{m}$ | Merge distance |
| $d_{\mathrm{MTC}}$ | $\mathrm{m}$ | Match distance |
| $\mathcal{E}$ | - | Set of edges |
| $\epsilon$ | - | Edge |
| $\varepsilon$ | - | Error threshold |
| $\boldsymbol{F}$ | - | State transition matrix |
| $\Phi$ | - | Selection rate |
| $\phi$ | - | Optimization goal |
| $f$ | - | State transition model |
| $f_{\mathrm{EKF}}$ | $\mathrm{Hz}$ | Sampling frequency of EKF |

| | | |
|---|---|---|
| $f_{\mathrm{pred}}$ | Hz | Prediction frequency |
| $f_{\mathrm{node}}$ | Hz | Node frequency |
| $\varphi$ | - | Message function |
| $\mathcal{G}$ | - | Graph |
| $\boldsymbol{H}$ | - | State observation matrix |
| $h$ | - | Observation model |
| $\Theta$ | - | Trajectory cost |
| $\boldsymbol{I}$ | - | Identity matrix |
| $\boldsymbol{K}$ | - | Kalman gain |
| $k$ | - | Number of predicted trajectories |
| $\kappa$ | $\mathrm{m}^{-1}$ | Curvature |
| $\mathcal{L}$ | - | Object list of detection input |
| $L$ | - | Loss |
| $\lambda$ | - | Superposition weight |
| $\mathcal{N}$ | - | Set of nodes |
| $\nu$ | - | Node |
| $\eta$ | - | Learning rate |
| $n_{\mathrm{b}}$ | - | Boundary sampling length |
| $n_{\mathrm{hist}}$ | - | History length |
| $n_{\mathrm{in}}$ | - | Number of objects in detection input |
| $n_{\mathrm{obj}}$ | - | Number of objects |
| $n_{\mathrm{pred}}$ | - | Prediction length |
| $n_{\mathrm{tp}}$ | - | Number of predictors |
| $n_{\mathrm{track}}$ | - | Number of tracked objects |
| $n_{\mathrm{veh}}$ | - | Number of vehicles |
| $\omega$ | - | Weight decay factor |
| $\boldsymbol{P}$ | - | State covariance matrix |
| $\psi$ | rad | Object orientation, yaw angle |
| $\psi_{\mathrm{res}}$ | rad | Residual of orientation |
| $\dot{\psi}$ | $\mathrm{rad\,s^{-1}}$ | Yaw rate |
| $\boldsymbol{Q}$ | - | Covariance matrix of observation noise |
| $\boldsymbol{R}$ | - | Covariance matrix of process noise |
| $\rho$ | - | Base curve |

| | | |
|---|---|---|
| $\rho_{\mathrm{sup}}$ | - | Superposition curve |
| $\boldsymbol{S}$ | - | Covariance matrix of residual noise |
| $s$ | $\mathrm{m}$ | Arc length |
| $\sigma$ | $\mathrm{m}$ | Standard deviation |
| $\mathrm{T}$ | - | Target object node |
| $\mathcal{T}$ | $\mathrm{s}$ | Time stamp of detection input |
| $t$ | $\mathrm{s}$ | Time |
| $\Delta t$ | $\mathrm{s}$ | Time step |
| $t_{\mathrm{delay}}$ | $\mathrm{s}$ | Time delay |
| $t_{\mathrm{MTC}}$ | - | Match counter threshold |
| $t_{\mathrm{obs}}$ | $\mathrm{s}$ | Observation time |
| $t_{\mathrm{pred}}$ | $\mathrm{s}$ | Prediction time |
| $\boldsymbol{u}$ | - | Control vector |
| $\boldsymbol{v}$ | - | Observation noise |
| $v$ | $\mathrm{m\,s^{-1}}$ | Velocity |
| $v_{\mathrm{obj}}$ | $\mathrm{m\,s^{-1}}$ | Object velocity |
| $v_{\mathrm{rel}}$ | $\mathrm{m\,s^{-1}}$ | Relative velocity |
| $v_{\mathrm{res}}$ | $\mathrm{m\,s^{-1}}$ | Residual of velocity |
| $\boldsymbol{w}$ | - | Process noise |
| $w_{\mathrm{w}}$ | - | Weight of MSE-loss |
| $x$ | $\mathrm{m}$ | Longitudinal position |
| $\boldsymbol{x}$ | - | Object state vector |
| $\hat{\boldsymbol{x}}$ | - | State estimation vector |
| $\boldsymbol{x}_{\mathrm{GT}}$ | - | Ground truth trajectory |
| $x_{\mathrm{loc}}$ | $\mathrm{m}$ | Local longitudinal position |
| $\boldsymbol{x}_{\mathrm{pred}}$ | - | Predicted trajectory |
| $x_{\mathrm{res}}$ | $\mathrm{m}$ | Residual of longitudinal position |
| $\tilde{\boldsymbol{y}}$ | - | Measurement residual |
| $y$ | $\mathrm{m}$ | Lateral position |
| $y_{\mathrm{res}}$ | $\mathrm{m}$ | Residual of lateral position |
| $z$ | - | Node feature vector |
| $\boldsymbol{z}$ | - | Observation vector |

# 1 Introduction

## 1.1 Motivation

Autonomous Vehicles (AVs) have the potential to shape a more comfortable [1], safer [2], more inclusive [3], and more efficient [4] type of transportation. The installation of AVs corresponds to the goal of releasing humans from their driving task. A principal feature that characterizes the driving behavior of proficient human drivers is anticipatory driving based on driving experience [5]. The fact that the lack of experience in interactive traffic scenarios is the major reason for fatal accidents emphasizes this statement [6]. Moreover, the risk of accidents is increased by inexperienced drivers, which are more likely to commit errors [7, 8] and by the missing ability to detect dangerous situations [9, 10]. Consequently, if the human driver is assumed as the model for the AV, the question arises of how anticipatory driving through driving experience can be algorithmically realized. A comparison of the current performance of an AV and a human driver reveals that even though AVs are already better at some aspects like quick response times, they perform worse in tasks of complex, difficult-to-predict scenes [11]. Hence, the task of motion prediction based on scenario understanding, which is the algorithmic equivalent of anticipatory driving based on driving experience, has not been solved yet.

The challenges in developing such algorithms are manifold. The first aspect is the complexity of behavior modeling in public road traffic [12, 13]. Even though there are distinct traffic rules, marked lanes, signs, and traffic lights, modeling driving behavior is difficult because the underlying intentions are unknown. In addition, individual driving styles must be considered, especially in mixed traffic with human drivers. Even with the full automation of vehicles, the behavior of cyclists and pedestrians remains difficult to predict. Another aspect is the interaction among traffic participants in dense traffic situations, which influences their behavior [14]. Maneuvers like overtaking, merging, and yielding are influenced by interaction. Thus, the prediction problem can not only focus on a single traffic participant. Instead, the whole scene has to be analyzed [15]. A third issue is noisy and incomplete sensor data and occlusions [16]. Essential aspects such as turn indicators or pedestrians' direction of view can not be reliably detected, and occlusions exacerbate the prediction task because they limit the available information. In addition, weather conditions negatively influence the sensor performance. A further major challenge is the reliability of the algorithms. The AV must ensure a high level of safety. As a rough reference, an EU regulation for the type-approval of AVs exemplarily determines an aggregated acceptance criteria of $10^{-7}$ fatalities per hour of operation [17]. To achieve this, unexpected, rare events such as road accidents, sudden vehicle breakdowns, and medical emergencies must be solved by the AV [18]. Handling these edge cases is stated as the main issue preventing large-scale autonomy [19].

In summary, the high relevance of motion prediction based on scenario understanding on the one hand and the manifold challenges to develop these two features, on the other hand, emphasize a strong need for research in this field to enroll large-scale autonomy on public streets.

## 1.2 Research Objectives

Motion prediction is extensively researched, with several competitions on large datasets, and the methods that have been developed show promising results. The proposed methods feature complex behavior models and interaction awareness and achieve high prediction accuracy on specific metrics. However, the reliability of these algorithms in terms of robustness under real-world conditions such as varying input data quality and the compliance with the constraints of a specific Operational Design Domain (ODD) have been sparsely covered yet. Both are crucial aspects for reliable real-world application. Thus, despite accurate prediction forecasts on fixed datasets, the reliability needs to be improved. Besides that, the adaptivity of the methods to unknown scenarios to maintain prediction accuracy is rarely investigated. Most state-of-the-art methods are developed on one target dataset they can accurately predict, but their behavior under varying scenes and their adjustment to these circumstances are not considered. The development of different prediction methods for specific use cases and object types additionally indicates no consensus about a universal method [20]. It even leads to implementing multiple independent prediction models in one software stack [21]. Thus, another crucial point is developing an adaptive prediction, which features accurate usage on diverse scenes. Beyond the isolated investigation of these two criteria, combining both to create a generic prediction framework is another point of interest (Figure 1.1).

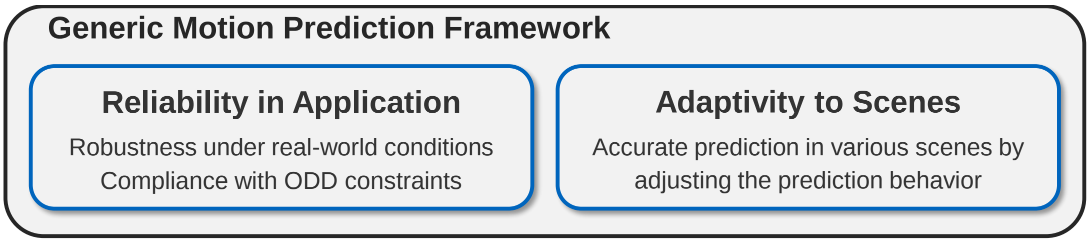

Generic Motion Prediction Framework
Reliability in Application
Robustness under real-world conditions
Compliance with ODD constraints
Adaptivity to Scenes
Accurate prediction in various scenes by adjusting the prediction behavior

Figure 1.1: The research objective of this thesis: The investigation of reliable and adaptive methods for motion prediction and their combination in a generic prediction framework.

This thesis investigates these aspects of achieving reliable and adaptive prediction behavior and discusses their combination in a generic framework. The term reliability comprises the deployment in a full AV software stack in real-world applications. It includes handling sensor noise and perception latency and guaranteeing to fulfill specific constraints in the target ODD. Adaptivity is given if a prediction method maintains accuracy in diverging scenes by adjusting its behavior to current circumstances. The term prediction accuracy is defined by a specific metric and related threshold value for the intended application, which the method has to ensure. In addition, an important aspect of adaptivity is recognizing algorithmic limitations to avoid erroneous predictions. A generic framework for motion prediction must incorporate both features. Reliability in a method's application under specific ODD conditions shall be given, but also the adaptivity to various scenes and the handling of out-of-distribution data is desired to expand the ODD.

## 1.3 Structure of the Thesis

The development of reliable and adaptive prediction methods contributing to a generic prediction framework for autonomous driving is approached as outlined in Figure 1.2. First, the related work is introduced, which comprises sensor late fusion, object tracking, and motion prediction (Chapter 2). The first two research areas are considered because they are essential upstream tasks to realize motion prediction. Based on these surveys, a review is drawn, which results in the identification of the research gap and the formulation of the research question. Five specific research items are derived, subject to investigation by the succeeding two major research streams. The first is reliable motion prediction, investigated in the ODD of autonomous racing (Chapter 3). Two methods are introduced, contributing to the reliable application of motion prediction algorithms. The first is a multi-modal sensor fusion and object tracking method, which increases the accuracy and plausibility of the detection input and compensates for its delay. The second is a physics-constrained deep neural motion prediction method that enables reliable prediction behavior of neural networks (NNs). The second research stream focuses on an adaptive prediction method for public road traffic (Chapter 4). The proposed method comprises a hybrid trajectory prediction, which integrally combines multiple prediction algorithms to enable a diverse behavior. The second part of the method is a self-evaluation algorithm. This evaluation adjusts the prediction behavior to the current circumstances and selects the most suitable method from the hybrid prediction model. The presented methods and results of the two research streams are discussed in Chapter 5 by reviewing the thesis' research items. In addition, the discussion focuses on merging the two research streams into one generic framework for motion prediction. Afterward, the thesis' research question is reviewed and an outlook on future work is given. The thesis terminates with a conclusion of the presented research (Chapter 6).

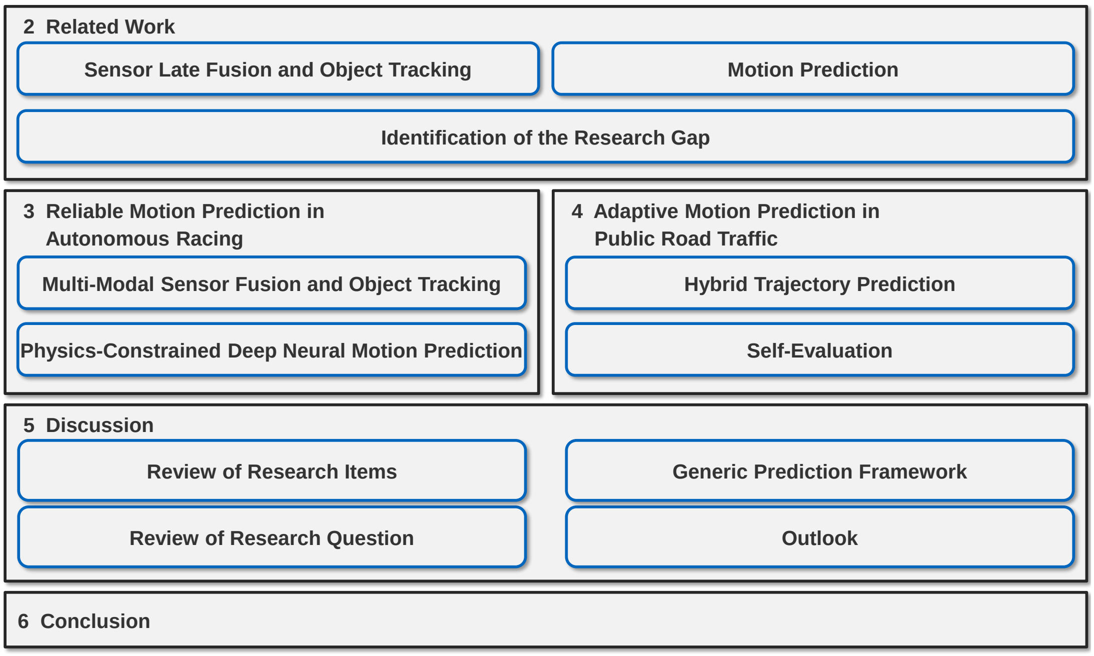

2 Related Work
Sensor Late Fusion and Object Tracking
Motion Prediction
Identification of the Research Gap
3 Reliable Motion Prediction in Autonomous Racing
Multi-Modal Sensor Fusion and Object Tracking
Physics-Constrained Deep Neural Motion Prediction
4 Adaptive Motion Prediction in Public Road Traffic
Hybrid Trajectory Prediction
Self-Evaluation
5 Discussion
Review of Research Items
Generic Prediction Framework
Review of Research Question
Outlook
6 Conclusion

Figure 1.2: Structure of the thesis.

# 2 Related Work

The related work of this thesis comprises sensor late fusion, object tracking, and motion prediction. In the current state of the art, these fields are researched separately. In contrast, this thesis investigates their combination intending to enable a generic motion prediction based on environmental perception.

The foundation for the review of the state of the art and the methods of the thesis is established by introducing relevant terms and definitions (Section 2.1). Subsequently, the surveys on sensor late fusion (Subsection 2.2.1) and object tracking methods (Subsection 2.2.2) are presented. While the latter is a core task of associating objects in consecutive frames, the focus is put on late fusion for two reasons. First, multi-modal sensor setups are the default case of current AVs, so various sensor inputs have to be assumed. Second, late fusion, as the most modular fusion method, is compatible with early and mid-level fusion methods and includes single-sensor setups. Because of this, it offers the highest degree of flexibility. The surveys close with a review of the presented late fusion and tracking methods (Subsection 2.2.3).

In Section 2.3, methods for motion prediction are presented, which comprise physics-based (Subsection 2.3.1), pattern-based (Subsection 2.3.2) and planning-based methods (Subsection 2.3.3). Besides the related work in motion prediction, a central aspect for the generalizability of these methods is presented in Subsection 2.3.4: The consideration of scenario understanding in the prediction task to enhance the overall prediction performance in terms of adaptivity and awareness. Afterward, the presented related work of motion prediction and scenario understanding is reviewed (Subsection 2.3.5).

The introduction of the related work yields the identification of the research gap and the formulation of the research question (Section 2.4). In addition, specific research items are defined to fill the research gap.

To correctly associate the review of the state of the art and the methodical procedure, the modular AV software stack shall be introduced at first. As outlined in Figure 2.1, the object tracking and motion prediction are located between the perception and the planning modules. The perception comprises the tasks of localization, mapping, and detection based on a multi-modal sensor input. The perception module provides lists of detected objects, which are input to the tracking module. In addition, the perception outputs an environment map and the ego state, which are used in the tracking and prediction modules. Within the tracking module, the lists of detected objects are fused in the assumed late fusion and associated in consecutive frames. The resulting tracks are specified by Unique Identifiers (UIDs). The output of the tracking is a unified object list comprising the current and past object states of the surrounding objects. The prediction module uses this unified list to estimate the future states of the objects. Depending on the overall software architecture and planning concept, the module's output can be represented on the behavior level or as occupancy grids, maneuvers, or trajectories. The object predictions are used in the planning module to determine the movement of the ego-vehicle, which is then executed by the control module.

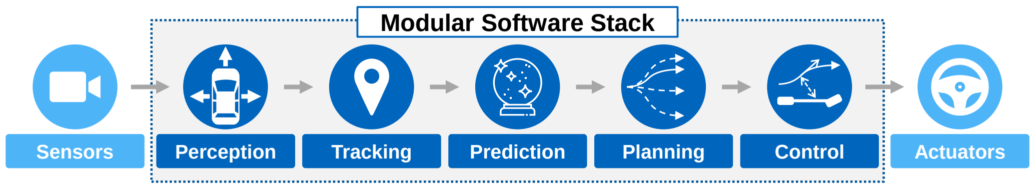


Figure 2.1: Schematic overview of the modular AV software stack [22]. Input to the software is data from sensors, which perceive the local environment and measure internal or external vehicle states. The autonomous software stack completely fulfills the driving task and outputs the desired command signals to the vehicle actuators. Tracking and prediction are between the perception and the planning module.

## 2.1 Terms and Definitions

The relevant terms and definitions in the scope of this thesis are given to establish a common understanding. International standards are used if available. Otherwise, definitions are taken from and aligned with the state of the art in autonomous driving and artificial intelligence.

**Autonomous Vehicle:**
An Autonomous Vehicle (AV) is a vehicle capable of operating without human involvement based on the environmental perception of the sensors it is equipped with. Neither is taking over control of the vehicle by a human nor is the presence of a human inside the vehicle required. According to the standard SAE J3016 [23], which contains taxonomies and definitions for terms related to driving automation systems for on-road motor vehicles, this definition corresponds to Level 4 and Level 5 of driving automation.

**Operational Design Domain:**
The Operational Design Domain (ODD) of an AV defines the limits within which the AV system or specific features are designed to operate [23]. A specific set of conditions describes the ODD. These include environmental (e. g. weather or road type), geographical (e. g. route), time-related (e. g. only weekend), and traffic-related (e. g. no pedestrians) factors.

**Scene:**
A scene is a spatiotemporal setup of objects from a specified Point of View (PoV) [24] and is constructed out of the environment, static, and dynamic objects. In addition, driving instructions can be given [25]. It can be considered as a discrete moment, a *snapshot* of the environment, the surrounding objects, and the observer's self-representation [26].

**Scenario:**
A scenario is defined as a sequence of scenes and describes the temporal evolution of concatenated scenes [26]. Thus, unlike a scene, a scenario lasts over a specific time span. In addition, actions and events may be given to specify the temporal evolution of the scenario.

**Object Tracking**:
Object Tracking is defined by the association and estimation of an object's state in consecutive frames based on an initial specification of the object's state. Thus, it comprises two basic steps: Creating Unique Identifiers (UIDs) for an initial set of object detections and maintaining these UIDs across multiple frames by associating the detected objects [27].

**Motion Prediction**:
Motion prediction refers to the human skill of anticipatory driving. It is defined by forecasting the future states of an object based on tracked object states and semantic information [28, 29].

**Scenario Understanding:**
Scenario understanding is the equivalent of human driving experience. It is described as the algorithmic capability to encode particular scenes or entire traffic scenarios. Based on scenario understanding, assumptions about the intentions and behavior of the surrounding objects are derived, and the criticality of the scenario can be assessed.

**Maneuver:**
A maneuver is a discrete action an object can execute without a detailed movement specification, such as 'turn right' or 'overtake left' [30].

**Trajectory:**
A trajectory is a path that an object in motion follows as a function of time. In the context of AVs, these are the two-dimensional positions over time.

**Reliability:**
Reliability is defined as the robust application of prediction methods in a specific ODD under real-world conditions while ensuring compliance with given ODD constraints. Real-world conditions comprise the aspects of deployment in a full AV software stack, fulfilling computational requirements on the target hardware and handling varying input data quality in terms of noise and latency.

**Adaptivity:**
Adaptivity is defined as adjusting the prediction behavior to maintain prediction accuracy in various scenes and recognizing algorithmic limitations to avoid erroneous predictions. Prediction accuracy refers to fulfilling a threshold value of a specific metric.

## 2.2 Sensor Late Fusion and Object Tracking

Even though there are major advances in sensor technologies for AVs, multi-modal sensor setups are still the dominant concept in the state of the art [31–35]. Fusion concepts are required to derive the maximal information content from these setups. Besides that multi-object tracking is necessary to associate succeeding frames to provide a consistent object motion history and to reduce sensor noise by estimation methods.

### 2.2.1 Sensor Late Fusion

Sensor fusion combines data from multiple sensor modalities to improve specific features of the detection task [36]. There are three common sensor fusion levels: Early fusion at the raw-data level, mid-level fusion at the feature level, and late fusion at the decision level [37]. The fusion of multi-modal sensor data is required because all commonly used sensors show limitations [38, 39]. The comparison of the three fusion approaches [40] reveals that early fusion enables the extraction of multi-modal features, but it is more sensitive to noise and sensor outliers, which reduces the robustness. The limitation of mid-level fusion algorithms is identifying suitable features to be fused, which requires a large amount of data. Besides, low and mid-level fusion methods show a higher sensitivity against sensor calibration errors. In contrast, late fusion approaches show higher robustness but lower performance in terms of mean Average Precision (mAP) because no intermediate sensor features can be combined. In addition, late fusion is less complex but offers worse classification performance. Early fusion is typically realized by camera-LiDAR fusion through deep learning [41–49]. Alternatively, deep learning camera-RADAR fusion approaches exist [50, 51]. In comparison, camera-LiDAR approaches are beneficial in terms of accuracy [52]. In the case of mid-level fusion, merging the data of two modalities on various steps to learn the relevant levels of feature fusion is proposed [53, 54]. Due to the modularity in combining multiple detection pipelines and robustness against calibration errors, sensor noise, and outliers, late fusion methods are the most suitable to fulfill the thesis' focal point on reliability. Thus, the focus is put on late fusion approaches in the following. For further information and reviews of sensor fusion techniques, the reader is referred to [55–57]. An analysis of the application of multi-sensor fusion approaches on embedded platforms is given in [58].

Late fusion is typically realized by recursive Bayesian estimation. The established filter techniques are the linear Kalman Filter (KF) [59], the Extended Kalman Filter (EKF) [60, 61], the Unscented Kalman Filter (UKF) [62] and Particle Filters (PFs) [63]. Instead of Bayesian filters, stochastic sampling approaches for data fusion can be used [64]. Multiple comparisons of different Bayesian filter methods for late fusion exist. In [65, 66], the UKF and the PF outperform the KF for a joint probabilistic camera-LiDAR data fusion with the highest calculation effort attributed to the PF. For combined detection through clustering and tracking of LiDAR and RADAR, the UKF shows superior performance compared to the EKF on a real-world data evaluation regarding tracking error but has a higher computation time, and its parameter tuning is more complex [67].

An important aspect is the adaptivity of the Bayesian filter parameters with respect to the received measurement. As a function of measurement distance, angular position, and object speed, the measurement uncertainties of the detections can be dynamically adjusted [68]. The validation on two state-of-the-art methods [66, 67] shows an improvement of $42\,\%$ on $1/10$-scaled AVs equipped with camera and LiDAR sensors compared to a fixed error model. Other approaches to improve the adaptivity are the hybrid combination of an UKF with a NN or the implementation of a motion model ensemble, which combines multiple models [69]. Regarding the applicability, the methods' calculation efforts must be considered.

The validation of late fusion methods in real-world applications shows the benefits compared to uni-modal methods in terms of tracking accuracy [70]. However, the shown validation of an EKF-based method is limited to low-speed maneuvers on a parking site. In contrast, a real-time detection and tracking system with camera-LiDAR fusion is proposed for urban environments [71]. The system is built from two independent detection pipelines of vision-based deep learning and LiDAR clustering, fused in a KF. A real-world evaluation is conducted in three scenarios: lane change, stop-and-go, and overtaking. However, besides the low-speed range below $10\,\mathrm{m\,s^{-1}}$, no investigation on large object distances and transient behavior is reported, which limits the validity of the approach. The limited validation also applies for camera-LiDAR late fusion proposed in [72]. It is only validated on low-speed stop-and-go scenarios extracted from the large-scale dataset KITTI [73]. The approach applies a Bayesian fusion to optimize the position estimation given by the camera and LiDAR. Again, the results reveal an improvement in terms of position accuracy compared to the non-fused detection pipelines. Late fusion can also be realized by deep learning techniques. A proposed method for 3D data association compares multiple motion models, which are then weighted and merged by means of a NN [69]. Regarding the deployment in an AV stack, the validation on real-word traffic data shows high position errors.

### 2.2.2 Multi-Object Tracking

Methods for multi-object tracking can be distinguished in online and offline tracking. While the former uses only current and previous frames to re-identify the detected object (real-time processing), the latter additionally uses future frames (posterior processing). In the field of AVs, only online tracking is suitable, which is surveyed in the following.

Deep-learning-based online tracking algorithms can be classified into deep network feature-based enhancement, deep network embedding, and deep network learning [74, 75]. While these methods demonstrate high tracking accuracy within specialized datasets, the applicability is limited to the particular ODD of the originating data, a fixed sensor configuration, and the reliance on annotated data [76]. Besides that, there are substantial demands for training and computational resources and the accuracy of these approaches heavily relies on the quality of the dataset, especially the quality of the labeling, which requires time and financial effort. Due to these reasons and the thesis' focus on real-world applications, they are not further considered.

Focussed on Bayesian filter methods, a central aspect of multi-object tracking is the association of new detections to old objects with specified UIDs from previous frames. One approach is to formulate a bipartite matching problem [77]. The bipartite graph comprises the two disjoint sets of newly detected objects and old tracked objects. The goal is to match these two graphs (Figure 2.2). To solve this assignment problem, the Hungarian method [78], also called Kuhn-Munkres-algorithm, can be used. The method is a combinatorial optimization algorithm that aims to find the maximum matching in the bipartite graph. As a metric for the cost function of a match, the Intersection over Union (IoU), the Mahalanobis distance [79, 80], or the Euclidean distance can be used. Alternatively, affinity models are suitable for the data association.

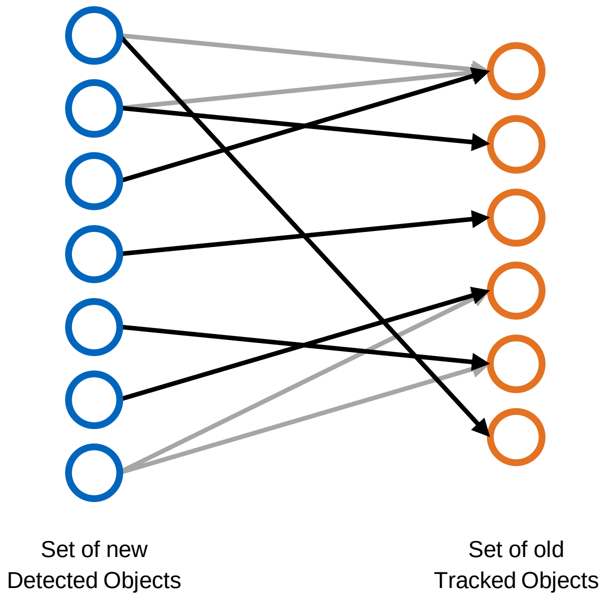


Figure 2.2: Bipartite matching problem between the two sets of new detected and old tracked objects. The goal is to match these two sets by finding the maximum matching (black) from all potential assignments (gray) based on a cost function.

A comparison of different cost functions for the data association using the Hungarian method with a KF for state estimation on the dataset of NuScenes [81] reveals the following: In terms of the Average Multi-Object Tracking Accuracy (AMOTA), the Mahalanobis distance in combination with a NN-based learned distance [82] achieves a score of $0.55$, which is superior to the IoU as cost function [83] with an AMOTA of $0.45$. Regarding calculation time, the IoU is beneficial, the authors report the real-time capability. Besides that, the authors conclude that using the KF is assumed to leave room for improvement by using more advanced state estimators. Compared to the LiDAR-only tracking approach of the two presented methods, a camera-based tracking method achieves an AMOTA of $0.59$ on the NuScenes dataset [84]. A two-stage data association method is proposed. The first step only matches high-confident tracks with new detections. Afterward, the second step associates the low-confident tracks with the remaining detections or high-confident tracks. Otherwise, the respective low-confident track is discarded. The Mahalonobis distance is used to determine the affinity between tracks and detections. The method is additionally validated on the Waymo dataset [85] achieving an AMOTA of $0.37$. The divergent results can be explained by the limited adaptivity of the algorithm and by the different complexities of the datasets.

### 2.2.3 Review of Late Fusion and Tracking Methods

Reviewing the related work of late fusion and multi-object tracking, it can be stated that deep-learning-based methods in both fields contain several limitations. The proposed methods show a high data dependency and training and calculation effort, limiting reliable application. Furthermore, the inherent opacity of deep learning-based algorithms constrains their interpretability. Thus, for these methods, the usage in a full AV stack is of high effort. In addition, the methods lack modularity for varying sensor setups and detection pipelines.

Focussed on the methods that are validated in real-world applications, challenging scenarios are missing to prove algorithmic limitations. The reported fusion methods' applications are at speeds of below $10\,\mathrm{m\,s^{-1}}$ or in steady-state scenarios. Transient behavior is not analyzed. The introduced tracking methods are validated in diverse scenes but only in fixed datasets. Full-stack real-world application is missing, which implies new issues of noisy and delayed detection input data. In addition, the demand for labeled data, which is required to train the learnable features of some methods, restricts the modularity to change sensor setups.

In summary, the state of the art misses fusion and tracking methods that are deployed in full AV software stacks and applied in demanding scenarios, which comprise high motion dynamics and transient situations. Besides that, the development of reliable methods that are able to handle noise, outliers, and accumulated latency in the detection input is required.

## 2.3 Motion Prediction

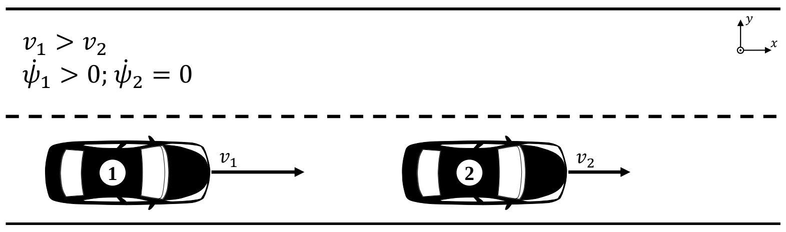
Motion prediction can be made on different degrees of abstraction [86]. Behavior prediction is the most abstract level and aims to estimate an object's intentions without details about the execution. Maneuver prediction tries to forecast discrete actions of the object. Trajectory prediction is the most detailed level. It can be realized unimodally that only one trajectory is predicted, or multi-modally, in the way that multiple trajectories with according probabilities are predicted. The tracked object history of the target object and the surrounding objects, object features, and semantic information such as road geometry and driveable areas are used to determine a motion prediction.

Motion prediction methods can be classified based on the underlying motion model and the approach for risk estimation [87]. Another classification distinguishes key attributes such as model architecture, training procedure and evaluation criteria, and proposed application of the prediction models [88]. Further reviews with a focus on deep learning [28, 89], pedestrian prediction [90, 91], motion prediction in the scope of the safety standards [92] of ISO 26262 [93] and SOTIF [94], and general decision-making tasks [95–97] exist. In the scope of this thesis, prediction methods are classified into physics-based, pattern-based, and planning-based prediction models, proposed in the author's prior work [86]. By this, the approaches are classified based on the respective investigation methodology of the object's motion. Table 2.1 summarizes the three classes and guides through the following survey.

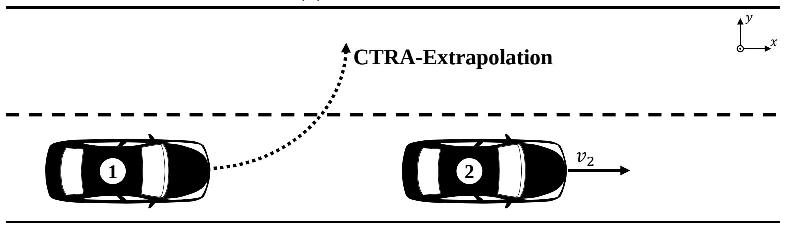
Table 2.1: Classes of motion prediction [86]. The concepts of the prediction classes and their major methods with descriptions and application fields are outlined.

| **Class** | **Concept** | **Methods** |
|---|---|---|
| Physics-based Prediction | Sense – Predict | **State Estimation:**<br>Probabilistic trajectory prediction and uncertainty quantification through kinematic models and Bayesian filters<br>Application: Short-term prediction, collision check<br>**Reachable Set:**<br>Reachability analysis of all possible states within physical constraints to determine an occupancy map as an over-approximation of all reachable states<br>Application: Online verification, safety assessment |
| Pattern-based Prediction | Sense – Learn – Predict | **Clustering:**<br>Clustering of trajectories to determine prototypical trajectories or maneuver clusters ($k$-means, agglomerative clustering)<br>Application: Dimension reduction, determination of prototypical trajectories<br>**Classificiation:**<br>Classification of object motion into predefined classes to determine future maneuvers (Support Vector Machine, Hidden Markov Models, Recurrent Neural Networks)<br>Application: Maneuver prediction<br>**Encoder-Decoder:**<br>Extraction and compressed representation of relevant information from input data (encoding) and derivation of future trajectories (decoding) through deep learning<br>Application: Long-term prediction |
| Planning-based Prediction | Sense – Reason – Predict | **Learning from Demonstration:**<br>Learning of optimal behavior from expert demonstrations by cost function estimation (Inverse Reinforcement Learning) or direct policy determination (Imitational Learning)<br>Application: Long-term prediction<br>**Interaction-aware Planning:**<br>Incorporation of interactions resulting from the behavior of surrounding objects into ego-motion planning by means of partially observable Markov Decision Processes or game theoretic approaches<br>Application: Holistic, interaction-aware planning |

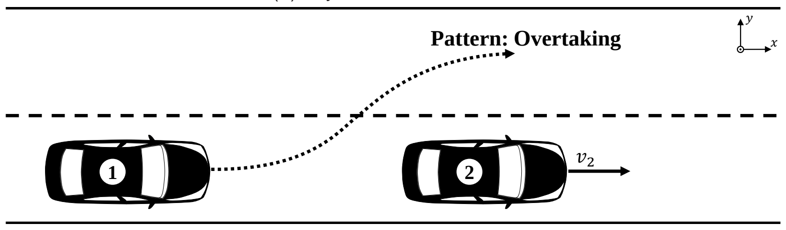
Figure 2.3 outlines the classification exemplary. The scene definition is given in Figure 2.3a. It shows two vehicles driving behind each other on a two-lane highway. The trailing vehicle, vehicle $1$, is faster than the leading ($v_1 > v_2$) and has a yaw rate $\dot{\psi}_1 > 0$, while vehicle $2$ has a yaw rate $\dot{\psi}_2 = 0$. If the three different classes of motion prediction are applied, the following forecasts result. The physics-based model extrapolates the current object state based on a constant velocity and yaw rate assumption (Figure 2.3b). The extrapolation does not hold for long-term prediction as the predicted trajectory indicates that vehicle $1$ will leave the road, which is improbable from an observer's PoV. In comparison, a pattern-based model aims to derive a motion pattern from the given scene consisting of the two vehicles, their states, distance to each other, and the road geometry (Figure 2.3c). In this case, an overtaking pattern is determined as most suitable and applied to forecast vehicle $1$'s motion. Lastly, a planning-based model applies a cost function to the given scene and assumes the future behavior based on the lowest costs (Figure 2.3d). The two exemplary options are switching lanes and acceleration on the fast lane with the costs $\theta_1$ and overtaking and switching back to the right lane with the costs $\theta_2$. Since $\theta_1 < \theta_2$ holds, the prediction of switching lanes and accelerating results for vehicle $1$.



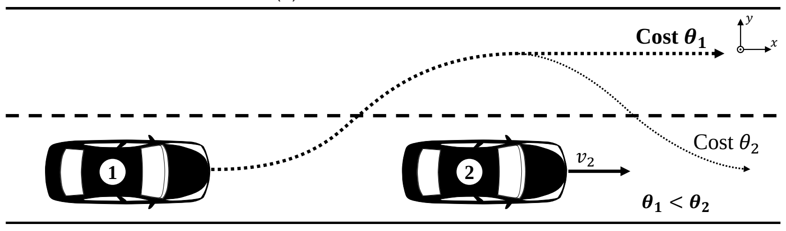
(a) Scenario Definition



(b) Physics-based Prediction



(c) Pattern-based Prediction



(d) Planning-based Prediction

Figure 2.3: Illustration of the three prediction classes [86]. An exemplary highway scene of two vehicles in a row is given. If a physics-based model is applied, the current state of vehicle 1 is extrapolated. A pattern-based model derives the forecast from known motion patterns. The planning-based approach compares the costs of the different options to determine the forecast.

### 2.3.1 Physics-based Prediction

Physics-based prediction methods use equations of motion from classical mechanics to model the future motion of an object. Dynamic or kinematic models can be used to describe the motion. The forecast can be described as trajectory by means of state estimation models or as occupancy grids by means of reachable sets (Table 2.1).

#### State Estimation

State estimation is realized by Bayesian filters, known from object tracking (Subsection 2.2.2). The filter extrapolates the current object state to the future through a motion model. Dynamic models like bicycle models and four-wheel models can be used to model the object's motion. However, kinematic models are preferred because the environment sensors can not measure the required object states to use a dynamic model, and the level of detail of dynamic models is not required. Figure 2.4 outlines common kinematic models and their dependencies.

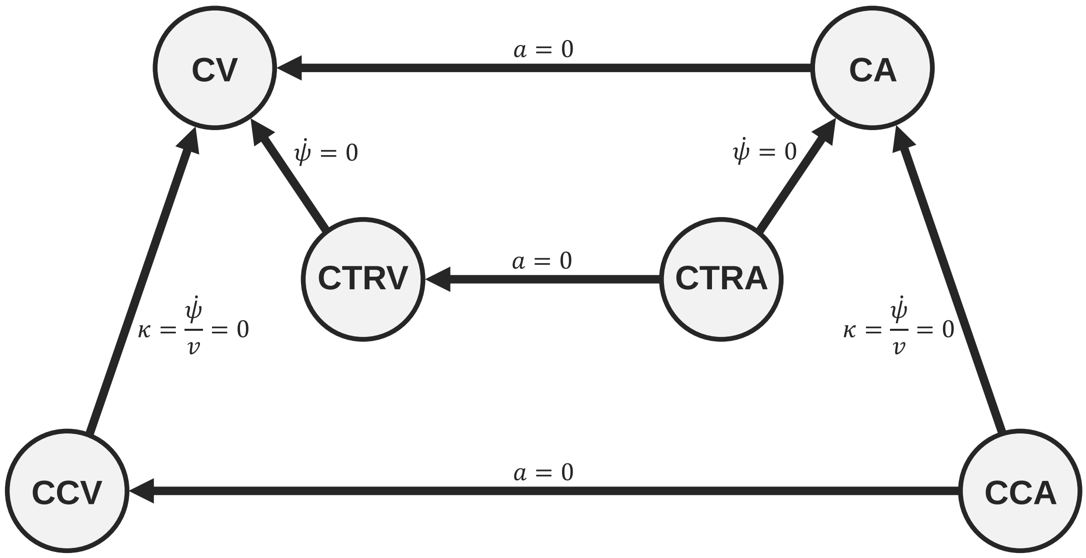


Figure 2.4: Overview of kinematic models [98]. By setting a specific state variable to zero, the more sophisticated models can be transformed into simpler ones.

The models with the lowest complexity assume linear behavior of Constant Velocity (CV) or Constant Acceleration (CA). They are beneficial because optimal propagation of the state probability is possible but are insufficient if rotating motion has to be modeled. The yaw rate can be considered in the state equations to achieve this. The resulting curvilinear models assume a constant yaw rate. Constant Turn-Rate and Velocity (CTRV) or Constant Turn-Rate and Acceleration (CTRA) motion behavior is assumed. These models' limitation is that they do not consider the correlation between velocity and yaw rate, which can change the yaw rate even if the vehicle does not move. To avoid this, the curvature $\kappa$ can be used, which describes the correlation between the yaw rate and the velocity. The motion models of Constant Curvature and Velocity (CCV) and Constant Curvature and Acceleration (CCA) result. The choice of which model to use depends on the motion behavior of the objects, available measurement features, and measurement accuracy. The acceleration and the yaw rate are especially difficult to model because they are prone to oscillations and, with state-of-the-art detection methods, not measurable. A study states that the best compromise between accuracy and computational costs is achieved by the CTRA-model, even though the sensitivity to noise is mentioned as drawback [98, 99].

The motion models can be applied separately in state estimators like KF [100], EKF [101], UKF [102] or PF [103, 104] but can also be combined to improve the adaptivity. The Interacting Multiple Model (IMM) [105] is an established method [106–109] to combine multiple models. It uses a Markov Model to calculate the probability for each kinematic model to occur and weights them accordingly to output the prediction. To enhance the models' plausibility, the state estimation can be extended by the Intelligent Driver Model (IDM) [101].

### Reachable Sets

An alternative approach for physics-based motion prediction is to assess all potential states an object can reach within a specified time horizon based on predefined constraints for dynamic limits. The set of possible states is defined as the reachable set [110–112]. The reachable set can be approximated using convex hulls to account for errors resulting from these assumptions and to simplify its shape for subsequent processing steps [110]. Given that the reachable set expands with the prediction horizon, occupying a substantial amount of space, it can be represented stochastically through a probability distribution [110]. The set-based prediction of traffic participants [113] is a dedicated tool that leverages reachable sets. It predicts future occupancy by considering all physically feasible maneuvers and additionally incorporates traffic rules to constrain the solution space. A drawback of considering traffic rules is the strong dependency on these rules and road users' compliance with them. Alternatively, to reduce the size of this set, constraints like kinematic simplifications can be applied, or non-holonomic constraints can be taken into consideration [114].

## 2.3.2 Pattern-based Prediction

Pattern-based prediction methods match a scene to the most probable previously learned or implemented pattern. The pattern description depends on the ODD and can comprise object states and features or semantic information such as road geometry and driveable areas. The abstraction level can range from single-trajectory to maneuver prediction. This prediction class can be divided into the three sub-categories of clustering, classification, and encoder-decoder models (Table 2.1).

### Clustering

Clustering groups trajectories to create prototypical trajectories, which are compared to the target object's movement to output its future motion. The prototypical trajectories are extracted from tracked data by clustering methods. $k$-means [115] and agglomerative clustering [116] can be used. The similarity between observed and prototypical trajectory is calculated by metrics like the Euclidean distance [117] or the modified Hausdorff distance [118]. Alternatively, non-metric similarity functions like Dynamic Time Warping (DTW) [119] or Longest Common Subsequence [120] can be used. A study comparing different approaches to determine the trajectory similarity reveals that the modified Hausdorff distance is the most suitable metric for intersection scenarios and highway traffic [121]. Another comparison at intersections through the $k$-means algorithm between the DTW function and Euclidean distance metric recommends DTW [122]. Through DTW, a higher distance between the cluster centers and a lower inter-cluster distance are achieved, which both correspond to better data separation.

The inherent drawback of clustering methods is the restriction of the prediction solution space to the number of prototypical trajectories. To counteract this, superposition can be applied. Gaussian Mixture Models, which superpose single trajectories based on the probability of matching the observation, can be used [123, 124]. A survey on methods for trajectory clustering, distance metrics, and data processing is given in [125].

## Classification

If desired maneuvers are a priori defined and the observations are classified into them to derive the prediction, a classification approach is given. For maneuver classification, Support Vector Machines (SVMs) [126], Hidden Markov Models (HMMs), Dynamic Bayesian Networks (DBNs) or NN-based approaches can be used. The SVM can be applied for lane change detection [127–129] and maneuver prediction at intersections [130]. To model sequential behavior, HMMs are beneficial, which belong to the class of discrete Markov models. Thus, they describe a scenario with discrete states and transition probabilities. The Markov property that only the current state is relevant for the next state transition or action applies [131]. The term hidden refers to the fact that the current state, which is the current scene, is not completely known but estimated. HMMs are used to predict lane changes [132] and for highway applications [133]. DBNs are applied to interactive scenarios [134, 135]. These models estimate the object's motion through Bayesian inference by outputting a probability mass function over a given set of maneuvers.

## Encoder-Decoder

Deep neural network encoder-decoders are currently the most performant prediction approaches in the competitions on large-scale prediction datasets [15, 81, 136, 137]. The models consist of an encoding part, which extracts information from the scene and transforms it into a compact representation, the latent space. Based on this latent representation, the decoder determines the motion forecast by transforming the latent information back to a physical representation. The latent scenario representation as a core part of these architectures is analyzed in [138]. The results presented show that clusters can be identified in the latent space for different maneuvers, but further analysis of the relation between given scenes, latent representation, and resulting prediction accuracy is recommended. Despite the common structure of encoder-decoders, they differ in the type of encoder network architectures used and the latent representation. Table 2.2 outlines the different architectures.

Table 2.2: Classification of encoder-decoder architectures for motion prediction. The classification is between the type of encoder network architecture used and the latent space representation [86].

| | **Discrete Encoder-Decoder**<br>*Discrete* | **Variational Autoencoder**<br>*Distributional* |
|---|---|---|
| **RNN**<br>*Temporal* | • Grid Map [139, 140]<br>• GAN [142–144]<br>• Spectral Clustering [145]<br>• Attention Mechanism [146] | • Kinematic Constraint Layer [141] |
| **CNN**<br>*Spatial* | • Rasterized Grids [147–151]<br>• Target-based [152] | |
| **RNN-CNN, ConvLSTM**<br>*Spatiotemporal* | • BEV-rendering [153–157]<br>• Gabor Convolutional Networks [159]<br>• Social Tensor [161–164]<br>• Spatial Horizon Interaction [167]<br>• Attention Mechanism [168–170] | • Conditional GAN [158]<br>• Regression-based Adaptation [160]<br>• Joint Distribution [165, 166]<br>• Diversity Sampling [160] |
| **GNN**<br>*Relational* | • Heat Map [171–173]<br>• Temporal Scene Graph [175–177]<br>• Target-based [181–185]<br>• Graph Convolutional Network [187–190]<br>• Motion Transformer [191–197]<br>• Transformer GAN [198, 199]<br>• Mix and Match [200] | • Multi-Graph Encoding [174]<br>• Attention Mechanism [178–180]<br>• Ego-agent conditioned [186] |

The encoder network architectures can be classified into temporal, spatial, spatiotemporal, and relational. The decoder is constructed out of Recurrent Neural Networks (RNNs) because of the temporal enrollment of the object's future. Convolutional Neural Network (CNN)-based decoders can be used to output occupancy maps. The latent space can be represented discrete or distributional. The input is mapped into the latent vector with specific values in the first case. Due to this fact, the models are prone to overfitting. In addition, the latent space is not constrained. Thus, there is no continuity given between different latent scene clusters. These issues can be compensated by using a distributional representation of the latent space, which encodes the scene into a continuous latent representation. The latent distribution can be regulated during training to enforce concatenating distributions. The Kullback-Leibler divergence [201] is used for the regularization. Even though the higher generalizability argues for variational autoencoder compared to discrete encoder-decoder, they are more difficult to train because of the regularization of the latent space [202] and show inferior results in the state of the art. Due to these facts, they are less used (Table 2.2), and the focus is on discrete encoder-decoders in the following.

Temporal encoding is realized by RNNs. Applications are shown for highway scenarios [139, 140]. The presented approaches rasterize all predicted trajectories into a grid map on which the ego-motion planning is based. To enhance the generalizability, Generative Adversarial Networks (GANs) can be used [142, 143]. Also, noise can be added to the latent representation, which improves the trajectory hypothesis coverage and, by this, the resulting risk assessment [144]. To model interactions, a spectral clustering [145] of the scene or Attention Mechanisms (AMs) [146] can be integrated into the input processing.

The spatial context of a scenario can be modeled by CNNs. Either static information, such as road geometry, can be encoded, or the objects can be placed in spatial relation to each other to model their interactions. Pure CNN-based approaches render the scene information into images and output future occupancy grids by concatenating CNNs in the decoder [147–150]. Additional dimensions can extend the rasterized grids to render further information [151]. To enhance the feasibility of the predicted trajectories, the network can be conditioned to determine the endpoints first and refine the trajectory in a second step [152].

To combine the advantages of temporal and spatial encoding, hybrid network architectures of RNNs and CNNs are proposed. Besides, ConvLSTMs, which are architectures dedicated to processing spatiotemporal data, are presented. The benefits of combining both approaches are shown in [153, 155], which report improvements utilizing spatiotemporal architectures compared to the spatial baseline architecture [147]. Also, the encoding of the object history through a RNN into a Social Tensor (SOT) [161], which is processed by CNN-layers, shows an improvement in the Root Mean Squared Error (RMSE) and Negative Log-Likelihood (NLL) compared to a RNN-based appraoch [140]. Further improvements of the SOT-approach are the additional encoding of the map information [163], non-spatial pooling by means of multi-head attention [162], and the weighting of interactions based on the target object's Field of View (FoV), which is beneficial in dense traffic scenarios [167]. By means of Self-Supervised Learning (SSL), pre-trained models can be adjusted during inference [164] or the training can be regulated for robustness enhancement [203]. Through AMs [204] such as Memory Augmented Neural Networks, the latent space is elementwise addressable. This results in a more efficient latent representation and prediction error reduction compared to the state of the art without memory augmentation [168].

The network architectures that show the best accuracy on the leaderboards of the public prediction datasets are Graph Neural Networks (GNNs) [205, 206], especially featured with AMs [204] and transformers [207]. They show superior performance because the constructed graphs can model non-Euclidean interactions [208], an efficient semantic encoding can be realized [171, 179, 180] or both aspects can be combined [172, 173, 176, 181]. Map encoding through a graph and the output of a heat map of possible future driving lanes of all objects in the scene has shown to perform best on various prediction challenges at the time of its publication [171]. The approach was improved by adding the interactions between agents to the GNN [172]. The efficiency of a vector representation of the scene instead of a rendered image and the usage of GNNs is reported to reduce the computational costs while maintaining the same prediction accuracy [194]. Using target-based Graph Neural Networks (tGNNs), semantic information can be encoded to determine the object's targets as a function of environment and interactions, which is shown to enhance the plausibility of the prediction [182]. A directed spatiotemporal graph in combination with RNNs has shown to be capable of predicting in vehicle-dominant datasets and dedicated pedestrian datasets [186]. With the abilities of self-attention and parallel processing of sequential input data, which allows the model to focus on specific parts and to improve efficiency, transformers show high accuracy in the task of motion prediction [209]. The attention is achieved by assigning weights to different parts of the input. The transformer's self-attention can be used to model the relationship between road components and objects [194]. Additionally, it is possible to predict the probability distribution of trajectory endpoints [183] through self-attention. Another form of GNNs for scene encoding and motion prediction are Graph Convolution Networks (GCNs). Since this type of GNN is used in this work (Chapter 4), its concept is introduced (Figure 2.5).

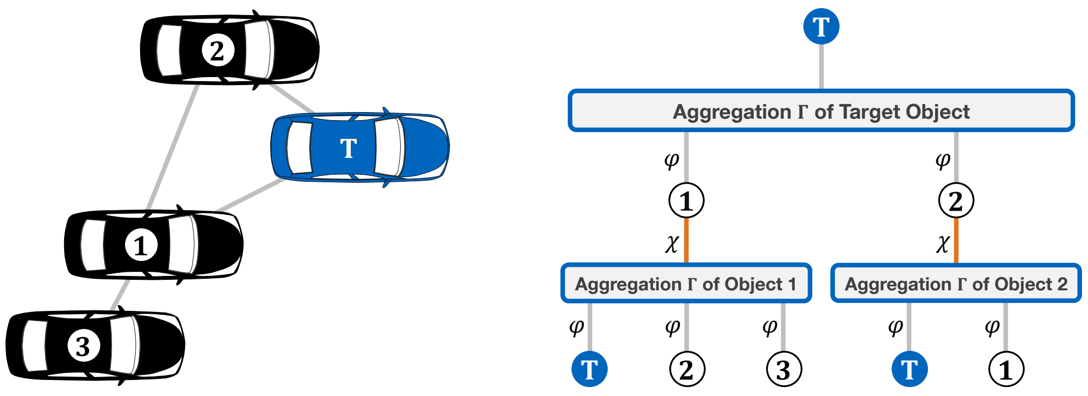

2
T
1
3
T
Aggregation Γ of Target Object
φ
φ
1
2
χ
χ
Aggregation Γ of Object 1
Aggregation Γ of Object 2
φ
φ
φ
φ
φ
T
2
3
T
1

Figure 2.5: Graph representation between multiple vehicles (left) and the update process of target object's node $\mathrm{T}$ by a GCN of second order (right). Each update step comprises the message passing through the message function $\varphi$, the aggregation $\Gamma$ of all messages per node, and the update of the node by the function $\chi$.

The objects in the scene are represented as nodes in a graph with connections, called edges, between them. Based on this representation, the idea of GCNs is creating and passing messages along the edges between the nodes [210]. The messages are passed between the nodes via the message function $\varphi$. The convolutional operation takes place in the aggregation function $\Gamma$ to filter the information received from the surrounding objects. This filtered information is then processed by the update function $\chi$ to receive the updated state of the respective node. The message and update functions are represented by NNs. The convolutional operation is useful for road traffic prediction because it is more efficient to aggregate the local graph structure than sequential GNNs [211]. Thus, the local interactions are combined with the relational interactions of the graph representation. In addition, GCNs are stated to generalize to unseen data because of the shared message and update functions [212]. Through replacing the aforementioned SOT by GCNs to model the interaction between the road users, an improvement in the prediction accuracy on the NGSIM dataset [208] compared to existing models is shown [187]. Additionally, a reduction of the calculation time through GCNs compared to CNNs is reported because the interactions are individually established [188]. To optimize the graph encoding, weighted adjacency matrices [189] or heterogenous convolutional operators can be used [190]. The latter outperformed the state of the art at the time of its publication on the Argoverse2 dataset [137].

### 2.3.3 Planning-based Prediction

Planning-based prediction methods assume that every object acts as an agent with a specific intention. These approaches are motivated by the PoV that all objects follow similar goals, and it is just the perspective that differs between ego-motion planning and the prediction of the surrounding objects (Table 2.1). So prediction and planning are interpreted as a change of perspective [213]. The assumed intention is used to predict the future movement of the agent. Reasoning about an agent's intention and explicitly deriving the related behavior require demonstrations to learn the behavior. Alternatively, the agent's intention can be implicitly considered in ego-motion planning, described as interaction-aware planning.

## Learning from Demonstration

Learning from demonstration can be realized by Inverse Reinforcement Learning (IRL) and Imitation Learning (IL). Both approaches follow the idea that expert demonstrations represent the optimal behavior. In the case of IRL, a reward function is learned from these demonstrations, which is used during inference to derive the object's behavior. In contrast, IL directly learns policies from the expert demonstrations and applies them to the target agent.

Due to the possible ambiguity of the optimal policy of IRL because of non-convex cost functions [214], Maximum Entropy IRL [215] can be used [216–219], which resolves the ambiguity by maximizing the entropy over the state-action distribution, i. e. creates a convex cost function. A drawback of IRL is the lack of applicability in sparse environments and with diverse agent goals. In these cases, IL is beneficial, which does not require any cost function but directly derives the policy. The simplest form of IL is Behavior Cloning (BC) [220], which was already used in one of the first AVs, ALVINN [221]. Due to the fact that BC solely imitates the expert demonstration, it features low adaptivity. To achieve a more general imitation, Generative Adversarial Imitation Learning (GAIL) can be used, which learns a policy against a discriminator that aims to distinguish between the expert trajectory and the learned policy. In a comparative application to highway scenarios to predict human driving behavior, GAIL outperforms BC in terms of accuracy [222]. Compared to IRL, GAIL is stated to be more efficient and better scalable to more complex tasks [223]. Another approach to avoid pure behavior imitation is the consideration of penalties during the training process, which additionally improves the adaptivity when the algorithm is applied to new scenarios [224]. Adversarial methods can also be applied to IRL, which results in Adversarial Inverse Reinforcement Learning (AIRL). AIRL determines a cost function to predict a trajectory that a discriminator assesses to improve the robustness [225].

## Interaction-aware Planning

Algorithms that fall under the category of interaction-aware planning iteratively incorporate the prediction of the surrounding objects in the ego-motion planning. By this, interactions between the ego action and the reaction by the surrounding objects are modeled. The two methods used for interaction-aware planning are the online partially observable Markov Decision Process (POMDP) [226, 227] and Game Theory (GT). Online POMDPs are applied for lane change maneuvers [228] and in intersection scenarios [229, 230]. In autonomous racing, an online POMDP, solved by reinforcement learning, is shown on $1/10$-scaled racing cars [231]. The algorithm adaptively adjusts risk in relation to the uncertainty given by the surrounding opponents. In the field of GT, stochastic games are used to describe the scenarios [232]. A Stackelberg equilibrium can be used to give an advantage to the ego vehicle if set as the leader. Cooperative formulations are proposed, too [233]. If applied in AVs, the same game-theoretic model is applied to every object in the traffic situation and is then iteratively solved. However, the iterative character limits the scalability, which can be compensated by the level-$k$ GT [234–236]. This theory individually assigns reasoning levels to the ego and the surrounding objects, which means that the iteration is stopped at level $k$ for the respective object [234]. Due to the common goal to win the race, the application of non-cooperative GT to autonomous racing is common [237–241]. A survey of these techniques and general planning methods is given in [242].

### 2.3.4 Consideration of Scenario Understanding

The state of the art is focused on pure motion prediction algorithms. The aspect of scenario understanding is neglected, even though an in-depth understanding of a scenario is essential for profound behavioral estimates and predictions [243]. There are two focal points: The incorporation of scenario understanding to derive more general prediction assumptions during the training process (offline) or the evaluation and adaption of the prediction method by means of scenario understanding during inference to maintain prediction accuracy in varying scenes (online).

A method to improve the prediction accuracy offline is to enforce an amplified consideration of semantic information instead of focussing on the motion history as an input feature in the training process [244]. The reported results show improved accuracy and prediction stability on the NuScenes dataset compared to the validation without scenario understanding [81]. Heuristic scenario understanding in the form of preliminary plausible information about the future trajectory can alternatively be added to the model to improve its accuracy [245]. The proposed model is created out of GNNs with AMs and conducts the plausibility check based on the object type and the road geometry, which are encoded in a latent distribution and used to determine possible goal points based on the similarity to latent clusters.

Using scenario understanding for online evaluation and adaption can be realized by self-evaluation. It is defined as the assessment of the algorithmic capabilities in relation to the current circumstances in a scene [246]. Thus, a combination of surrounding effects and internal algorithmic features is assessed. An approach for self-evaluation is estimating the model's accuracy and confidence through Bayesian belief networks [247]. The validation shows that ego-motion planning gets safer and more efficient with additional confidence-awareness. Limitations of this approach are that it is only validated in the specific field of human motion prediction in quadrocopter applications and shows high computational complexity. Another self-evaluation approach proposes incorporating constraints into the prediction algorithm to detect its limitations [246]. Three constraints are proposed: the time required for emergency braking, maneuver constraints such as the time needed to complete a driving maneuver, and model constraints in dependency on the prediction horizon. Instead of a temporal evaluation, a causal self-evaluation can be created, which establishes interpretability and checks compliance with the traffic rules [248]. The interpretability is created by clustering the predicted trajectories in high-level maneuvers, which are then compared to the traffic rules. It offers the advantage that the accuracy of the trajectory predictions is maintained while the additionally created clusters are used to evaluate the feasibility compared to the traffic situation. Alternatively, the uncertainty of the model's prediction can be quantified through evidential deep learning [249], which learns a Dirichlet distribution on the classified maneuvers. The quantified confidence can be used to detect failures, which are assumed to result from high prediction uncertainties [250]. This hypothesis is validated with different prediction algorithms, uncertainty estimation methods, and uncertainty scores. Besides, the detection of algorithmic limitations can be realized by anomaly detection [251]. An anomaly is detected by sampling predictions of all objects, including the ego object, and assigning costs to these samples. Then, the costs of the ego-motion planner, which is based on an in-parallel determined single-trajectory prediction, are compared to the cost distribution of the samples. An anomaly is given if the costs of the actual prediction the planner uses are above the $p$-quantile of the distribution. The validation of this approach shows an improvement compared to the baseline methods without anomaly detection. Lastly, a performance monitor, which directly outputs an anomaly score to assess the prediction for black-box trajectory prediction models, is proposed as a self-evaluation method [20]. The score is determined by comparing the feature space of a scenario, which is its encoded object motion and semantic information, to the feature space distribution from the training data. The closer the encoded feature space is to the training data distribution, the lower the score. The predicted trajectory is determined in parallel to the anomaly score by an ensemble of multiple models.

### 2.3.5 Review of Prediction Methods

The related work in motion prediction reveals a trade-off between deterministic models with physics-based approaches and complex model approaches, mainly based on the deep-learning architectures [86]. The former can be used in safety applications such as online verification [252]. In addition, they can be applied in steady-state scenarios and show high accuracy in pedestrian prediction [253]. However, the underlying assumptions lead to conservative behavior or rely on distinct traffic rules in case of reachable sets. In the case of state-estimation methods, the ability to depict transient motion patterns is not given. The more advanced modeling approaches are able to describe these motion patterns if the amount and quality of training data are given. While these models achieve high accuracy for specific prediction tasks, their prediction accuracy deteriorates if applied on out-of-distribution data [254]. Additionally, the state of the art does not offer full stack application of the top-ranked models of the public prediction competitions. The computational constraints and the required data mining for the real-world application in a specific ODD are crucial bottlenecks to realize this.

With the focus on resolving the described trade-off between the models' features, hybrid methods, which combine approaches from different classes, are proposed. Combinations of multiple physics-based models [102, 255] or combinations of a physics-based model and a deep-learning approach [256–259] are proposed. A common approach is a weighted combination through Gaussian Mixture Models [256] or likelihood optimization based on the models' uncertainties [257] (Figure 2.6). Besides that, combinations of planning-based and pattern-based approaches exist [260, 261]. While the mix of physics-based models is still limited to their underlying laws of physics to describe motion, the mix of physics-based models and deep learning aims to merge the underlying principles for more comprehensive motion models. Compared to isolated pattern-based models, these blends of principles show similar accuracy but enhanced interpretability and robustness against unknown data by incorporating the physics-based aspect [256]. Alternatively, ensemble methods [20, 69], which use multiple models in parallel, can be utilized to consider various prediction approaches. The drawback of ensemble methods is that multiple models are executed in parallel, which makes them computationally inefficient.

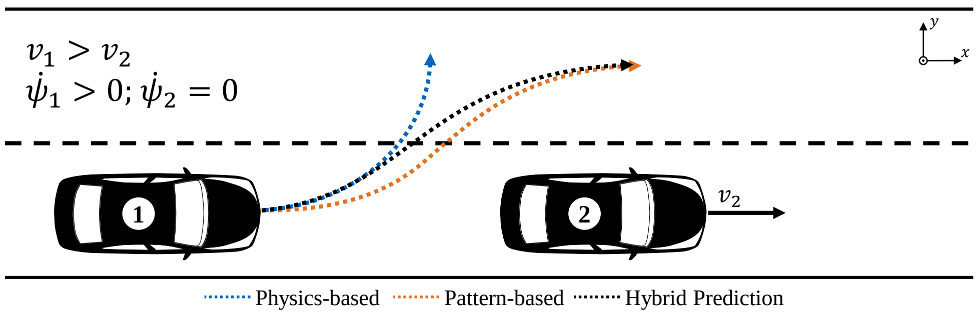


Figure 2.6: The hybrid prediction is composed of a physics-based and a pattern-based model. Heuristics, optimization methods, or probabilistic models are used to determine the weighted combination, which is either constant or dynamic along the horizon.

Another limitation in the state of the art is that the incorporation of scenario understanding to enable a more profound scene encoding is sparsely shown. The presented work of scenario understanding is based on cautious heuristics that lead to conservative behavior or focuses on the prediction adjustment without assessing erroneous predictions. Further methods require the ground truth for failure evaluation.

In summary, a prediction method that combines the strengths of multiple models for accurate prediction in diverse scenarios misses in the state of the art. Beyond the scenario coverage, it is desirable to detect the algorithmic limitations, which is crucial for safe application. Scenario understanding for self-evaluation offers features to achieve this, but it is rarely realized in the state of the art.

## 2.4 Identification of the Research Gap

Besides the isolated reviews of fusion and tracking (Subsection 2.2.3) and prediction methods (Subsection 2.3.5), it shall be discussed which specific criteria are required to realize a generic prediction framework for motion prediction in autonomous driving and to what extend the state of the art fulfills them. The discussion leads to identifying the research gap and formulating the research question, which builds the foundation of this thesis. Lastly, five research items are defined, which guide through the succeeding methods.

The two proposed criteria required for the generic framework are reliability and adaptivity, both defined in Section 2.1. The aspect of reliability is chosen because of the gap between the prediction method's accuracy on fixed data and the performance when deployed in a full software stack [254]. One reason for this gap is that the vehicles' interaction is missing on fixed datasets compared to full-stack real-world applications. In addition, a trade-off between the computational efficiency of the prediction models and their accuracy is identified, which motivates the investigation of reliable applications. This argument is confirmed by experiments, which show that increasing hardware capacities correlates with the algorithmic performance of AV software [262], while the algorithms have to run in real-time on limited computational recourses on the AV's hardware. Another aspect is discussed in a survey of tracking and prediction methods [263]. The survey concludes that the determination of a consistent object history and the accurate estimate of the object's current state are identified as essential to base the prediction upon. It is argued that detecting abnormal behavior and predicting surrounding objects, irrespective of traffic rules, require precise tracking of the objects' motions. Since this level of input data quality can not be ensured in real-world applications to varying environmental conditions and traffic situations, the aspect of reliability to handle varying data quality is a point of interest. Coping with the perception latency as the last criterion of reliability is motivated by two studies. In the specific field of autonomous racing [264] and the related autonomous application of autonomous quadrocopoters [265], it is proven that the overarching objective of enabling reliable application is linked to the permissible latency of the system. Distinct correlations are identified between maximum latency and vehicle performance [264] and safety guarantees [265].

The definition of the requirement for adaptivity is motivated by the decrease in the accuracy of public prediction competitions' top-ranked models when evaluated on unknown scenes. Two of these methods [168, 186] have been analyzed on three different datasets [81, 266, 267] regarding their adaptivity. The datasets are split into separate driving domains, which differ in road type, origin country, and object speed ranges. The cross-evaluation of the methods on domains distinct from the training domain reveals an increase of the average displacement error of up to $78\,\%$ [268]. Avoiding erroneous predictions by recognizing algorithmic limitations is another aspect of adaptivity as defined in Section 2.1. To quantify this aspect, the rate of mispredictions, which can be interpreted as outliers, can be used. The NuScenes prediction competition [81] defines the MissRate2k $\mathrm{MR}_{2_k}$ as:

$$\mathrm{MR}_{2_k} = \frac{1}{n_{\mathrm{obj}}} \sum_{l=1}^{n_{\mathrm{obj}}} \mathbb{1}\left( \bigvee_{i=1}^{k} \left( \max_{\boldsymbol{x}_m \in \boldsymbol{x}_i} \|\boldsymbol{x}_{l,m,\mathrm{pred}} - \boldsymbol{x}_{l,m,\mathrm{GT}}\|_2 > 2.0 \right) \right) \tag{2.1}$$

The equation states that the MissRate2k $\mathrm{MR}_{2_k}$ is the share of mispredictions among all predicted objects $n_{\mathrm{obj}}$. A misprediction is given if the deviation along the trajectory between the prediction $\boldsymbol{x}_{\mathrm{pred}}$ and the ground truth $\boldsymbol{x}_{\mathrm{GT}}$ exceeds $2\,\mathrm{m}$ at any point. For each object, the $k$ most likely predictions are taken and evaluated for mispredictions. Since the miss rate quantifies an upper error threshold instead of determining an exact Euclidean error, it can be used to quantify the rate of erroneous predictions. When analyzing the top-ranked models in the prediction competition, high miss rates ($> 10\,\%$) on the test data can be observed. Since the algorithms are optimized for low mean Euclidean errors, avoiding mispredictions is not ensured. If the related work is classified by these two criteria, the following qualitative evaluation results (Figure 2.7).

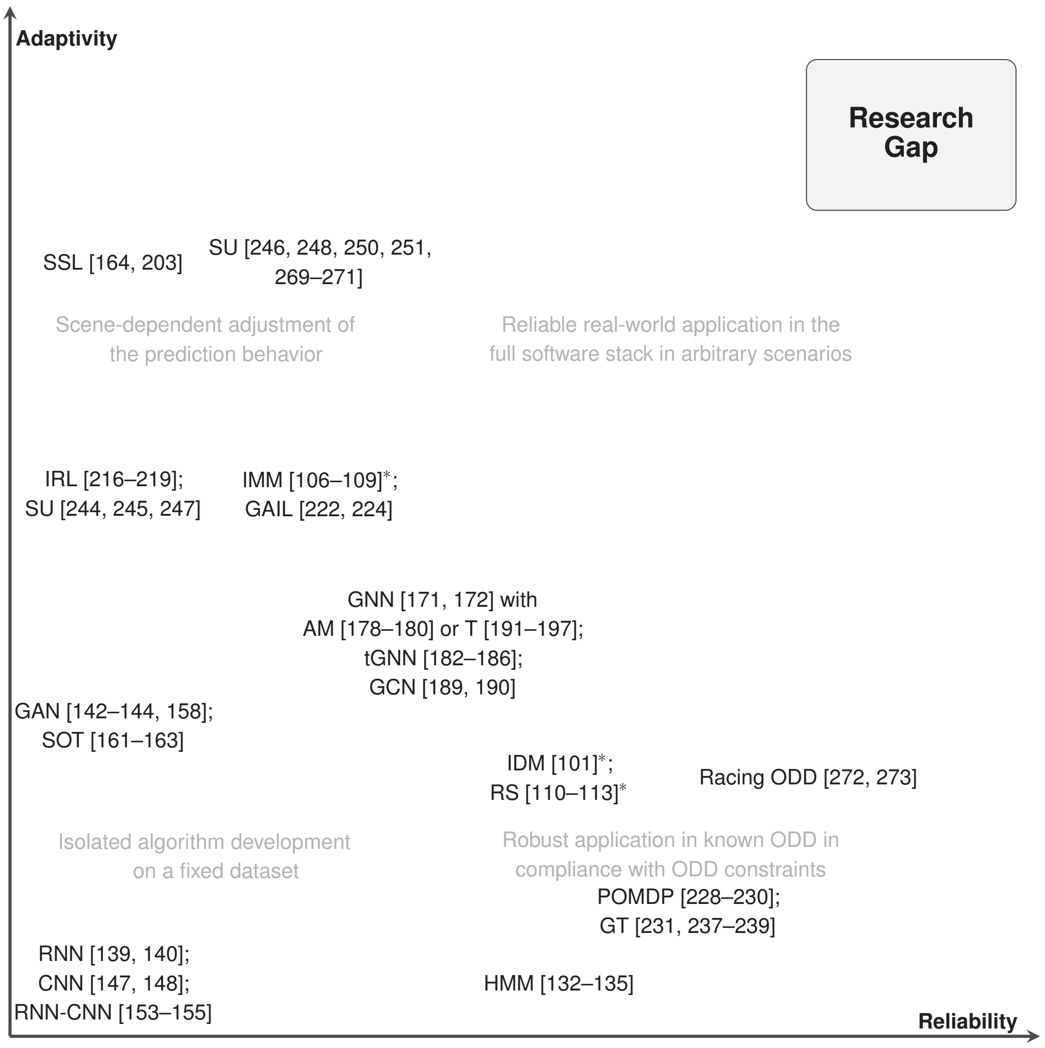

Adaptivity
Research Gap
SSL [164, 203]
SU [246, 248, 250, 251, 269–271]
Scene-dependent adjustment of the prediction behavior
Reliable real-world application in the full software stack in arbitrary scenarios
IRL [216–219]; SU [244, 245, 247]
IMM [106–109]*; GAIL [222, 224]
GNN [171, 172] with AM [178–180] or T [191–197]; tGNN [182–186]; GCN [189, 190]
GAN [142–144, 158]; SOT [161–163]
IDM [101]*; RS [110–113]*
Racing ODD [272, 273]
Isolated algorithm development on a fixed dataset
Robust application in known ODD in compliance with ODD constraints
POMDP [228–230]; GT [231, 237–239]
RNN [139, 140]; CNN [147, 148]; RNN-CNN [153–155]
HMM [132–135]
Reliability

Figure 2.7: Qualitative evaluation of the state of the art regarding reliability and adaptivity, which are the two identified criteria for a generic prediction framework. (*: short-term prediction)

Prediction methods with low reliability and low adaptivity are developed and optimized on isolated datasets. For the first prediction challenge [136], pattern-based methods based on RNNs and CNNs have been developed. New models with more advanced deep learning methods emerged with improved generalizability in the following years. Especially GNNs with different forms of attention mechanisms (AM) and transformers (T), target-based GNNs (tGNN), and graph convolutional networks (GCN) exceed former architectures in terms of generalizability [274]. The generalizability leads to improvements in both aspects. The adaptivity is enhanced because more general patterns are learned, while the reliability is improved through higher robustness against the input data variation. However, these methods show several limitations. Regarding adaptivity, the previously outlined decrease in accuracy and increased miss rates if applied to unseen data are not remedied. With a focus on reliability, the applicability is questionable because of high calculation times. A full-stack application of these algorithms has not yet been demonstrated and is constrained by the high efforts required in data engineering to prepare the methods for a specific ODD. In addition, the black-box behavior of deep learning methods can lead to unpredictable failures of the motion prediction module [250]. Thus, neither adaptive behavior nor reliable application is proven.

Moving up along the vertical axis, adaptivity is enhanced through more general problem formulations such as Inverse Reinforcement Learning (IRL) or Generative Adversarial Imitation Learning (GAIL). Alternatively, ensembles of multiple models, such as Interacting Multiple Models (IMM), are used to achieve adaptivity. Since these ensembles mainly consist of physics-based methods to limit the calculation effort, they only apply to short-term prediction. Another approach to achieving adaptivity is Self-Supervised Learning (SSL), which follows the idea of encouraging the model to train on unlabeled data. The approach can improve the adaptivity during the offline training process by forcing algorithms to autocomplete patterns. During the inference, SSL can be used to optimize the model on specific behavior. This approach improves the adaptivity in specific situations but is prone to overfitting and unstable learning behavior, which results in low reliability. Considering scenario understanding (SU) is another pivotal feature in establishing adaptivity. Similar to the human driving experience, scenario understanding can be used to supervise the forecast or analyze the scenario regarding the expected prediction error. While this group of methods shows improved adaptivity, their reliability in being deployed in a full software stack is not shown. In addition, the presented features to adapt the algorithm are not investigated under real-world input data. Furthermore, some methods can only detect prediction errors a posterior or make conservative assumptions, leading to frozen robots at application.

Along the horizontal axis, reliable applications of prediction methods are grouped. Maneuver-based prediction methods such as hidden Markov models (HMM) can be seen as reliable but only in a narrow field of scenarios because road traffic on a large scale is not graspable by a few maneuvers. Thus, the adaptivity is graded low. Physics-based approaches such as reachable sets (RS) are assessed to show a good compromise between adaptivity and reliability but are limited to short-term predictions. To apply reachable sets for long-term prediction, considering traffic rules is required to shrink their size. However, if an edge case occurs, which is an abnormal scene faced during inference, such as traffic participants violating these traffic rules, reachable sets need additional features to cope with it. So, the reliance on traffic rules is a considerable bottleneck to realize adaptive behavior [55]. Algorithms that fall under the field of interaction-aware planning are assessed to show reliable behavior. Partially observable Markov decision processes (POMDP) or game-theoretic approaches (GT) are well suited to model interactions in specific ODDs with known intentions of all objects. Their limitation is that the rule-based character does not scale for broad autonomous applications with unknown intentions of the objects. Reliable applications of GT-approaches are shown on $1/10$-vehicles in racing scenarios. Lastly, the full-scale applications in the racing ODD at high speeds can be seen as the most reliable prediction application in a specific ODD. An entire AV software stack on a full-scale vehicle at its dynamic limits requires reliable software for various reasons. For performant racing, the prediction has to ensure compliance with the ODD to enable aggressive ego-planing behavior. Especially in interactive scenarios, a reliable prediction is required to base the ego-motion planning on. Furthermore, computational constraints must be met, and noise and latency in the input data must be handled. The existing approaches for autonomous racing mainly focus on the compliance with simple race rules and assume race line following or parallel driving of the opponents. Regarding adaptivity, these approaches are assessed to be limited because of their focussed development for the target ODD of the race track. The transferability of the developed prediction algorithms from the race track to public roads has not been shown yet.

Concluding the review on the state of the art, three points can be stated. First, more advanced network architectures increase the generalizability in terms of improving reliability and adaptivity but show limitations regarding both criteria. Second, in both directions of reliability and adaptivity, approaches and shown applications exist that fulfill these criteria on an advanced level. Nevertheless, the complete fulfillment of these criteria has not yet been achieved. The third conclusion is that no method combines both criteria in a generic prediction framework. Based on these conclusions, the research gap can be identified (Figure 2.7): No generic motion prediction method exists that is reliably applicable in a full AV software stack on arbitrary real-world scenarios while ensuring accurate predictions. The gap leads to the research question of this thesis:

*How can reliable and adaptive prediction methods for autonomous driving be developed and combined in a generic framework?*

To answer the research question and fill the identified research gap, the following research items are proposed to be investigated:

**RI1** How can the detection input be improved to enable reliable application?
**RI2** How can black-box models be constrained to increase their reliability?
**RI3** How can multiple models be combined for adaptive prediction behavior?
**RI4** How can scenes be evaluated regarding the expected prediction performance?
**RI5** Which factors influence inaccurate predictions?

The first two research items, RI1 and RI2, focus on the aspect of reliability. This includes optimizing the detection input data as a crucial prerequisite for enabling a reliable motion prediction (RI1). RI2 investigates how the reliability of black-box models, which show a comprehensive prediction behavior but are unreliable in the application, can be improved by external constraints without losing accuracy. These two research items are investigated in Chapter 3.

After analyzing the reliability criterion, the remaining research items focus on establishing adaptivity. One aspect of it is combining multiple models from different prediction classes (RI3). Besides that, the feature of self-evaluation shall be analyzed (RI4). It is crucial knowledge to scene-dependently choose the most suitable prediction approach and correctly estimate the algorithmic limitations. The knowledge of the limitations can be used to collect edge-case data for further algorithm improvement or to dynamically adjust the predictions. RI5 comprises an in-depth analysis of the reasons for inaccurate predictions, which builds the baseline for dedicated algorithmic improvements and creates a comprehensive understanding for future applications. These three research items are examined in Chapter 4.

After the investigation of the five proposed research items, they are reviewed regarding their fulfillment (Section 5.1). Since these items focus on developing either reliable or adaptive motion prediction methods, combining them in a generic framework for motion prediction is discussed in Section 5.2. Finally, the research question is reviewed (Section 5.3) and an outlook on future research is given (Section 5.4).

# 3 Reliable Motion Prediction in Autonomous Racing

Reliable real-world application of motion prediction methods is a sparsely investigated issue in the state of the art. While there are promising results in simulation environments or on replayed data, the full-stack real-world application of developed algorithms to prove reliability is neglected. The following chapter addresses this issue by proposing a reliable motion prediction method for real-world applications, focussing on the research items RI1 and RI2 (Section 2.4). The methods are developed for and investigated in the inaugural events of the Indy Autonomous Challenge (IAC) and Autonomous Challenge at CES (AC@CES). They are deployed in the software stack of TUM autonomous motorsport [275, 276].

The two events' autonomous racing ODD offers a challenging high-speed environment to test developed methods in a full-stack autonomous car without risk for humans [277]. With the focus on tracking and prediction, three reasons can be named, which show the advantages of using this ODD to prove reliability. The high relative speeds between the ego vehicle, the environment, and the surrounding objects generate motion distortion in the sensors' inputs due to their rolling shutter, which leads to deteriorated detection performance in position estimation and an increased False-Positive (FP)-rate. In addition, the vehicles' high dynamic limits enable more volatile motion patterns, which must be tracked and predicted. Lastly, the non-cooperative character of the race can lead to aggressive and unexpected driving behavior, which has to be encoded to ensure an accurate motion prediction.

At first, the testbed and race format of the autonomous racing ODD are introduced (Section 3.1). Then, a multi-modal late fusion and object tracking method is presented (Section 3.2). It is revealed, to be an essential module to close the gap between detection algorithms, which only output single frames and are limited to the sensor measurements, and the prediction algorithms, which expect full object histories containing equidistantly sampled object states. Thus, tracking is crucial to enable reliable motion prediction. The late fusion approach is chosen to enhance the modularity of the stack. Further detection pipelines can be directly added, which is beneficial for the parallel development of independent pipelines. Since the events were the first autonomous head-to-head races, the required data quality to develop integral early fusion approaches was not given. Besides that, late fusion approaches are more robust against noise and less dependent on the sensor setup configuration. After introducing the proposed method, it is validated on real-world data from the race day at the AC@CES 2022.

A physics-constrained prediction method is presented in Section 3.3. It has two contrary goals to fulfill. On the one hand, there is the requirement to develop a sophisticated model that is able to predict a broad range of trajectories. On the other hand, quality constraints of kinematic feasibility and plausibility of the trajectory and a long-term prediction of $5\,\mathrm{s}$ have to be fulfilled, which are essential for stable ego-motion planning at high speeds [278, 279]. The model should additionally feature robustness against noisy input data. So, a reliable method for motion prediction with low data dependency for development and training but still sophisticated prediction behavior has to be implemented. Similar to introducing the fusion and tracking approach, the method and its respective validation are presented in this section.

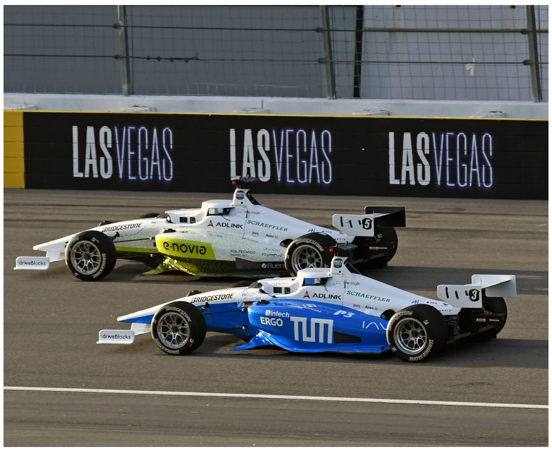
The two introduced methods for a reliable prediction in autonomous racing are discussed in Section 3.4. The discussion focuses on reviewing the methods for the ODD of autonomous racing and transferring them to new ODDs on public roads.

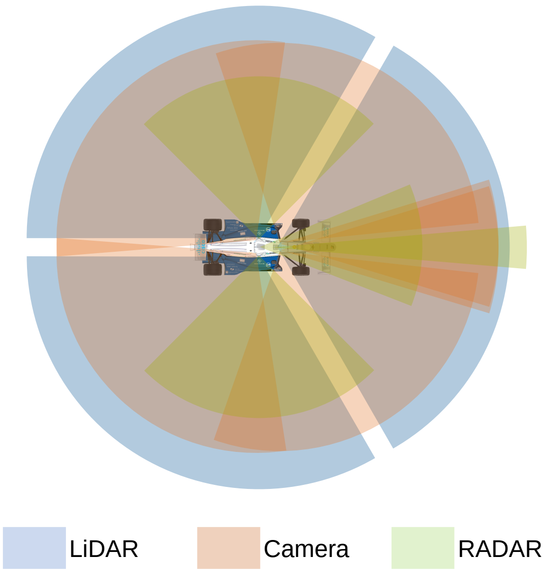
The proposed multi-modal late fusion and tracking method is published by the author [280]. The related software [281] and data [282] are provided open-source. The physics-constrained prediction is presented in the author's publication [283] with the software [284] and data [285] made publicly available.

## 3.1 Testbed and Race Format

The testbed for developing and evaluating the proposed methods comprises three elements: The autonomous race car, the race tracks, and the Hardware-in-the-Loop (HiL)-simulator. In addition, the race formats are introduced.

The autonomous race car AV-21 is based on the Dallara IL-15, used in the Indy Lights Series, and equipped with a multi-modal sensor setup (Figure 3.1a). The car's sensor setup consists of three Luminar H3 LiDARs, which cover $360\,°$ FoV, three RADAR sensors by Aptiv to cover the front (model: MRR) and side (model: ESR 2.5) area, and six Mako G319C cameras by Allied Vision to cover the full view around the vehicle (Figure 3.1b). Further details about the hardware setup of the race car can be found in [276].

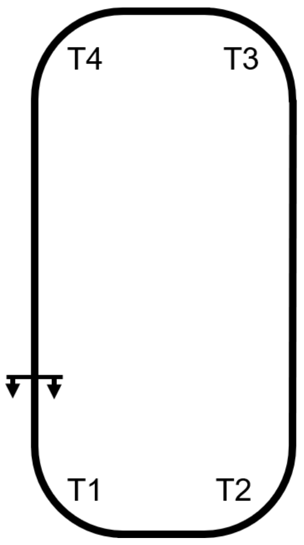

(a) The AV-21 on the Las Vegas Motor Speedway. Image Credits: Indy Autonomous Challenge



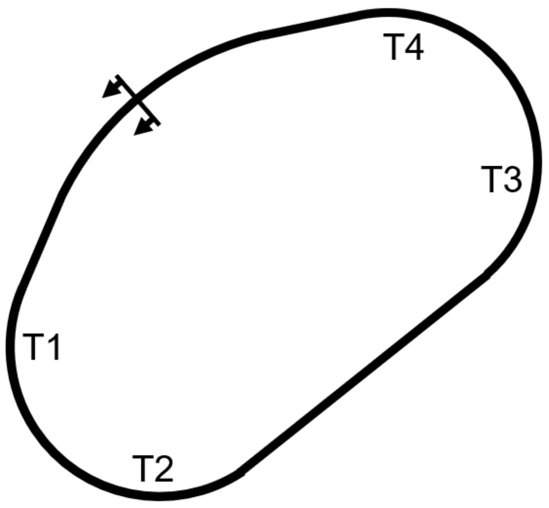
(b) Field of View of the AV-21's multi-modal sensor setup [39].

Figure 3.1: The autonomous race car AV-21 and its sensor setup used to develop and evaluate the methods.

The methods are validated in two competitions. The first competition is the IAC 2021 on the Indianapolis Motor Speedway (IMS) (Figure 3.2a), which comprised a simulation race and the final real-world race. The oval race track of the IMS has a length of $4023\,\mathrm{m}$ with a banking of $9\,°$ throughout the turns. While the final event was held as a single-vehicle competition due to the high complexity of developing a full AV software stack for the inaugural edition, the simulation race was held as a multi-vehicle competition with interactive overtaking maneuvers. The second competition is the AC@CES 2022 at the Las Vegas Motor Speedway (LVMS) (Figure 3.2b). Its track length is $2410\,\mathrm{m}$ with varying banking angle between $9\,°$ and $22\,°$ along the track. A head-to-head overtaking competition between two race cars was held at that event [286]. Both cars had to show a successful overtaking maneuver at each target speed step to continue the race. The



(a) Map of the Indianapolis Motor Speedway.



(b) Map of the Las Vegas Motor Speedway.

Figure 3.2: Maps of the race tracks on which the two competitions were held [276].

target speed increased by specific steps until one of the competitors was not able to overtake anymore. The overtaking procedure was orchestrated in a way that the leading car had to maintain the target speed while the trailing car had to complete the overtaking maneuver within a specific area along the race track. Then, the roles switched. The cars ran completely autonomously during the overtaking competition; the only signals they received were the current target speed and the race flag, which defined the cars' roles.

Due to the limited testing time on the race tracks, simulation was a crucial tool to develop and test the proposed methods. For that reason, a HiL-simulator is built upon which multiple software stacks can race against each other [275]. The HiL-simulator can simulate the entire autonomous software stack for up to ten agents in real-time, employing the same interfaces and callback functions as those on the real vehicle. All agents run on identical hardware, and their parameterization can vary individually. The only limitation is that the perception input is simplified to reduce computational load. A synthetic object list emulator, which serves as input to the tracking module, is implemented to achieve this. This generator is equipped with various features to mimic realistic perception behavior, including restricted sensor ranges, limited FoVs, the addition of Gaussian noise, variations in measured object states, and the simulation of FPs and False-Negatives (FNs). The object emulator tool, implemented by the author, is additionally used for the automated end-to-end optimization pipeline [287] and the full-stack latency analysis [264] of the software.

## 3.2 Multi-Modal Sensor Fusion and Object Tracking

Object fusion and tracking for autonomous racing have to fulfill two goals. At first, the information of heterogeneous detection pipelines has to be fused to maximize the information about the surrounding objects' states. The states shall be tracked over multiple time steps to establish a consistent object history. Second, the overall software delay, which correlates with the performance of the AV [264], has to be reduced. Thus, besides the required real-time capability of the module itself, the perception software delay shall be considered. A crucial constraint for developing the method is that no labeled data is given. Based on these requirements and the review of the state of the art (Subsection 2.2.3), the late-fusion and object tracking method shown in Figure 3.3 is proposed.

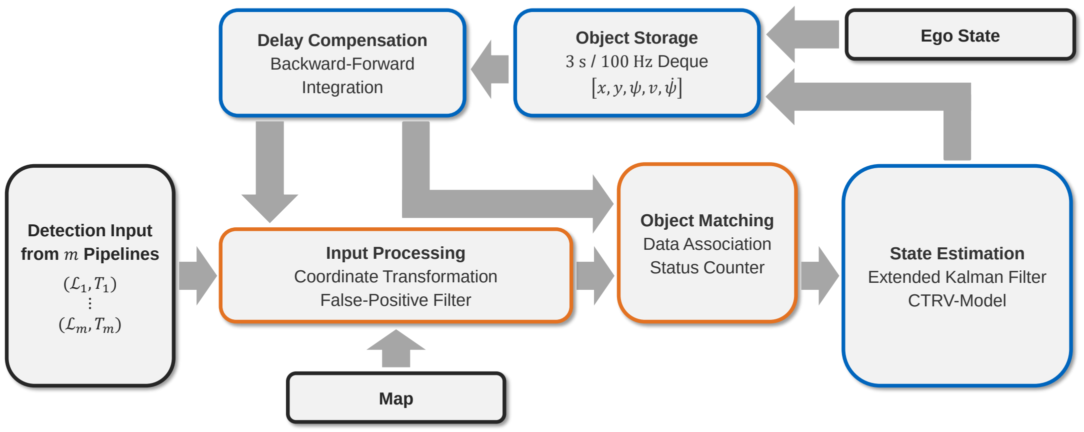


Figure 3.3: Interfaces (black) and structure of the multi-modal sensor fusion (orange) and object tracking (blue) method [280].

The multi-modal late fusion can handle input from multiple heterogeneous detection inputs. The raw input is first transformed into a global coordinate system before it is filtered for FPs. Two filters are applied. The first one filters for out-of-track objects based on the provided map. The second filter clusters multi-detections per object. Then, distance-based matching associates the filtered object lists with the currently tracked objects in chronological order. These multi-modal late fusion features are presented in Subsection 3.2.1. The object tracking is realized by an EKF with a kinematic motion model. The updated object states are added to the object storage. A central feature of the tracking method is the delay compensation. Using this feature, the delay from sensor recording to the receipt of the detected object list is considered. By this, motion prediction and ego-motion planning receive updated, optimized tracks of the surrounding objects. The tracking procedure is introduced in Subsection 3.2.2.

Based on the introduction of the method, the data used for the evaluation is analyzed in Subsection 3.2.3. The method's evaluation is presented in Subsection 3.2.4. The full parameterization of the fusion and tracking model is given in the Appendix A.1. The method and results presented in this section are based on the author's publication [280].

## 3.2.1 Multi-Modal Late Fusion

The detection input is passed through multiple input processing steps before the actual fusion by matching old tracked and newly detected objects. This input data processing comprising the transformation and FP-filters and the succeeding object matching are described in the following.

### Input Processing

The proposed method expects the input of the ego vehicle's dynamic state, a map detailing the track boundaries, and object lists from the detection pipelines (Figure 3.3). The number of detection pipelines $m$ is flexible and can be modularly extended, with no restriction on the type of sensors used. They can vary regarding detected object features, detection sensitivity, and cycle times. By this, heterogeneous sensor modalities can be merged to optimize the information about the surrounding objects. Each object in the object list $\mathcal{L}_i$ is characterized by the detection pipeline-dependent measured or estimated features and the respective uncertainties. In addition to the object list, the sensor time stamp $\mathcal{T}_i$ is part of the detection input. It is the time stamp of the sensor recording, so the whole perception delay of signal processing and algorithmic calculations is included. These detection input data in the form of the tuples $(\mathcal{L}_i, \mathcal{T}_i)$ are processed in chronological order, starting with the oldest.

After receiving the detection input, a coordinate transformation is applied to convert the local, vehicle-based coordinates to the global world coordinate system. The required ego state is received from the ego state estimation module. The received ego state is low-pass filtered to mitigate the high-frequency noise from internal state sensors such as the inertial measurement unit because ego and surrounding object state estimation have different time constants.

While FNs in the form of missing object detections in single inputs can be bridged by the matching procedure, additional filters of the transformed object list are implemented to reduce FPs. The reduction of FPs is crucial to avoid emergency maneuvers and is important to safely use sensitive sensor modalities. Two plausibility checks to filter FPs are applied. First, objects located outside the track boundaries are removed using the provided HD map, a valid assumption for the closed race track ODD. Second, a check is conducted to identify overlapping detections within the object list from one modality. A $k-d$ tree clustering [288] with a fixed distance threshold is implemented for this purpose.

## Object Matching

After processing the detection inputs, object matching occurs. The Hungarian method [78] is used to solve the assignment problem between newly detected and old tracked objects. The Euclidean distance is used as cost function, which is revealed to be the most robust and calculation-efficient option. The investigation of further options showed that the IoU and the NLL are less robust, require higher calculation time, and need further state variables [289]. Only with idealized, low-noise data the more complex cost functions showed an improvement. The association problem to obtain the overall minimum distance costs between $n_{\text{track}}$ old tracked and $n_{\text{in}}$ new detected objects can result in three cases:

1. **New Unmatched Object**
   In cases where the detection object list contains more elements than currently tracked objects ($n_{\text{in}} > n_{\text{track}}$) or the inter-object distance exceeds the maximum match distance, new objects are initialized. This includes assigning an UID, setting up an object-specific status counter, and initializing the new object's state estimation.
2. **Old Matched Object**
   A successful match is given if a newly detected object is assigned to an old tracked object and the inter-object distance is below the maximum match distance. In this case, the current state estimation and object history are updated by the received detection, and the object-specific status counter is incremented by the detection pipeline-specific match value. The corrected posterior state is saved in the object storage.
3. **Old Unmatched Object**
   If there are more old tracked objects than the detection object list contains ($n_{\text{track}} > n_{\text{in}}$) or the inter-object distance is above the maximum match distance, old unmatched objects remain. In this case, the predicted object state (prior estimate) is stored in the object storage without a measurement update. Also, the status counter of the object is decreased. If it reaches zero, the object is removed from the storage.

The mentioned match distance and status counter specify the matching behavior and the handling of FNs. The former variable defines the distance threshold to match the detected and tracked object. A higher distance reduces the FP-rate, avoiding multiple tracks of one object. However, in the case of wheel-to-wheel racing, it can lead to incorrect merges of two adjacent objects. In contrast, a low match distance results in a higher probability of multiple tracks for one object since the detected and tracked positions must lie closer to each other. The status counter bridges FNs, which are inputs with missing detections of existing objects. It represents the number of unsuccessful matches until the object is discarded and not tracked anymore.

Thus, it symbolizes the trust in an object's existence. It is initialized for each new object by a positive integer (case 1) and increased by the detection pipeline-specific match counter if a successful match occurs (case 2). The counter is always decreased by $1$ if an unsuccessful match takes place (case 3). Thus, it defines how many detection inputs without successful matches are left until a tracked object is discarded. The status counter is limited to an upper threshold. By this, the deletion of objects is secured within a specific number of unsuccessful detections.

### 3.2.2 Object Tracking

Based on the introduced fusion, the actual object tracking through the state estimation is performed. Besides the state estimation, the object storage, filter parameterization, and novel perception delay compensation are presented.

#### State Estimation

The state estimation is conducted by the EKF. The EKF is selected due to its capability to approximate non-linear models compared to the linear KF. In addition, it offers the advantage of low calculation effort compared to the UKF and the PF because the computationally efficient recursive update from the KF is maintained and the determination of the first-order partial derivatives is of small effort for the given low-dimensional system. Moreover, from the application's perspective, the EKF allows for interpretable parameter optimization in scenarios where large data is unavailable in contrast to the UKF [290]. The EKF builds on the following state-space model:

$$\boldsymbol{x}_t = f(\boldsymbol{x}_{t-1}, \boldsymbol{u}_t) + \boldsymbol{w}_t \tag{3.1}$$

$$\boldsymbol{z}_t = h(\boldsymbol{x}_t) + \boldsymbol{v}_t \tag{3.2}$$

$\boldsymbol{x}_t$ represents the object's state vector at the time $t$. It is derived from the state transition model $f(\boldsymbol{x}_{t-1}, \boldsymbol{u}_t)$, which gets the old object state $\boldsymbol{x}_{t-1}$ and the control vector $\boldsymbol{u}_t$ as input. By means of the observation model $h(\boldsymbol{x}_t)$, the observation states in the vector $\boldsymbol{z}_t$ are calculated. The filter model assumes zero-mean multivariate Gaussian noise, which is quantified by the process noise $\boldsymbol{w}_t$ and observation noise $\boldsymbol{v}_t$. Based on this state-space model, the state estimation is conducted alternating between the prior estimation (Predict) and the posterior correction (Update) [291]:

**Predict**

$$\hat{\boldsymbol{x}}_{t|t-1} = f(\hat{\boldsymbol{x}}_{t-1|t-1}, \boldsymbol{u}_t) \tag{3.3}$$

$$\boldsymbol{P}_{t|t-1} = \boldsymbol{F}_t \boldsymbol{P}_{t-1|t-1} \boldsymbol{F}_t^\top + \boldsymbol{Q}_t \tag{3.4}$$

$$\boldsymbol{F}_t = \left.\frac{\partial f}{\partial \boldsymbol{x}}\right|_{\hat{\boldsymbol{x}}_{t-1|t-1}, \boldsymbol{u}_t} \tag{3.5}$$

**Update**

$$\tilde{\boldsymbol{y}}_t = \boldsymbol{z}_t - h(\hat{\boldsymbol{x}}_{t|t-1}) \tag{3.6}$$

$$\boldsymbol{S}_t = \boldsymbol{H}_t \boldsymbol{P}_{t|t-1} \boldsymbol{H}_t^\top + \boldsymbol{R}_t \tag{3.7}$$

$$\boldsymbol{H}_t = \left.\frac{\partial h}{\partial \boldsymbol{x}}\right|_{\hat{\boldsymbol{x}}_{t|t-1}} \tag{3.8}$$

$$\boldsymbol{K}_t = \boldsymbol{P}_{t|t-1} \boldsymbol{H}_t^\top \boldsymbol{S}_t^{-1} \tag{3.9}$$

$$\hat{\boldsymbol{x}}_{t|t} = \hat{\boldsymbol{x}}_{t|t-1} + \boldsymbol{K}_t \tilde{\boldsymbol{y}}_t \tag{3.10}$$

$$\boldsymbol{P}_{t|t} = (\boldsymbol{I} - \boldsymbol{K}_t \boldsymbol{H}_t) \boldsymbol{P}_{t|t-1} \tag{3.11}$$

The predict step at time $t$ comprises the prior estimation of the object's state $\hat{\boldsymbol{x}}_{t|t-1}$ (Equation (3.3)) and the related estimation uncertainty (Equation (3.4)). The state estimation results from the state transition of the former posterior estimate to the current time step. The uncertainty of the prior state estimation is given by the covariance matrix $\boldsymbol{P}_{t|t-1}$. It is the sum of the posterior covariance matrix of the former time step $\boldsymbol{P}_{t-1|t-1}$ multiplied by the state transition matrix $\boldsymbol{F}_t$ and the covariance matrix of the process noise $\boldsymbol{Q}_t$. As Equation (3.5) shows, the state transition matrix is the first-order partial derivative of the state transition model evaluated at the former posterior estimate $(\hat{\boldsymbol{x}}_{t-1|t-1}, \boldsymbol{u}_t)$, i. e. a local linearization is performed. The update step uses a measurement input to update object state and uncertainty estimations. To achieve this, the measurement residual $\tilde{\boldsymbol{y}}_t$ between the received measurement $\boldsymbol{z}_t$ and the measurement prediction based on the prior state $h(\hat{\boldsymbol{x}}_{t|t-1})$ is calculated (Equation (3.6)). The uncertainty of the measurement residual is given by the residual covariance $\boldsymbol{S}_t$, which is calculated by the covariance matrix of the observation noise $\boldsymbol{R}_t$, the prior covariance matrix $\boldsymbol{P}_{t|t-1}$ and the observation matrix $\boldsymbol{H}_t$ (Equation (3.7)). The latter is the first-order partial derivation of the observation model around the current prior estimate (Equation (3.8)). To perform the update step, the Kalman gain $\boldsymbol{K}_t$ is needed, which is derived from the prior covariance matrix, the observation matrix, and the residual covariance (Equation (3.9)). Then, the updates of object state $\hat{\boldsymbol{x}}_{t|t}$ and related covariance matrix $\boldsymbol{P}_{t|t}$ are calculated (Equation (3.10) and (3.11)). The Kalman gain specifies the weighting of the update step. The magnitude of the Kalman gain can be interpreted as the trust in the new measurement compared to the prior estimation.

A kinematic, point-mass model is used for the state transition model. A CTRV-model is selected. The selection is based on two assumptions. First, the yaw rate shall be incorporated to model the objects' movement in turns accurately. Second, the acceleration shall be neglected to avoid oscillating speed estimation because of the additional integration part from a non-measured variable. Except for standing start situations where high accelerations occur, the acceleration is of low magnitude compared to the average object speed on oval race tracks, which additionally justifies the assumption of constant speed. The resulting model equations of the CTRV-model are:

$$\begin{pmatrix} x \\ y \\ \psi \\ v \\ \dot{\psi} \end{pmatrix}_t = \begin{pmatrix} x - v\Delta t \sin\psi \\ y + v\Delta t \cos\psi \\ \psi + \Delta t \dot{\psi} \\ v \\ \dot{\psi} \end{pmatrix}_{t-1} \tag{3.12}$$

$x$ and $y$ denote the longitudinal and lateral position of the object. $\psi$ and $\dot{\psi}$ are the object's yaw angle and its derivative, while $v$ represents the object's speed. $\Delta t$ is the size of the time step. By default, it is set to $\Delta t = 1/f_{\text{EKF}}$ with $f_{\text{EKF}} = 100\,\text{Hz}$. The small step size reduces the approximation error of the state estimation model (Equation (3.12)) and the linearization error of the EKF (Equation (3.5) and (3.8)). It's important to mention that the time step $\Delta t$ used for the state estimation is not influenced by the module's callback frequency or the detection pipelines' frequency. As a consequence, the prediction step of the state estimation is constantly executed to iteratively estimate the state of the tracked objects, and the update step of the filter is only called when a successful match occurs (case 2). This approach ensures that the object states are equidistantly sampled. In addition, this design enhances the module's capability to handle missing detection inputs.

## Object Storage

The object states, either predicted or updated by a received measurement, are stored in a double-ended fixed-length queue. Its length is determined by the EKF update frequency and the desired temporal object history. The default setting is a $3\,\text{s}$ object history, which implies together with the sampling frequency of $f_{\text{EKF}} = 100\,\text{Hz}$ a storage length of $300$.

## Parameterization

The initialization of the estimated object state plays a crucial role in the transition when a new object is initialized. Although the measured features from the respective detection pipelines serve as the basis for state initialization, additional assumptions are made to enhance the accuracy of the initial state estimation. The first assumption is that all detected vehicles move around the race track in the same direction. Thus, the orientation of the track's centerline is used as an initial estimate for the object's yaw angle if the detection pipeline does not measure it. The second assumption involves the object's speed, which is assumed to be of a similar magnitude as the ego vehicle's speed. In the case that the object's speed is not given in the measurement vector, the ego speed multiplied with an empirical scalar factor $k_v$ is used to initialize the object's speed estimate. Both estimations are made with high measurement uncertainties, which results in small Kalman gains. Thus, in a few steps, the EKF can correct these variables based on the kinematic dependencies. The yaw angle assumption is additionally applied to every measurement input, which does not include this variable. It was revealed to enhance the overall stability of the estimation.

The covariance matrix of the process noise $\boldsymbol{Q}_t$ is parameterized based on the expected deviation of the tracked objects from the CTRV-behavior. Assumptions are made on how much the race car can change its state within its dynamic limits in a time step $\Delta t$. Based on these assumptions, the process noise is parameterized, representing the error between the CTRV-model and the ground truth movement. The covariance matrix of the measurement noise $\boldsymbol{R}_t$ is individually parameterized per detection pipeline. Low measurement uncertainties for the position measurement of the LiDAR detection and low uncertainties for the speed measurement of the RADAR are assumed. A relevant fact regarding the tuning of the measurement noise matrix is the fact that the output of the detection algorithms is used as measurement variables in contrast to direct sensor measurements. Thus, the measurement noise relies on the properties of the detection algorithm, such as reduced position accuracy with higher object distance because of the limited sensor resolution. To cope with this issue, adaptive measurement matrices in dependency of the object distance have been investigated [292, 293]. However, the missing data availability prevented appropriate tuning to achieve improvements in real-world applications.

## Delay Compensation

The proposed method incorporates a feature for functional delay compensation. The functionality is designed to compensate for the accumulated delay from the sensor trigger until the detection input is received in the fusion and tracking module. This delay compensation through backward-forward integration (Figure 3.4) allows for the output of an updated, high-frequency sampled tracked state while accounting for delayed detection inputs.

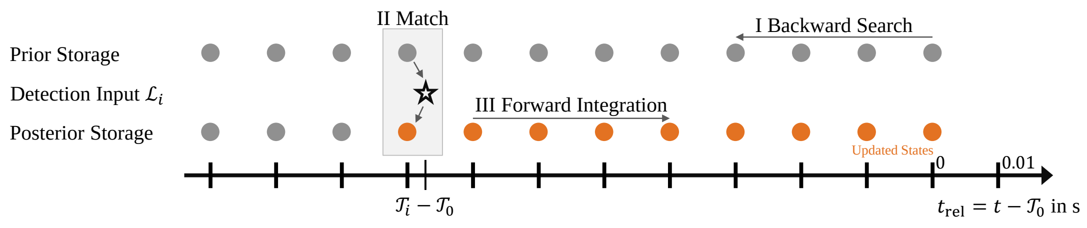

II Match
I Backward Search
Prior Storage
Detection Input $\mathcal{L}_i$
III Forward Integration
Posterior Storage
Updated States
0
0.01
$\mathcal{T}_i - \mathcal{T}_0$
$t_{\text{rel}} = t - \mathcal{T}_0$ in s

Figure 3.4: Temporal representation of the delay compensation. A backward search in the object storage is conducted to match the past object state to the detection input. The updated past object state is used for the forward integration to update all succeeding states.

The compensation processes the chronologically sorted detection inputs, starting with the oldest. The time stamp of the received sensor data $\mathcal{T}_i$ is used in the backward step to search the observation storage for all past object states corresponding to that time stamp. These past object states are utilized to match the respective detection input $\mathcal{L}_i$. In cases of a successful match, the update step of the EKF is applied to the matched past object state (Equation (3.6) - (3.11)). The optimized past object state is then used to correct all subsequent states. This correction is accomplished through iterative prediction steps (Equation (3.3) - (3.5)). Thus, the state transition model and the state uncertainty are forward integrated based on the optimized past object state. The correction is conducted either up to the time stamp of the next detection input or, if only one detection input is available, up to the current tracking time stamp. In the first case, the object matching function is called again to associate the next detection input with the corrected object storage entries in the past. Then, the forward integration is continued.

An exemplary scene is depicted in Figure 3.5. Before processing the received detection input, the tracked object state is synchronized with the ego time stamp through forward integration (gray cross, Equation (3.3) - (3.5)). The only detection within the track boundaries originates from the LiDAR (orange-black star & box) with a delay of $211\,\mathrm{ms}$, which corresponds to $13.7\,\mathrm{m}$ movement of the object at $233\,\mathrm{km\,h^{-1}}$. The implemented delay compensation considers the delay from sensor triggering to detection input receipt and performs a backward search within the object storage (gray dotted). The object state entry with the smallest temporal difference from the detection time stamp is selected to perform the object matching (gray bold dot). In this case, the temporal difference is only $1.1\,\mathrm{ms}$. Any inaccuracies resulting from equidistant sampling are negligible, given the high sampling frequency of $100\,\mathrm{Hz}$. The past object state matches with the LiDAR detection (case 2), so the update step of the EKF (Equation (3.6) - (3.11)) can be executed. The updated object state (orange bold dot) is used to correct the entries in the object storage by iteratively executing the prediction step of the EKF to forward integrate the estimated state and related uncertainty (orange dotted). An updated object history (orange dots) and current state (orange cross) result.

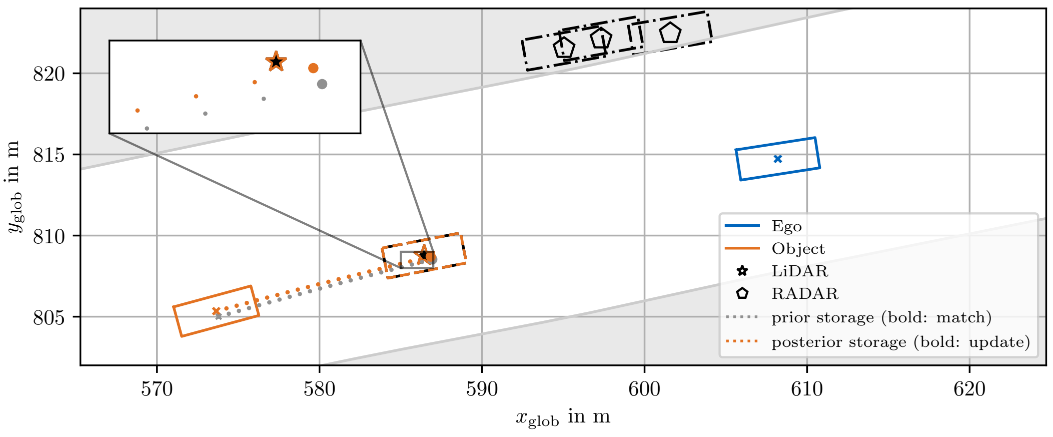


Figure 3.5: Exemplary high-speed real-world scene at AC@CES 2022 [280]. The ego speed is $255\,\mathrm{km\,h^{-1}}$, the object speed is $233\,\mathrm{km\,h^{-1}}$. The FPs of the RADAR are out of track (gray area) and removed by the out-of-track filter. The LiDAR's perception delay of $211\,\mathrm{ms}$ is compensated by the backward-forward integration, which corresponds to $13.7\,\mathrm{m}$ distance compensation.

### 3.2.3 Data Analysis

The evaluation data is analyzed to put the presented results in the context of the investigated scenarios. The data are recorded by the perception software stack on the AV-21. Two detection pipelines are used: A LiDAR and a RADAR detection algorithm [276]. The data contain high-speed overtaking maneuvers against two different opponents in the semifinal and final runs of the AC@CES 2022. They comprise overtaking maneuvers between $35\,\mathrm{m\,s^{-1}}$ and $75\,\mathrm{m\,s^{-1}}$. The data is unlabeled. Thus, just the raw object list as outputted by the perception algorithms and the respective time stamps are given. In total, $11\,052$ detection input samples of the LiDAR algorithm and $4723$ samples of the RADAR algorithm are recorded. These detection input samples stem from $28$ valid objects detected by the LiDAR pipeline and from $26$ valid detected objects by the RADAR pipeline. The difference can be explained by the diverging FoVs (Figure 3.1b). In the scope of this evaluation, an object is defined as valid if it is identified at least three times without being discarded. The high total numbers of valid objects result from the fact that during the races, the opposing car left the FoV of the ego car after every maneuver. Thus, the opposing car is reinitialized in the tracking for every overtaking maneuver. The data is available open-source [282] and analyzed in the following over various metrics (Figure 3.6).

The first violin plot shows the distribution of the object speed $v_{\text{obj}}$. Peaks at the target speed stages can be seen. The object speed primarily influences the transition when a new object is initialized until it is stably tracked. Higher speed leads to a higher difference in the position between frames to match the object, exacerbating reliable tracking. The second violin plot shows the relative speed between the target and the ego object $v_{\text{rel}}$. The peak at $0\,\mathrm{m\,s^{-1}}$ represents scenarios where both cars drive behind each other before an overtaking maneuver is started. The peaks at $-8\,\mathrm{m\,s^{-1}}$ and $8\,\mathrm{m\,s^{-1}}$ are the speed difference during the overtakes. The relative speed between the ego and the surrounding object is of high relevance because it correlates with distortions of the sensor input due to the sensors' rolling shutter. The effect leads to a deteriorated position estimation and an increase of the FP-rate in the detection input. The third distribution contains the object distance $d_{\text{obj}}$. It ranges from below $10\,\mathrm{m}$ up to $105\,\mathrm{m}$ as the maximum detection range for the RADAR, and up to $98\,\mathrm{m}$ for LiDAR detections. The peaks of this distribution are within a distance of $30\,\mathrm{m}$ to $50\,\mathrm{m}$, which is the characteristic position the cars have before an overtake. The object distance influences the detection accuracy, especially the position measurement. If the object distance is converted in the ego object's coordinate frame, the local longitudinal position $x_{\text{loc}}$ can be analyzed, which is given in the fourth plot. Since the RADAR has no rear view, its detection is only in the front of the car. The LiDAR

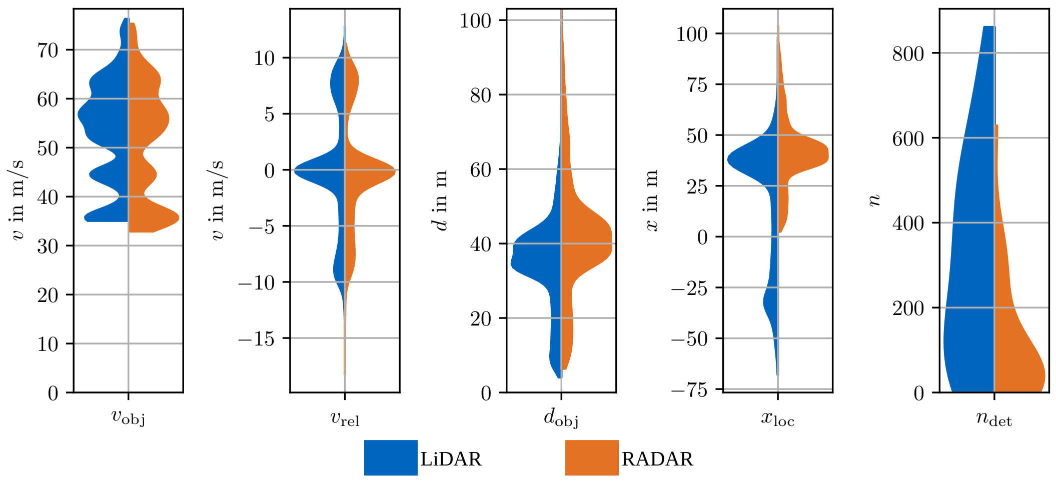


Figure 3.6: Data analysis of the recorded real-world data at the AC@CES 2022 [280]. The data distribution of raw object lists from the LiDAR and the RADAR detection algorithms is shown. Distributions over the object speed $v_{\mathrm{obj}}$, relative speed $v_{\mathrm{rel}}$, object distance $d_{\mathrm{obj}}$, the object's local longitudinal position $x_{\mathrm{loc}}$ and the number of detections per track $n_{\mathrm{det}}$ are given.

algorithm detects objects in the rear up to $57\,\mathrm{m}$. The lower detection range to the rear can be explained by the occlusions and reflections caused by the rear wing of the race car. The number of detections per tracked object visualized in the last violin plot $n_{\mathrm{det}}$ ranges from three measurements up to 800 successfully matched detections, which corresponds to more than $40\,\mathrm{s}$ of consecutive tracking.

### 3.2.4 Results

The validation of the proposed method is performed on the introduced real-world data. At first, the overall tracking performance is evaluated. Next, a sensitivity analysis is conducted to analyze the performance under varying parameter settings. Afterward, the delay compensation approach is analyzed. The last part of the validation is the method's transient behavior.

#### Tracking Perfomance

The tracking performance comprises two indicators: the measurement residual (Equation (3.6)) and the precision. The measurement residual is an established metric for assessing state estimation methods, particularly in cases where ground truth data regarding the exact position of objects is unavailable. For the given method, the residuals of the longitudinal and lateral position, the yaw angle, and the velocity (RADAR only) are evaluated. The target behavior of the residuals is a zero-mean Gaussian distribution with the same standard deviation as assumed in the observation noise [294]. This ensures effective measurement noise reduction by the EKF. The second criterium of precision is defined as:

$$\mathrm{Precision} = \frac{\mathrm{TP}}{\mathrm{FP}+\mathrm{TP}} \tag{3.13}$$

$\mathrm{TP}$ represents True-Positives (TPs), which are correctly identified objects, $\mathrm{FP}$ represents False-Positives (FPs), which are incorrectly detected ghost-objects. The determination of TP-objects is done manually for the given data. TP-objects are all objects within a sensor's range with at least three consecutive detections. Each TP-object receives a UID while it remains within the detectable area. FPs are ghost objects with assigned UIDs or real objects that are briefly tracked and then discarded by the algorithm. It is important to mention that this definition of TP and FP differs from the one used in common detection benchmarks. In contrast to their conventional frame-by-frame approach, the entire scenario is considered to evaluate the algorithm's precision in the given case. This scenario-based definition is more appropriate for evaluating the fusion and tracking methods because it considers the method's consistency in keeping a tracked object. The tracking performance of the EKF in combination with a CTRV-model is compared against the kinematic models of CV and CTRA (Table 3.1, Row $1-3$).

Table 3.1: Tracking performance and sensitivity analysis of the measurement residuals (mean $\mu$, standard deviation $\sigma$) and precision [280]. The precision's absolute (abs.) and relative (rel.) values compared to the default setup of EKF with CTRV are given.

| | | $x_{\mathrm{loc}}$ in m | | $y_{\mathrm{loc}}$ in m | | $\psi$ in ° | | $v$ in m s$^{-1}$ | | Precision | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $\mu$ | $\sigma$ | $\mu$ | $\sigma$ | $\mu$ | $\sigma$ | $\mu$ | $\sigma$ | abs. | rel. |
| EKF, CTRV | | −0.08 | 0.73 | 0.03 | 0.38 | −0.11 | 2.58 | −0.21 | 2.76 | 0.5 | – |
| EKF, CV | | −0.10 | 0.72 | 0.63 | 0.56 | 3.15 | 2.89 | −0.13 | 2.94 | 0.37 | −0.26 |
| EKF, CTRA | | −0.06 | 0.58 | 0.03 | 0.38 | −0.13 | 2.59 | −0.12 | 2.69 | 0.5 | 0.0 |
| Node frequency $f_{\mathrm{node}}$ in Hz | 10 | −0.03 | 0.80 | 0.02 | 0.35 | **−0.07** | 2.69 | −0.24 | 3.30 | 0.42 | −0.15 |
| | 20 | −0.02 | 0.72 | 0.01 | 0.33 | −0.10 | 2.65 | −0.20 | 3.31 | 0.53 | 0.06 |
| | 100 | −0.09 | 0.74 | 0.03 | 0.38 | −0.12 | 2.58 | −0.23 | 2.96 | 0.5 | 0.0 |
| Bounds filter $d_{\mathrm{OBF,out}}$ in m | 0 | −0.08 | 0.74 | 0.03 | 0.40 | −0.12 | **2.55** | −0.23 | 3.38 | 0.42 | −0.15 |
| | 0.75 | −0.09 | 0.73 | 0.03 | 0.38 | −0.12 | 2.64 | −0.22 | 2.87 | 0.5 | 0.0 |
| Match distance $d_{\mathrm{MTC}}$ in m | 1 | −0.06 | **0.32** | 0.03 | 0.24 | −0.16 | 2.63 | **0.02** | **1.39** | 0.13 | −0.74 |
| | 2 | −0.04 | 0.50 | 0.03 | 0.30 | −0.11 | 2.56 | −0.04 | 2.02 | 0.23 | −0.54 |
| | 7 | −0.08 | 0.80 | 0.03 | 0.40 | −0.11 | **2.55** | −0.29 | 3.60 | 0.53 | 0.06 |
| Match counter threshold $t_{\mathrm{MTC}}$ | 6 | −0.07 | 0.71 | 0.03 | 0.35 | −0.11 | 2.56 | −0.17 | 2.81 | 0.38 | −0.24 |
| | 12 | −0.08 | 0.73 | 0.03 | 0.37 | −0.09 | 2.59 | −0.21 | 2.82 | 0.46 | −0.08 |
| | 37 | −0.08 | 0.74 | 0.03 | 0.38 | −0.11 | 2.58 | −0.23 | 2.96 | 0.5 | 0.0 |
| Filter frequency $f_{\mathrm{EKF}}$ in Hz | 10 | −0.06 | 1.51 | 0.02 | 0.33 | −0.42 | 2.62 | −0.22 | 3.34 | 0.31 | −0.38 |
| | 50 | −0.06 | 0.74 | 0.03 | 0.36 | −0.15 | **2.55** | −0.20 | 2.98 | 0.53 | 0.06 |
| | 200 | −0.08 | 0.81 | 0.03 | 0.41 | −0.08 | 2.64 | −0.24 | 2.74 | 0.52 | 0.03 |
| Merge distance $d_{\mathrm{MRG}}$ in m | 1.7 | −0.08 | 0.69 | 0.03 | 0.37 | −0.12 | 2.60 | −0.14 | 2.73 | 0.36 | −0.28 |
| | 3.4 | −0.09 | 0.72 | 0.03 | 0.37 | −0.12 | 2.58 | −0.20 | 2.93 | 0.46 | −0.08 |
| | 7.2 | −0.06 | 0.71 | 0.03 | 0.38 | −0.10 | 2.60 | −0.19 | 2.61 | 0.53 | 0.06 |
| Sensor modality | L | **0.00** | 0.61 | **0.00** | **0.22** | **−0.07** | **2.55** | – | – | **0.59** | **0.17** |
| | R | −0.20 | 1.18 | −0.03 | 0.97 | −0.27 | 3.57 | −0.08 | 2.30 | 0.32 | −0.36 |

The combination EKF with CTRV shows a low biased longitudinal residual. It can be explained by the negative bias on the speed estimation. The same correlation can be seen in the longitudinal residual's standard deviation of $0.73\,\mathrm{m}$, which results from the speed standard deviation of $2.76\,\mathrm{m\,s^{-1}}$. Compared to the ego-object distance (Figure 3.6), this deviation is assessed to be sufficiently low to ensure performant ego-motion behavior. In the lateral direction, the majority of residuals fall within $\pm 1\,\mathrm{m}$, following the $3\sigma$-rule applied to the standard deviation of $0.38\,\mathrm{m}$. It is of special interest to adjust the wheel-to-wheel racing behavior and lateral safety distance of the ego vehicle. The heading residual has a standard deviation of $2.58\,°$. It results from deriving the object's heading from the track centerline's orientation. The velocity residuals reflect the deviation between the RADAR's measurement and the CTRV-model because it is the only detection pipeline that measures this variable. The method shows a low precision of $0.5$. The values can be explained by an analysis of TP- and FP-objects in the validation data. In the used data, FPs are mainly objects tracked briefly at the edge of the sensor range that are discarded because of missing rematches due to inconsistent object detections. Thus, multiple UIDs are set up for the same object before it fully enters the sensors' FoVs and is detected and tracked consistently. The design decision to initialize an object after three matches is however reasonable because FPs at the edge of the sensors' FoVs do not influence the ego behavior because of the high distance. At the same time, the design allows for early initialization of objects when they finally enter the sensors' FoVs, which is then of high relevance for the ego behavior if the object comes closer.

In comparison, the change to the CV-model results in an equal longitudinal but increased lateral residuals. It highlights the impact of the turn rate to adequately model the object's behavior in turns. The positive lateral bias shows the inability to model the object's motion into the left turns. In correlation with the lateral residual, the heading residuals increase due to the constant yaw angle assumption. The speed estimation remains equal. Regarding the precision, the simplified motion model decreases by $-26\,\%$. If the acceleration is considered (CTRA), the longitudinal position and speed residuals improve. A positive influence of acceleration modeling can be observed. However, a qualitative evaluation of transient scenarios such as overtaking with longitudinal acceleration revealed that the CTRA-model is prone to longitudinal oscillations, especially when the speed is not measured. Thus, there is a trade-off for real-world application, and the necessity of using the less robust CTRA must be analyzed regarding the expected object behavior. The precision remains unchanged between the CTRV- and CTRA-model.

## Sensitivity Analysis

The sensitivity analysis shall give insights into the tracking performance under different parameter settings. Seven model parameters and the two sensor modalities are varied around their default value. Table 3.1, from row 4, provides the results of the analysis. The default setting for each variation is given by EKF + CTRV.

At first, the ROS2-node callback frequency $f_{\text{node}}$ is analyzed. With a $90\,\%$-quantile of $9.27\,\text{ms}$ on a single Intel i7-9850H CPU core, the frequency is varied from $10\,\text{Hz}$ up to $100\,\text{Hz}$ around the default frequency of $50\,\text{Hz}$. If the frequency is increased no improvement of the measurement residuals is achieved. It could be interpreted that due to the delay compensation, the tracking performance is less dependent on low delays to receive the detection input. A lower node frequency yields a deterioration of the speed residual due to missed RADAR measurements and reduced precision because of a lower number of LiDAR and RADAR inputs.

The outer out-of-bounds filter distance $d_{\text{OBF,out}}$ is used to specify the distance from the track boundary within detections are declared as FPs and removed. The default value is $0.3\,\text{m}$. The main influence of the out-of-bounds filter distance is on the FP-rate. A deterioration of precision by $15\,\%$ is observed when the out-of-bounds filter is set to $0$. Thus, the majority of FPs are detected close to the outer track boundary, which can be verified by an unchanged precision for $d_{\text{OBF,out}} = 0.75\,\text{m}$.

The match distance $d_{\text{MTC}}$ specifies the threshold for a successful match. The variation around the default value of $5.0\,\text{m}$ reveals a trade-off. While the residuals of the longitudinal position and velocity improve with lower match distances because only detections close to the estimation are considered, the precision decreases because of less successful matches and more FPs. For $d_{\text{MTC}} = 7.0\,\text{m}$, the precision shows a small increase, which indicates that the detections are within the specified threshold to match the tracked objects.

The upper threshold of the match counter $t_{\text{MTC}}$, indicating the limit for the number of misdetections before an object is removed from the object storage, is set to $25$ by default. A reduction to $12$ and $6$ yields an increase in the FP-rate because the objects are kept for less time in the storage. A match counter threshold set to $37$ does not affect the FP-rate.

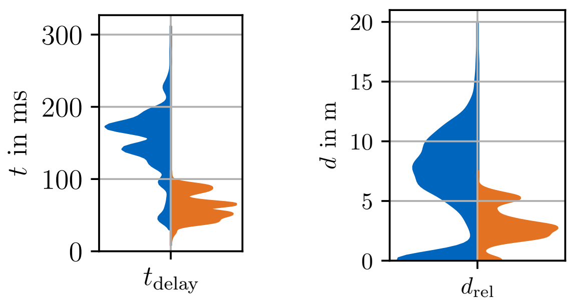
The prediction step of the EKF's kinematic model is carried out at the frequency $f_{\text{EKF}}$. By default, it is $100\,\text{Hz}$. A small deterioration in measurement residuals is observed if the frequency is doubled. This is reasoned by the deviating state forward integration because of the smaller step size $\Delta t = 1/f_{\text{EKF}}$. Specifically, the negative impact of the unmeasured yaw rate on the yaw estimation increases with higher frequency. In contrast, a filter frequency lower than $50\,\text{Hz}$ has a negative impact on measurement residuals because the linearization error increases.

The clustering to avoid multi-detections per object, and by this, the number of FPs around one TP object is parameterized with the merge distance $d_{\text{MRG}}$. The default value of the merged distance is $5.1\,\text{m}$, which is twice the width of the race car in the competition. Lower precision confirms the effect that more objects are initialized close to each other if the distance is reduced. Raising the merge distance increases precision. However, it must be considered that in close wheel-to-wheel racing of multiple objects, a high merge distance could result in the clustering of two TP objects.

Lastly, the influence of the two sensor modalities is analyzed. The LiDAR (L) algorithm accurately measures the object's position, as shown by the decrease in the standard deviation and shift toward a zero mean of the longitudinal and lateral residuals. The more precise position measurement additionally improves the tracking of the object's heading. Moreover, the LiDAR algorithm has a lower number of FPs, which improves the precision. Compared to the parameterization of the measurement uncertainty (Appendix A.1), the longitudinal and yaw-angle standard deviations are too low, while the lateral standard deviation is overestimated. The benefit of the RADAR (R) is the ability to measure the object's speed. The single modality run reveals a reduction in the velocity residual, even though the position and yaw residuals become worse, which degrades the velocity prior and negatively influences the velocity residual. The precision decreases by $36\,\%$ due to the RADAR's FP-rate. Compared to the filter parameterization (Appendix A.1), the longitudinal, lateral, and yaw standard deviations are modeled with higher uncertainty than the evaluation reveals. In contrast, the assumed measurement uncertainty of the velocity is by an order of magnitude too small when compared with real-world results.

## Delay Compensation

The assessment of the delay compensation feature is presented in Figure 3.7. The distributions of all samples, which are successfully processed by the delay compensation algorithm, are shown. The LiDAR has an average delay of $151\,\mathrm{ms}$, with a $90\,\%$-quantile of $191\,\mathrm{ms}$, while the RADAR exhibits an average delay of $62\,\mathrm{ms}$, with a $90\,\%$-quantile of $89\,\mathrm{ms}$. Taking into account the ego speed, the ego object's average traveled distance is $6.20\,\mathrm{m}$ for LiDAR and $2.98\,\mathrm{m}$ for the RADAR sensor, with distances reaching up to $19.97\,\mathrm{m}$ at maximum.



(a) Temporal distribution of compensated delay.

(b) Spatial distribution of compensated delay.

🟦 LiDAR 🟧 RADAR

Figure 3.7: Temporal distribution of delay from sensor time stamp to tracking subscription of LiDAR and RADAR and related moved object distance [280].

Considering the delay-compensated detection inputs, the associated tracking performance is implicitly reflected in the overall tracking performance since the provided real-world data includes the original sensor time stamps. Therefore, the presence of low-biased residuals and a robust tracking accuracy provides evidence of the effectiveness of the delay compensation method (Table 3.1).

## Transient Behavior

To investigate the transient behavior of the EKF, the residuals against the time elapsed since the initialization of an object's track are analyzed. Figure 3.8 outlines the residuals grouped per observation time for different sensor setups.

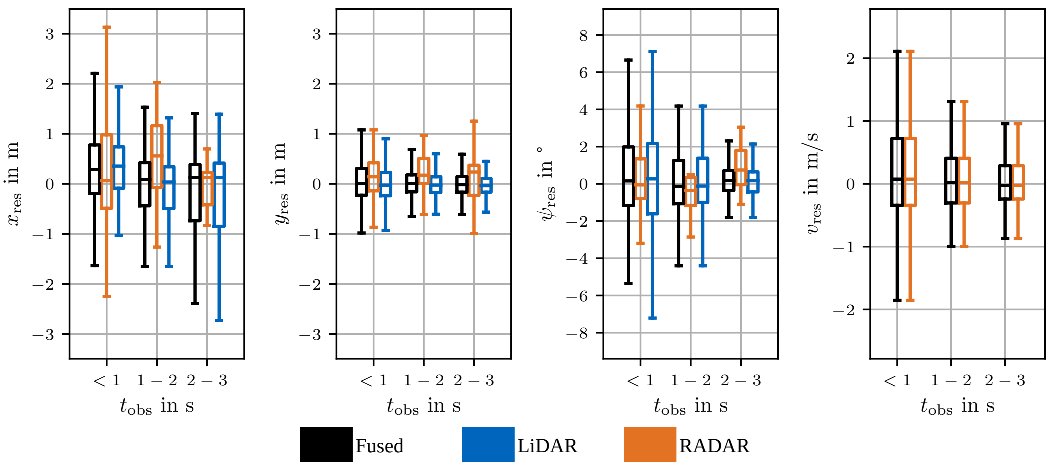


Figure 3.8: Residual error in the transition of the fused system, of the LiDAR and RADAR detection inputs [280].

In the first plot, which shows the longitudinal residuals, it can be seen that the RADAR residuals consistently improve over the observation time. In contrast, the residuals of the fused system and the LiDAR residuals only improve after the first second. Considering the tracking performance, it can be interpreted that the longitudinal position estimate has not stabilized within the specified time frame. A decisive factor for the transient behavior is the ratio of the measurement covariance compared to the process covariance. This ratio shows the trust in a new measurement compared to the kinematic process and influences the magnitude of the Kalman gain (Equation (3.9)). For the longitudinal position, the set standard deviations (Appendix A.1) of $x_{\mathrm{std,L}} = 0.3\,\mathrm{m}$ for the LiDAR and $x_{\mathrm{std,R}} = 3.0\,\mathrm{m}$ for the RADAR result in covariances of $0.09\,\mathrm{m}^2$ and $9\,\mathrm{m}^2$. In comparison, the longitudinal process uncertainty per second is $\dot{x}_{\mathrm{std,Q}} = 0.2\,\mathrm{m\,s^{-1}}$. If it is multiplied with the average cycle times of $110\,\mathrm{ms}$ for the LiDAR and $60\,\mathrm{ms}$ for the RADAR and squared, process covariances of $4.8 \cdot 10^{-4}\,\mathrm{m}^2$ for the LiDAR and $1.4 \cdot 10^{-4}\,\mathrm{m}^2$ for the RADAR result. Comparing the measurement and process covariances a difference of multiple magnitudes can be observed. Thus a low trust is put into the measurement input, which results in a slow adaption to the measurements and explains the low dynamic of the transition. The choice is, however, reasonable because it offers high stability in steady-state tracking. Thus, the trade-off is balanced toward stable tracking instead of high dynamics in transitions.

The fused residuals (black) associated with the lateral position $y_{\mathrm{res}}$ and yaw angle $\psi_{\mathrm{res}}$ experience a reduction during the initial $3\,\mathrm{s}$. Even though the balance between measurement and process covariance (Appendix A.1) is of the same magnitude as the longitudinal direction, the residuals converge faster due to the low lateral measurement error of the LiDAR. In the case of the yaw angle, it must be considered that it is not received from the detection system but derived from the orientation of the race track's centerline. Thus, the converging behavior reflects the error between the track's and the object's estimation. Without real measurements, this error is remedied by the low state uncertainties in lateral and longitudinal positions, which influence the yaw angle estimation through the kinematic dependencies in the CTRV-model (Equation (3.12)).

The last distribution shows the transient behavior of the speed estimation. Only the RADAR sensor measures this value. The distribution over the observation horizon continuously decreases, which indicates consistency between measurements and estimations and the associated covariances.

## 3.3 Physics-Constrained Deep Neural Motion Prediction

Specific requirements based on the intended usage for autonomous racing and the overall software concept are given for the prediction method [295]. A long-term prediction with a horizon of $5\,\mathrm{s}$ is requested by the downstream planning concept [278, 279] and necessary to enable interactive racing at speeds up to $300\,\mathrm{km\,h^{-1}}$. Additionally, the motion planner demands the output of a single trajectory to determine the collision cost function. To enable real-world application, a reliable prediction behavior has to be secured. Reliability comprises the robustness of the prediction accuracy against noisy input data and specific conditions for the trajectory. The first condition is the trajectory's smoothness, which is mathematically defined as a continuously differentiable curve. Early experiments revealed that oscillating trajectories, even with low amplitude around the ground truth, deteriorate the ego vehicle performance due to the sensitivity of the motion planner. The second condition is the guarantee of feasibility for the predicted trajectory. Feasibility is given if the predicted trajectory lies inside the race track and if the trajectory is driveable within the dynamic limits. Besides the reliability, the prediction method shall be able to express various maneuvers, including race-line following and interactive overtaking. Regarding the computational resources, deploying the full software stack on the hardware in the race car constrains the hardware usage and execution frequency of the method. A module frequency of $> 50\,\mathrm{Hz}$ on a single-core CPU without GPU usage is defined. Lastly, a limitation is that no public racing dataset is available for development. The state of the art is reviewed for suitable methods based on these requirements (Table 3.2).

Table 3.2: Comparison of single-trajectory prediction methods based on the defined requirements for autonomous racing. In addition, the proposed method MixNet is listed.

| | **Physics-based** | **Maneuver Prediction** | **Encoder-Decoder** | **Planning-based** | **MixNet** (proposed) |
|---|---|---|---|---|---|
| **Prediction Horizon**<br>$5\,\mathrm{s}$ time span | - | + | + | o | + |
| **Prediction Variety**<br>Variable object behavior | - | - | + | o | + |
| **Trajectory Quality**<br>Smooth, driveable, inside track | + | + | - | - | + |
| **External Constraints**<br>Low dependency on rules | + | - | + | o | + |
| **Computation Effort**<br>$50\,\mathrm{Hz}$, single-core, no GPU | + | o | o | - | + |
| **Data Requirements**<br>Low reliance on expert data | + | o | o | - | + |

Due to the expected motion complexity of the race cars and the required long-term prediction horizon, purely physics-based methods are discarded. Even though the assumptions of these methods are universal and independent of external rules, they do not hold for interactive scenarios and long-term predictions. Approaches such as reachable sets in combination with the dynamic limits of the race cars would result in large occupancy grids and conservative ego behavior. In addition, the overall software architecture demanded a single trajectory prediction instead of an occupancy grid. Common maneuver prediction approaches with clustering and classification can fulfill the reliability requirement and the output of $5\,\mathrm{s}$ trajectory predictions. However, the unstructured environment of autonomous racing does not offer the necessary external information on traffic lanes, which the algorithms rely on. Additionally, it remains uncertain if reducing the expected behavior to a discrete amount of maneuvers leads to performant racing. Encoder-decoder models are assessed to meet the requirement of long-term prediction of manifold behavior. The reliability of the real-world application is not given because their unconstrained behavior can not fulfill the conditions of smoothness and feasibility. The size of state-of-the-art models must also be limited to fulfill the computation constraints. Lastly, planning-based algorithms like game theoretic approaches exhibit a high complexity to implement a reliable interaction behavior. Further methods of learning from demonstration, such as IRL, are not applicable because of the high demands of expert data to train the model. In addition, the high computational efforts of these models exceed the requirements.

With the conclusion that none of the common architectures from the state of the art fulfills the formulated requirements, the idea is to combine an encoder-decoder model with physical constraints to ensure a diverse prediction behavior based on comprehensive scene encoding and reliability at application. The concept of the physics-constrained prediction method, the MixNet, is outlined in Figure 3.9. Inputs to the method are the object history derived from the tracking and the track map. These inputs are passed through a NN-based encoder. In contrast to common approaches, the succeeding decoder does not output the trajectory prediction but instead predicts superposition weights, which are used to derive a smooth and driveable path by combining base curves received from the track map. With this approach, sophisticated patterns can be extracted, and complex object behavior can be modeled. The drawback of NN-based encoder-decoders, its missing reliability, is compensated by the output restriction to the base curves' superposition. Through the linear combination of the curves, it is secured that the prediction stays inside the track and that the trajectory shape does not show oscillations. A plausibility check follows the base curve superposition to ensure consistency between the object's current position and the predicted trajectory. Lastly, an interaction model follows to enable collision-free trajectory prediction in interactive scenarios. For the given network, an average computation time on a single CPU core (Intel i7-4720HQ 2.6 GHz) of $9\,\mathrm{ms}$ is achieved, meeting the computational requirements. The missing data issue to develop the model is solved by data mining at the HiL-simulator and by using the data from the official simulation race of the IAC.

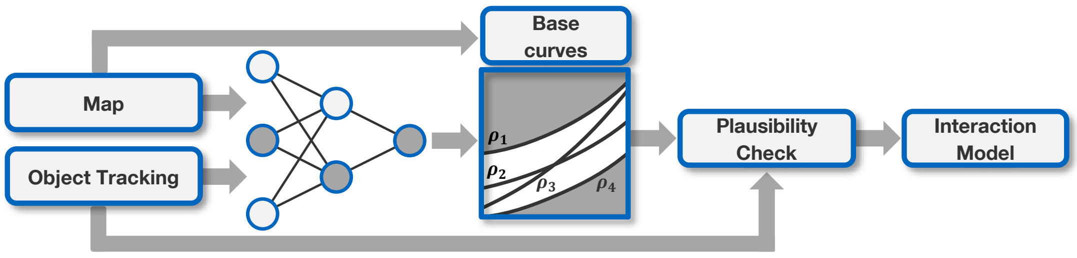


Figure 3.9: Overview of the physics-constrained prediction method, the MixNet [283]. It combines a comprehensive scene encoding utilizing a NN-based encoder and physics constraints through base curves derived from the track to improve the feasibility of the predicted trajectory. Plausibility checks and an interaction model create the final prediction output.

The prediction behavior in an exemplary scene is shown in Figure 3.10. The physics-constrained prediction approach is compared to an unconstrained encoder-decoder and a rail-based approach. The unconstrained approach is equipped with the same encoder architecture but determines the prediction through an LSTM-based decoder. The rail-based approach uses the heuristic that the object continues to move in parallel to the boundaries. To consider the fact that the objects follow the race line if no disturbance is given, the approach is extended to shift the trajectory back to the race line at the end of the prediction horizon. Besides these three predictions, the network input, which is the object's history and boundary samples, and the ground truth are visualized. The exemplary scene shows that the MixNet outputs a smooth trajectory. The predicted trajectory does not have any oscillations. At the beginning of the prediction horizon, a shift from the current object position to the outer bound can be seen because of the superposition curve's offset, which is corrected by the plausibility check. The prediction shifts to the inner side when the turn starts and is close to the ground truth in the lateral direction. In the longitudinal direction, the prediction is too short because constant velocity is assumed along the path, and the acceleration along the straight is neglected. In comparison, the unconstrained benchmark model is equally accurate at the beginning of the prediction horizon but shows a lateral offset at the end. While the Euclidean accuracy is on a similar level, the lateral oscillations at the end of the trajectory are a distinct drawback. The model can not guarantee a smooth output. The rail-based prediction shows the same level of accuracy at the end of the prediction horizon, which confirms the decision to shift the rail toward the race line. However, the approach falls short at the first half of the prediction horizon because the lateral movement to the outer side of the race track is not depicted as the rail starts at the current object position.

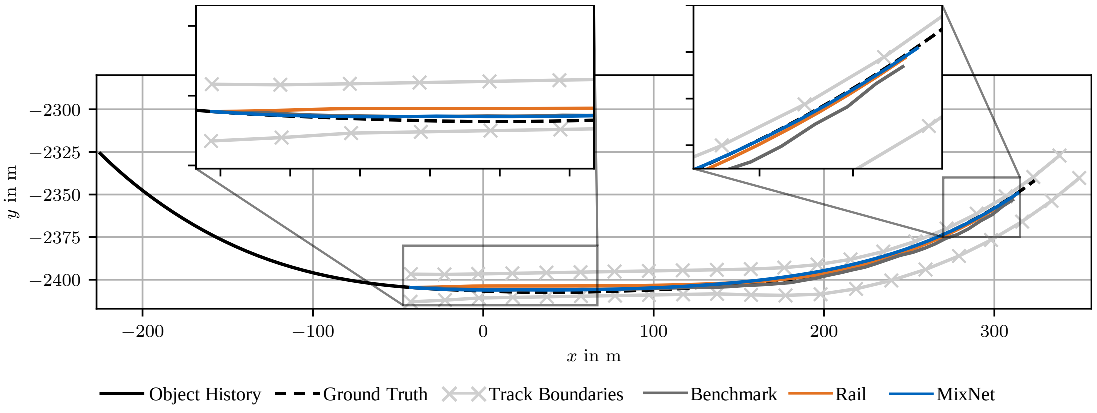


Figure 3.10: Exemplary simulated scene at the Indianapolis Motor Speedway [283]. Based on the object's history and the sampled boundaries, the MixNet predicts a trajectory without oscillations and is close to the ground truth. The unconstrained benchmark model and the rail-based prediction are shown in comparison.

The remaining section is organized as follows. In Subsection 3.3.1, the network architecture of the MixNet is introduced. Afterward, an analysis of the used base curves is conducted (Subsection 3.3.2). The second part of the method, the interaction model, is introduced in Subsection 3.3.3. Then, the model's data and training procedure (Subsection 3.3.4) and the validation regarding prediction accuracy and robustness are presented (Subsection 3.3.5). The parameterization of the method is given in the Appendix A.2. The presented method (Subsection 3.3.1 - Subsection 3.3.3) and data and training procedure (Subsection 3.3.4) are based on the author's publication [283].

### 3.3.1 Hybrid Network Architecture

The architecture of the MixNet is outlined in Figure 3.11. The LSTM layers within the encoder serve to generate a latent summary by encoding the object's history received from the tracking and the track map. The input data for the network comprises the past $3\,\mathrm{s}$ of 2D positions from $n_{\mathrm{obj}}$ objects sampled at a rate of $10\,\mathrm{Hz}$. Additionally, the left and right track boundaries from the current objects' positions onwards are fed into the network, sampled equidistantly up to a $d_{\mathrm{b,l}} = 400\,\mathrm{m}$ horizon with a sampling interval of $d_{\mathrm{b,d}} = 20\,\mathrm{m}$, which results in input dimension of $n_{\mathrm{b}} = d_{\mathrm{b,l}}/d_{\mathrm{b,d}}$. The hidden states of the encoder LSTMs are concatenated and passed through a linear layer to produce the latent representation. Then, the $n_{\mathrm{obj}}$ objects are predicted independently from each other in the decoder. In contrast to common approaches [296, 297], the decoder does not directly output future states through an LSTM network. Instead, it generates the trajectory forecast in two separate steps for path and velocity. The decoupling of path and velocity enables the fulfillment of the smoothness condition and ensures the feasibility of a curve inside the track during the path creation.

The path is constructed as a linear combination of multiple predefined base curves with weights determined by the network via linear layers and the SoftMax activation function. Four base curves are used: the two-track boundaries, a minimal curvature race line [298], and the track's centerline. To obtain the superposed curve,

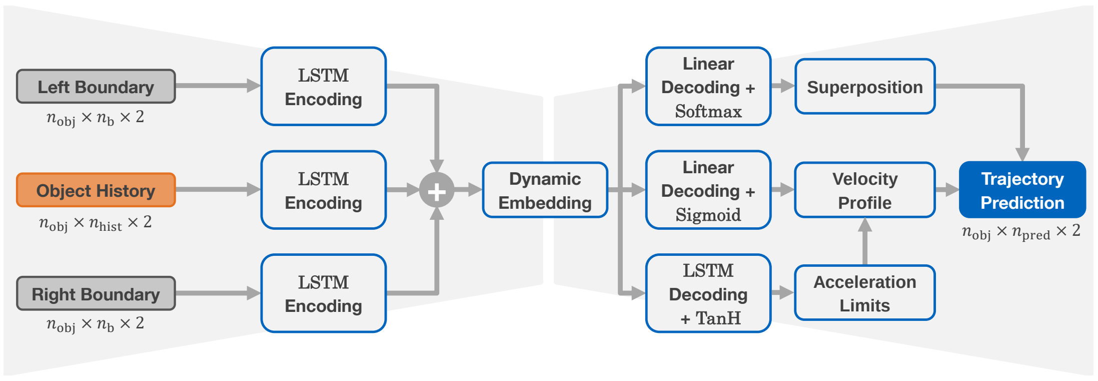

Left Boundary
$n_{\mathrm{obj}} \times n_{\mathrm{b}} \times 2$
Object History
$n_{\mathrm{obj}} \times n_{\mathrm{hist}} \times 2$
Right Boundary
$n_{\mathrm{obj}} \times n_{\mathrm{b}} \times 2$
LSTM Encoding
LSTM Encoding
LSTM Encoding
Dynamic Embedding
Linear Decoding + Softmax
Linear Decoding + Sigmoid
LSTM Decoding + TanH
Superposition
Velocity Profile
Acceleration Limits
Trajectory Prediction
$n_{\mathrm{obj}} \times n_{\mathrm{pred}} \times 2$

Figure 3.11: The architecture of the physics-constrained prediction method MixNet [283]. The prediction is divided into a path prediction by superposition of base curves and the prediction of initial velocity and constant acceleration values to get a piece-wise linear velocity profile. Inputs to the network are $n_{\mathrm{hist}}$ past 2D-positions of $n_{\mathrm{obj}}$ objects and the related left and right track boundaries in driving direction, sampled in $n_{\mathrm{b}}$ steps. Output is the trajectory prediction of $n_{\mathrm{obj}}$ objects in 2D with a horizon of $n_{\mathrm{pred}}$ steps.

the following linear combination is applied:

$$\rho_{\mathrm{sup}}(s) = \sum_{c \in \mathcal{C}} \lambda_c \rho_c(s) \quad \text{with } \lambda_c \in [0,1] \;\; \forall c \in \mathcal{C} \text{ and } \sum_{c \in \mathcal{C}} \lambda_c = 1 \tag{3.14}$$

The equation states that the superposed curve $\rho_{\mathrm{sup}}$ along the arc length $s$ of the track is constructed out of the base curves $\rho_c$ by weighting them with their corresponding weight $\lambda_c$. The given constraint that the weights of all base curves in the set $\mathcal{C}$ are between $0$ and $1$ and have to sum up to $1$ ensures that the superposed curve lies within the left and right boundaries. The constraint is enforced by applying the SoftMax activation to the output of the linear layer.

The velocity profile is piecewise linear and is determined by predicting an initial velocity and five constant acceleration values, one for each second of the prediction horizon. The initial velocity is calculated using linear decoding and the Sigmoid output activation. The five acceleration values are determined by LSTM decoding with the TanH output activation. The output of the activation function is multiplied by the acceleration limits to obtain the acceleration values. Since $-1 < \mathrm{TanH} < 1$, it is ensured that the limits are not violated.

The final trajectory prediction is obtained by resampling the path according to the velocity profile. Depending on the expected object behavior, replacing the velocity profile prediction with a constant velocity assumption is possible. Alternatively, only the initial value can be replaced, while the LSTM decoder still determines the acceleration profile. The constant velocity assumption is especially useful for oval race tracks because of the high ratio of high-speed sections.

Since the superposition weights are constant along the prediction horizon and the model's objective of minimizing the prediction error over the entire prediction horizon, it is not ensured that the predicted trajectory originates from the object's current position (Figure 3.12). A lateral deviation between the object's current position and the superposition curve can occur. To avoid this, a plausibility check is applied to the prediction output (Figure 3.9). It checks the first steps of the predicted trajectory regarding their lateral deviation compared to the current object's position. If it exceeds a predefined threshold, a lateral shift function is used to transition from the current position to the superposition path. The shift function applies a linear increasing weight factor to the lateral displacement between the current ego position and the predicted trajectory. By this, the lateral displacement is bridged at the beginning of the prediction horizon, and continuity is ensured. While the smoothness of the trajectory for the further horizon is not affected, a kink can occur at the end of the lateral shift. Instead of using constant weights and the shift function, multiple superposition weights along the prediction horizon can be determined to avoid the initial offset [299]. This approach determines piecewise constant path segments for each second of the prediction horizon and creates the path prediction by blending the segments. As the reported evaluation reveals, the prediction accuracy is improved in addition to avoiding the lateral offset. However, the approach is less robust because oscillations between path sections occur.

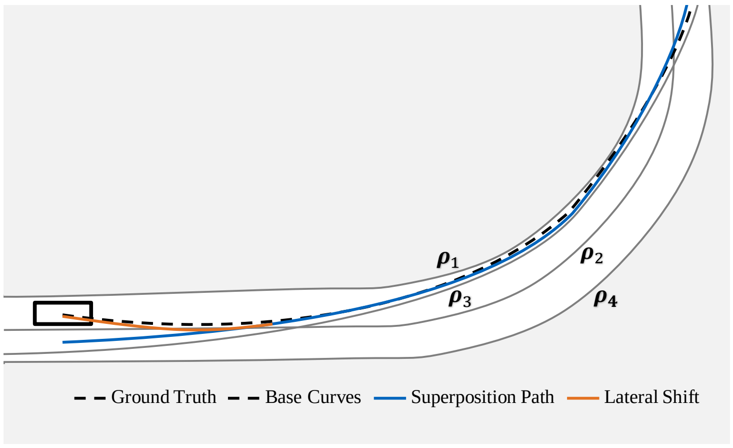


Figure 3.12: Lateral gap at the beginning of the superposition path resulting from constant weights ($\lambda_1 = 0.2,\ \lambda_2 = 0.0,\ \lambda_3 = 0.8,\ \lambda_4 = 0.0$) to superpose the base curves $\rho_i$. Without the plausibility check, a lateral gap between the object's position and the superposition path occurs. With the lateral shift function, the gap is closed.

## 3.3.2 Base Curve Analysis

Per default, four base curves are specified and superposed (Equation (3.14)). To validate the choice of the base curves, their usage by the algorithm is investigated, and an ablation study is conducted. For the investigation, synthetic trajectories are generated by superposing the four base curves by random weights. From these synthetic trajectories, the input object history and the ground truth to be predicted are derived. Since the ground truth weighting of the synthetic trajectories is known, the experiment shall be used to evaluate the usage of the base curves by the MixNet compared to the usage for their creation. The graphical analysis of this relation is given in Figure 3.13.

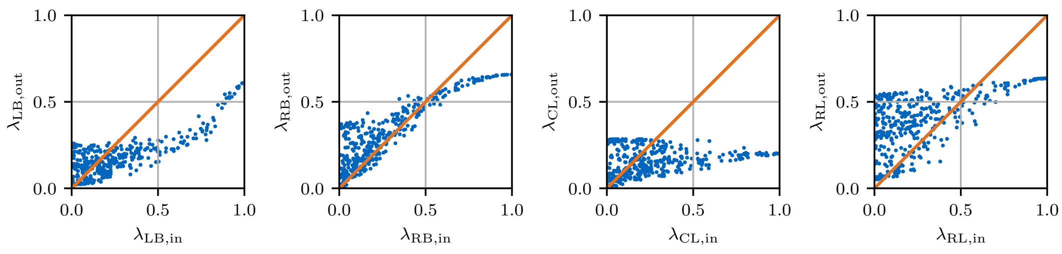


Figure 3.13: The relation between the input superposition weights of the synthetically generated trajectories $\lambda_{i,\mathrm{in}}$ and the outputted weights of the MixNet $\lambda_{i,\mathrm{out}}$ for the four base curves of the left boundary $\mathrm{LB}$, the right boundary $\mathrm{RB}$, the center line $\mathrm{CL}$ and the race line $\mathrm{RL}$.

If the MixNet would create its prediction with the same weights, a match between input weights $\lambda_{c,\mathrm{in}}$ to $\lambda_{c,\mathrm{out}}$ for all trajectories (blue dots) would results. In this case, all trajectories would lie on the bisectrix of both axes (orange line). When analyzing the actual input-output distribution, varying usage of the base curves can be observed. An overuse of the base curves for low input weights ($\lambda_{i,\mathrm{in}} < 0.5$) can be observed, which refers to samples with weights determined by the network higher than those of the input. The overuse vanishes for all base curves with increasing input weights. If the usage proportions of the four base curves are compared, it can be stated that the left boundary ($\mathrm{LB}$) and the center line ($\mathrm{CL}$) are used in low ratios by the network.

The low usage of the left boundary can be explained by the fact that the race track runs counter-clockwise. Thus the race line is close to the inner boundary in turns and can replace the left boundary in these segments. However, while the race line is close to the inner bound in the apex of a turn, and a replacement is reasonable, the replacement can result in a lateral deviation at the exit of the turn (Figure 3.14). In the exemplary scene, the race line is used instead of the left boundary to approximate the object's motion in the turn. However, at the turn's exit, the race line's usage, which drifts to the outer bound for minimal curvature, leads to a lateral deviation compared to the ground truth, which remains at the inner bound. Considering the current object speed could prevent this effect since at low speeds, a shorter path is optimal because higher curvature is dynamically possible.

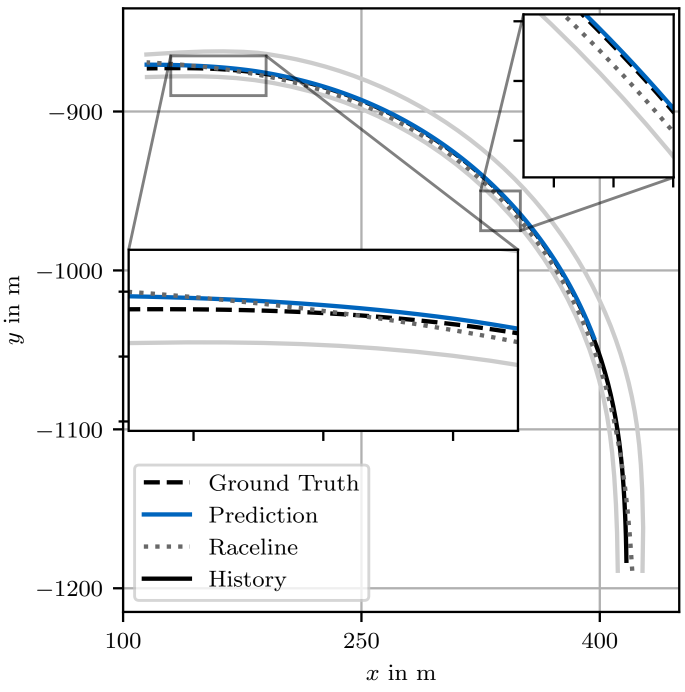


Figure 3.14: Exemplary scene with overuse of the race line in a turn, which leads to lateral deviation at the exit of the turn. The driving direction is counter-clockwise.

The low usage of the centerline can be explained by the approximation of the centerline through the two boundaries:

$$\rho_{\mathrm{CL}}(s) \approx 0.5\,\rho_{\mathrm{LB}}(s) + 0.5\,\rho_{\mathrm{RB}}(s) \tag{3.15}$$

If we assume this approximation, the weights corresponding to the centerline can be replaced by the two boundary weights, and the following redistribution can be made:

$$\lambda_{\mathrm{LB}} = \lambda_{\mathrm{LB}} + 0.5\,\lambda_{\mathrm{CL}} \tag{3.16}$$

$$\lambda_{\mathrm{RB}} = \lambda_{\mathrm{RB}} + 0.5\,\lambda_{\mathrm{CL}} \tag{3.17}$$

If the approximation and redistribution are applied to the synthetic trajectories, the left and right boundary weights, depicted in Figure 3.15, result. The usage of the left boundary increases but stays within the range $0.2 \leq \lambda_{\mathrm{LB,out}} \leq 0.3$ for inputs weights $\lambda_{\mathrm{LB,in}} < 0.7$. The right boundary weights show a proportional ratio between input and output weight for $\lambda_{\mathrm{LB,in}} < 0.75$. It can be interpreted that the right, outer boundary is not replaceable because, especially in turns, it is the only base curve that moves the predicted path away from the inner bound. Through this redistribution, the Euclidean prediction accuracy deteriorates by $2.52\,\%$ for the synthetic trajectories samples. Thus, the assumption of Equation (3.15) can be assessed as valid for this experiment. However, further investigation, especially in interactive scenarios, is required to validate the assumption to replace the centerline entirely. In summary, the analysis reveals that the input-output correlation does not show proportional behavior but all curves are used in considerable ratios, which justifies their choice for the target ODD.

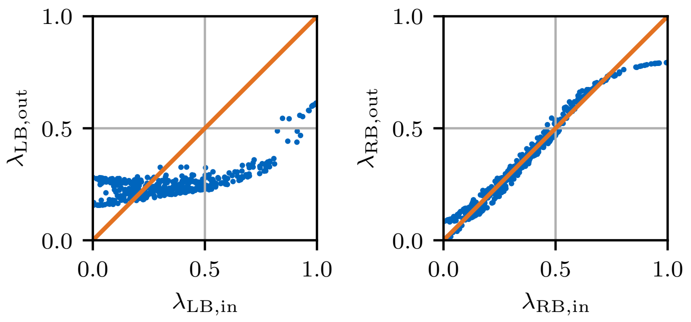


Figure 3.15: The relation between the input weights of the synthetically generated trajectories and the output weights of the MixNet if the weights of the centerline are redistributed to the left and right boundaries.

## 3.3.3 Interaction Model

The network does not consider any information about surrounding objects when predicting the target object. Even though all objects in a scene are processed in parallel, they are predicted separately (Figure 3.11). This design decision ensures reliable prediction of the surrounding objects in the cases of race line following. However, explicit consideration of interaction is required in overtaking or getting overtaken maneuvers. Furthermore, through an interaction model, it shall be ensured that the predictions of multiple objects are collision-free to enhance the plausibility.

The approach for autonomous racing of game theoretic modelling [300] has been discarded because of the high computing time of the iterative execution and the required amount of data with interactive scenarios. Instead, a two-step approach is proposed to incorporate interaction awareness. At first, each vehicle's trajectory, including the ego vehicle's, is predicted by the MixNet without considering surrounding objects. Then, these independently predicted trajectories are checked for collisions based on the vehicles' geometry and the minimal interaction distance. If the interaction distance is ensured, no adjustment of the trajectories is made because no relevant interaction between the objects is assumed. Otherwise, if the interaction distance between the predicted trajectories is violated, the trajectories are adjusted through fuzzy logic.

The fuzzy logic is based on the rules of the competition [286]. The rules state that the leading vehicle must 'hold its race line.' So, the trailing vehicle's trajectory is adjusted respectively. The logic determines the highest probable maneuver out of 'overtaking left,' 'overtaking right,' and 'staying behind.' The probabilities are calculated based on the position along the track, the relative position between the two vehicles, and the speed of both vehicles. If an overtake maneuver is the most probable, the predicted trajectory is adjusted along its lateral positions to the respective side. In the case of the maneuver 'staying behind,' the longitudinal positions are adjusted to predict a decelerating behavior. The procedure is continued for all the following cars in a hierarchical manner. Since this hierarchical procedure does not guarantee collision-free predictions for all vehicles in the end, the process can be repeated iteratively. Collisions with the ego vehicle's trajectory are always prioritized and solved last. In the given setting, the iterations are limited to two, which has been shown to solve most of the remaining collisions and does not violate the constraint of the calculation time.

### 3.3.4 Data and Training

The data mining and subsequent training of the proposed model are presented in the following. With the absence of public racing datasets and the complexity of the racing ODD, data mining is a crucial part of the development. Based on the data mining, the training process is described, including the definition of the loss function to balance the different targets of the training process against each other.

#### Data Mining

The training data is obtained by emulating the software in the multi-agent simulation on the HiL-simulator with varying parameters. In addition, data from the official simulation race of the IAC are used. Data from this race have the advantage of containing trajectories from other AV software stacks. The recorded data comprise the tracked object lists, which are used to reconstruct the opposing vehicles' trajectories. These are split in object history as input to the prediction algorithm and ground truth object future to determine the loss.

By utilizing the tracked object lists to construct the dataset, the input distribution of tracked objects and tracking quality is aligned with the expected distribution during the actual race. In addition, the synthetic object list emulator is applied to introduce variations in perception quality, augmenting the dataset. Zero-mean Gaussian noise with different longitudinal and lateral variances is added. The variances are adjusted to the real-world detection performance, which was determined during early perception-only test rides. The required track map to complete the input to the prediction model is derived from an offline-generated HD map. In total, $226$ virtual races are mined. On average, $3.4$ vehicles compete per race. A dataset of $358\,025$ training samples is generated from these races. For the validation of the method (Subsection 3.3.5), $10$ races on the HiL-simulator are recorded, following the same procedure but using a different parameter set. The whole dataset is available open source [285].

#### Training

The loss function incorporates two terms. A path loss penalizes the superposition of the base curves, which is related to the error in the lateral direction. In addition, a velocity loss quantifies the error of the velocity profile, which reflects accuracy in the longitudinal direction. The path loss $L_{\text{path}}$ between the lateral deviation of the superposed curve $\boldsymbol{x}_{\text{pred}}$ and the ground truth $\boldsymbol{x}_{\text{GT}}$, is defined as:

$$L_{\text{path}} = L_{\text{w}} + L_{\text{mse}} \tag{3.18}$$

$$L_{\text{w}} = \frac{\sum_{m=1}^{n_{\text{w}}} \|\boldsymbol{x}_{m,\text{pred}} - \boldsymbol{x}_{m,\text{GT}}\|_2^2 \left(1 + w_{\text{w}}\left(1 - \frac{m}{n_{\text{w}}}\right)\right)}{n_{\text{w}}} \tag{3.19}$$

$$L_{\text{mse}} = \frac{\sum_{m=n_{\text{w}}+1}^{n_{\text{pred}}} \|\boldsymbol{x}_{m,\text{pred}} - \boldsymbol{x}_{m,\text{GT}}\|_2^2}{n_{\text{pred}} - n_{\text{w}}} \tag{3.20}$$

$$\text{with } w_{\text{w}} = 0.5, n_{\text{w}} = 10, n_{\text{pred}} = 50 \tag{3.21}$$

To consider the higher relevance of the first prediction steps, the weighted Mean Squared Error is used for the first $n_\mathrm{w}$ time steps of the prediction horizon (Equation (3.19)). In addition, the weighted loss is beneficial to reduce the lateral offset at the beginning of the prediction horizon. The weight decreases linearly from an initial weight of $1+w_\mathrm{w}=1.5$ to $1.0$ along the weighting horizon $n_\mathrm{w}$. The loss along the remaining prediction horizon is determined by the Mean Squared Error (MSE) (Equation (3.20)). The MSE is also used for the velocity loss $L_\mathrm{vel}$, which is determined between the predicted velocity $\boldsymbol{v}_\mathrm{pred}$ and the ground truth $\boldsymbol{v}_\mathrm{GT}$:

$$L_\mathrm{vel} = \frac{\sum_{m=1}^{n_\mathrm{pred}} \|v_{m,\mathrm{pred}} - v_{m,\mathrm{GT}}\|_2^2}{n_\mathrm{pred}} \tag{3.22}$$

The overall loss function is constructed out of path and velocity loss:

$$L = L_\mathrm{path} + \Delta t^2 L_\mathrm{vel} \tag{3.23}$$

The time step size $\Delta t$ is set to $0.1\,\mathrm{s}$, according to the frequency of the input objects lists $f_\mathrm{pred}=10\,\mathrm{Hz}$. By multiplying the velocity loss with the squared time step size, both terms have the same unit of $\mathrm{m}^2$. Bayesian optimization with the goal of minimizing the loss $L$ is applied to the training parameters. The parameters depicted in Table 3.3 result from the optimization.

Table 3.3: Training parameters for the physics-constrained motion prediction method.

| Parameter | Value |
|---|---|
| Learning rate $\eta$ | $5\cdot10^{-5}$ |
| Learning decay factor $\gamma$ | $0.997$ |
| Weight decay factor $\omega$ | $10^{-7}$ |
| Batch size $\lvert B\rvert$ | $128$ |

### 3.3.5 Results

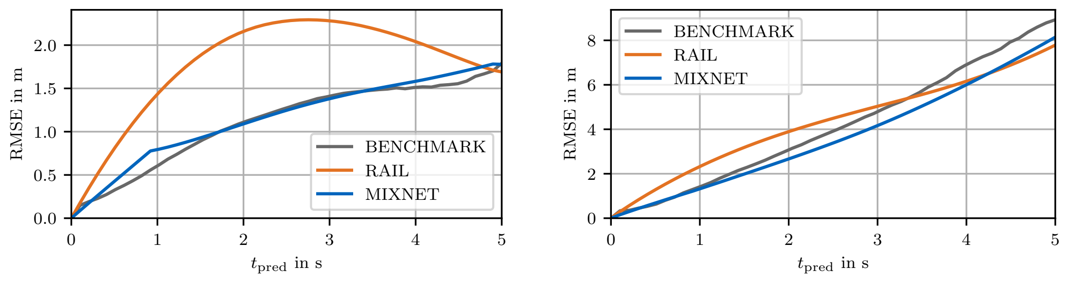
The evaluation of the MixNet comprises three parts. At first, the overall prediction accuracy is analyzed. Next, a sensitivity analysis is conducted to investigate the prediction behavior under varying conditions. The evaluation terminates with an analysis of the method's robustness against noise in the tracked object states.

The MixNet is evaluated against the unconstrained encoder-decoder architecture and the rail-based prediction with race line shift, which extrapolates the object's state in parallel to the track boundaries and shifts to the race line at the end of the prediction horizon (Figure 3.10). The MixNet is parameterized to use the initial velocity information from object tracking in combination with the piece-wise constant acceleration profile determined by the decoder.

#### Prediction Accuracy

The prediction accuracy is analyzed by the RMSE in the lateral and longitudinal directions against the prediction horizon averaged over all objects $n_\mathrm{obj}$, which is defined as:

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{l=1}^{n_\mathrm{obj}} \|\boldsymbol{x}_{l,\mathrm{pred}} - \boldsymbol{x}_{l,\mathrm{GT}}\|_2^2}{n_\mathrm{obj}}} \tag{3.24}$$



(a) RMSE in lateral direction.

(b) RMSE in longitudinal direction.

Figure 3.16: RMSE over the prediction horizon in lateral (left) and longitudinal (right) direction of the MixNet compared to the unconstrained benchmark model and the rail-based prediction.

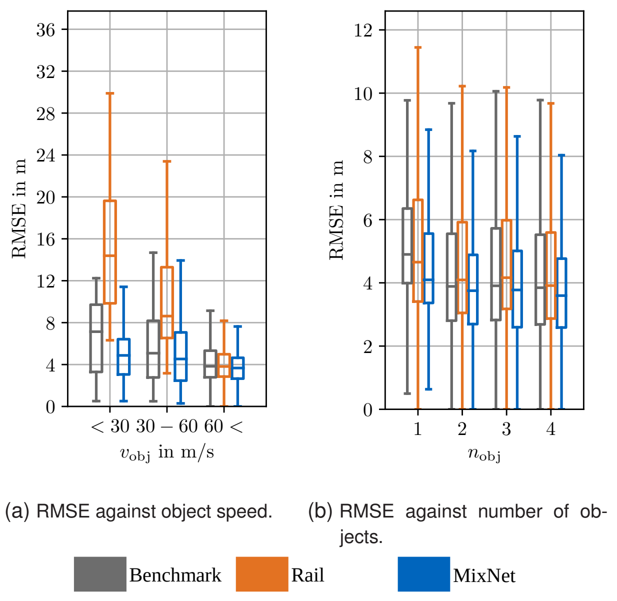
It can be split into a lateral and a longitudinal component by replacing the object state $\boldsymbol{x}$ with the longitudinal positions $x$ or the lateral positions $y$, respectively. Figure 3.16 outlines the RMSE of the MixNet, the unconstrained benchmark model, and the rail-based prediction in the lateral and longitudinal direction.

In the lateral direction, MixNet's error is on the same level as the benchmark model. This outcome supports the hypothesis that the chosen base curves adequately cover the occurring object movements. In contrast, the rail-based prediction shows higher lateral deviation. The behavior of the reducing prediction error from $2.7\,s$ on can be explained by the data distribution of test data. There is a high ratio of interactive scenarios in which the objects leave the race line, shift to a parallel line for overtaking, and then back to the race line after the successful overtake. The rail-based approach can not model this behavior because it starts from the current object's position. Thus, it can not model the shift to a parallel line during the overtaking maneuver but only the shift back to a minimal-curvature race line at the end of the maneuver. In the longitudinal direction, the MixNet shows the lowest prediction error. In this direction, the comparison with the rail-based approach is of special interest because the rail-based approach uses the constant velocity assumption in contrast to the piecewise constant acceleration profile of the MixNet. It can be seen that the consideration of the object's acceleration improves the speed prediction up to a prediction horizon of $4.2\,\mathrm{s}$.

The average overall RMSE on the recorded interactive scenarios is $4.10\,\mathrm{m}$ for the MixNet. In comparison, the benchmark model shows a prediction error of $4.48\,\mathrm{m}$, and the rail-based model with race line shift achieves a prediction accuracy of $5.06\,\mathrm{m}$.

## Sensitivity Analysis

A sensitivity analysis of the RMSE against the object speed and the number of objects per scenario is conducted. The error distribution against the object speed is depicted in Figure 3.17a. The predictions are categorized into three bins: $v < 30\,\mathrm{m\,s^{-1}}$, $30\,\mathrm{m\,s^{-1}} \leq v \leq 60\,\mathrm{m\,s^{-1}}$, and $60\,\mathrm{m\,s^{-1}} < v$, corresponding to low-speed and start scenarios, mid-speed scenarios, and top-speed scenarios. While the MixNet's median decreases with increasing speed, the mid-speed scenarios show the highest variation, which the higher dynamic options of lateral and longitudinal accelerations can explain. Thus, the transient scenarios are revealed to be challenging. A high speeds, the lowest volatility can be observed. The unconstrained benchmark model shows the highest median error and variation for low speeds. Compared to the error distribution at high speeds, which is on the same level as the MixNet, the unconstrained model shows a higher sensitivity. Thus, the physics constraints stabilize the network behavior to be less sensitive against speed variations. The rail-based model's sensitivity shows a similar behavior as the MixNet's but the error distribution is on a higher level overall. Hence, the two models have the same stability against object speed variations but the MixNet is beneficial because of its higher flexibility to create trajectories.



(a) RMSE against object speed. (b) RMSE against number of objects.

Figure 3.17: Error distribution against the object speed and the number of objects.

The second sensitivity analysis is against the number of objects per scenario (Figure 3.17b). The medians for all models range between $3.6\,\mathrm{m}$ and $4.8\,\mathrm{m}$ over the number of objects. The prediction errors are on a similar level with varying numbers of objects. This can be explained by the observation that the actual wheel-to-wheel racing with more than two vehicles, which would result in strong lateral interaction, is only for a short time span. Instead, scenarios with more than two vehicles primarily involve sequential overtaking maneuvers between two vehicles. Thus, for the majority of the scenes with $n_{\mathrm{obj}} > 1$, a similar behavior is observed as in the case of $n_{\mathrm{obj}} = 1$, and the prediction models are of similar accuracy.

## Robustness

One key indicator of a reliable prediction is robustness against noise in the tracked object states. Therefore, an additional validation with varying zero-mean Gaussian noise is conducted to assess this feature. It should be noted that the original test data already contains noise based on the measurements derived from the early real-world tests. In the following study, additional noise is added to these data (Table 3.4).

Table 3.4: Robustness of the unconstrained benchmark model, the rail-based model, and the MixNet against zero-mean Gaussian noise in the lateral and longitudinal direction.

| **Standard Deviation** | | **Benchmark Model** | | **Rail with Race Line** | | **MixNet** | |
|---|---|---|---|---|---|---|---|
| – | | RMSE | | RMSE | | RMSE | |
| $\sigma_{\mathrm{lon}}$ in m | $\sigma_{\mathrm{lat}}$ in m | abs in m | rel in % | abs in m | rel in % | abs in m | rel in % |
| 0.0 | 0.0 | 4.48 | - | 5.06 | - | **4.10** | - |
| 0.5 | 0.5 | 4.69 | +4.6 | 5.12 | **+1.2** | **4.36** | +6.5 |
| 1.0 | 1.0 | 5.43 | +21.2 | 5.29 | **+4.5** | **4.64** | +13.4 |
| 1.0 | 0.0 | 5.09 | +13.6 | 5.18 | **+2.4** | **4.41** | +4.7 |
| 0.0 | 1.0 | 4.82 | +7.5 | 5.17 | **+2.2** | **4.49** | +9.7 |

If noise is added in both directions simultaneously, the advantage of the MixNet compared to the benchmark model gets visible for higher standard deviations ($\sigma_{\mathrm{lon}} = 1.0\,\mathrm{m}$, $\sigma_{\mathrm{lat}} = 1.0\,\mathrm{m}$). In this case, the benchmark model loses prediction accuracy by $21.2\,\%$ compared to $13.4\,\%$ by MixNet. This observation indicates that the constraints applied to MixNet yield the desired model behavior of increased robustness against input variations. In comparison with the rail-based prediction with race line consideration, the MixNet has a higher relative increase of the RMSE error. The rail-based prediction deteriorates by less than 5 % through all experiments. However, the variation of the object position only affects the start point of the rail-based prediction but not the length of the prediction because the additional input variable to the rail-based method of the object's speed is not varied by noise.

Focusing on MixNet's behavior, it can be observed that the noise in the lateral direction has more influence on the prediction accuracy than the longitudinal noise, which differs from the other models. The lateral noise especially disturbs the weight selection for the base curves, which impacts the prediction accuracy. With a focus on the absolute RMSE errors, the MixNet shows the lowest RMSE over all experiments.

## 3.4 Discussion

Based on the identified research gap, the proposed methods for tracking and motion prediction focus on reliability, validated in the ODD of autonomous racing. An approach for multi-modal fusion of heterogeneous sensor pipelines in combination with a tracking method, including a novel delay compensation approach, is implemented to provide low-biased and consistent object histories as necessary input to the prediction module (Section 3.2). The reliable prediction is realized by combining data-based scene encoding with physics constraints and rule-based interaction to ensure accurate prediction while maintaining essential quality guarantees for the full-stack application (Section 3.3). In the following, the presented results in the ODD of autonomous racing (Subsection 3.4.1) and their transfer to further ODDs (Subsection 3.4.2) are discussed.

### 3.4.1 Validation of the Methods

The evaluation of the tracking method shows a low-biased tracking behavior of the EKF with a CTRV-model, indicating that the parameterization is appropriate for the given detection inputs (Table 3.1). Low lateral residuals, particularly, are crucial for the specific use case of wheel-to-wheel racing. A shortcoming is the speed estimation, which shows volatile behavior that can lead to reduced tracking accuracy in transient situations. The low precision mainly results from objects on the edge of the sensors' FoVs. Besides the CTRV-model, alternative kinematic models are analyzed. The CV-model shows decreased lateral residuals and precision. The CTRA-model is of equal performance with a slight improvement in the longitudinal direction but is prone to oscillations because of the non-measured acceleration state. The sensitivity analysis (Table 3.1) reveals at first that the precision is influenced by the three variables of the match distance, the match counter threshold, and the merge distance. The respective parameters are set to balance the trade-off to stably match new objects, to minimize the influence of multiple detections for one object, and to discard objects without detections after a specific period. A second finding of the analysis is that the filter frequency directly manipulates the kinematic state's forward propagation, degrading the precision and residuals if a lower frequency is used. A third observation is the influence of the sensor modalities, shown by the single sensor validations. They reveal that the measurements from the LiDAR sensor are essential to track the objects' positions accurately. Despite the low positional accuracy, the benefit of RADAR is the measurement of the object's speed. The investigation of the method's transient behavior reveals a low dynamic in the longitudinal direction (Figure 3.8), which results from the setting of process and measurement covariances. An optimization of the filter parameters to balance transient behavior and stable tracking could improve the transient behavior. To resolve the trade-off between high dynamics in transitions and stable tracking in steady-state scenarios, adaptive values for process and measurement covariance parameters can be implemented.

The validation of the MixNet shows an improvement in the prediction error compared to the unconstrained benchmark model and the rail-based approach while ensuring quality guarantees of a non-oscillating trajectory (Figure 3.16). The results demonstrate that the constraints do not negatively influence the network's capabilities to model the motion behavior in the given ODD. A distinct aspect for future research is revealed by the conducted analysis of the base curves selection and usage by the network (Subsection 3.3.2). The qualitative study on the replacement of the centerline by the two boundaries suggests possible optimization for the selection of the used curves. An inherent shortcoming of the method is the occurrence of lateral gaps at the beginning of the prediction horizon because of the constant superposition. While this is remedied by means of a correction shift, an integral solution by the algorithm itself should be intended. The idea to use multiple superposition parameters along the prediction horizon [299] shows improved accuracy but loses reliability. These two factors must be balanced against each other when optimizing the base curve composition. The sensitivity analysis of the MixNet reveals a low bias against the number of objects (Figure 3.17). The bias toward the object's speed is assumed to be influenced by the speed distribution of training data as there is a high ratio of high-speed scenarios in the data. In addition, the reduced accuracy in the mid-speed range indicates that the model falls short in scenarios with an increased dynamic potential of the objects. The sensitivity against noisy input data influences the prediction accuracy, especially in the lateral direction (Table 3.4). However, for the intended real-world application, it is relevant to consider the occurring noise during inference and, based on this, estimate its actual influence. Under the consideration of the noise observed in applying the tracking method on real-world data (Table 3.1), the model can be assessed as reliably applicable. Regarding the absolute prediction error, the MixNet outperforms unconstrained and rail-based models when noise is added.

### 3.4.2 Transferability to new Operational Design Domains

While autonomous racing is suitable to validate algorithms at high speeds in real-world scenarios, the question remains of how to transfer the developed methods for the usage in further ODDs. When the proposed methods for autonomous racing are analyzed, the following conclusions can be drawn.

The multi-modal late fusion and tracking method's low-biased residuals are assessed to meet the requirements for ODDs of public road traffic. The residuals are expected to decrease with lower object speeds and distances. A crucial element is the used kinematic model in the EKF. While the filter method is evaluated to be adaptive with a low effort of adjusting the parametrization for new ODDs, the usage of the CTRV as the kinematic model involves limitations. At first, the heading estimation is a crucial aspect. It has to be ensured that the detection provides the object's orientation. The assumption taken for the autonomous racing ODD that all objects drive in the same direction does not generally hold. Especially when it comes to different types of objects beyond motorized vehicles, estimating the heading based on the road map information is a safety risk. In addition, the sensors must measure the velocity of the objects to use the model. Since the yaw rate is a non-measured state, it must be validated if oscillations in this state occur. If this is the case, simpler models, such as the CV-model, should be considered. Furthermore, using the CTRV-model to track other object types, such as pedestrians and cyclists, must be validated because their motion behavior differs from vehicles. Besides the state estimation, input processing and matching assumptions made for the racing ODD have to be revised critically regarding the adaptivity of the method. The out-of-track filter of FPs might be usable for vehicles in public. However, due to errors or changes in the underlying map, which specifies the driveable area, this implies a safety risk. For pedestrians, the out-of-track filter is not useable. Besides that, the parameterization to merge two detections has to be reviewed. In dense traffic, FNs could occur if two close objects are merged into one. The same applies to the match distance, which could result in the wrong association of neighboring objects to one UID. The consideration of further features, such as the object size, is recommended to match tracked and detected objects reliably. The introduced feature of the delay compensation is considered to be adaptive to new ODDs. It depends on using a suitable kinematic model but is generalizable regarding the backward-forward integration. An advantage of the method is that it does not need labeled data to be deployed in new scenarios, which lowers the effort of integration and optimization.

The transfer of the MixNet to the ODD of public road traffic implies the following challenges. The first issue is the selection of suitable base curves. Due to the fact that typical rides are one-way from one location to another, the base curves have to be updated continuously. Exact map information must be given. Also, the amount and shape of the curves have to be evaluated for specific situations. While the application on closed highways is similar to closed race tracks, the application of superposed curves for roundabouts or junctions must be analyzed. For these non-closed road geometries, the space between the base curves is not necessarily a driveable area, which violates one essential quality guarantee. An example could be a T-junction with two base curves for turning left and right, resulting in implausible superposed curves toward the non-driveable area (Figure 3.18). Thus, further heuristics must be used to adjust the superposition approach to open road geometries. A variation of the concept to select the most suitable base curves instead of superposing them is discussable. The idea is similar to the presented state of the art of classifying motion patterns into prototypical trajectories (Subsection 2.3.2) with the known drawback of being limited to the selected curves in the output. Lastly, the interaction model, which is implemented using fuzzy logic, must be reviewed. While the decision between 'overtaking left,' 'overtaking right,' and 'staying behind' might be suitable for highway scenarios, dense urban road traffic is assessed to require more sophisticated models.

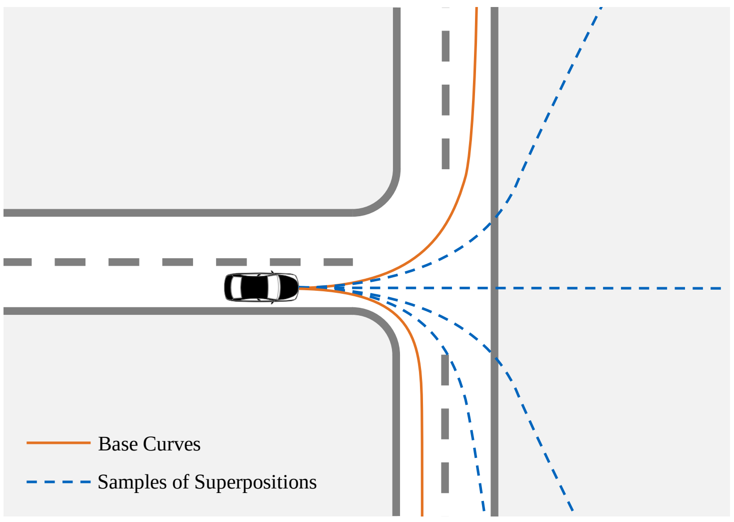


Figure 3.18: Application of the MixNet at a T-junction. Two base curves derived from the road network are input to the method. Superposing these base curves results in paths that leave the driveable area. The feasibility condition of staying inside the driveable area is violated for non-closed road geometries.

# 4 Adaptive Motion Prediction in Public Road Traffic

The second identified criterion of a generic prediction framework is adaptivity (Section 2.4). Its realization shall be investigated by the research items RI3, RI4, and RI5 in this chapter. The criterion is motivated by the human skill of adapting driving behavior through experience. Human drivers achieve this in two steps. First, if human drivers enter an unknown scenario, they try to adapt their behavior based on known patterns [301]. The known patterns can be referred to as driving experience. Human drivers can additionally assess if their skills are insufficient in specific scenarios, and their safety could be impaired [302]. The second aspect of human adaptivity is that they learn from unknown scenarios. Especially if they commit errors, they have high competence to learn from them [303]. Thus, their driving experience grows, and by this, they are able to solve future scenarios more confidently. Consequently, if the human drivers' skills of adaptivity are taken as a reference, incorporating scenario understanding into motion prediction algorithms has the following two goals. First, through scenario understanding, the prediction method shall be scene-dependently adapted, and the most suitable manner to predict an object's motion shall be used. Hence, the prediction behavior shall be adjusted in every individual scene throughout a scenario. Also, particular scenes that challenge the AV software in predicting surrounding objects and are expected to result in erroneous predictions shall be detected to prevent safety-critical situations. The second goal is to use these scenes as data samples to expand the scenario understanding and prediction abilities for future applications.

This chapter focuses on the first goal: The scene-dependent adaption of the prediction behavior and the detection of scenes that an algorithm cannot accurately predict by means of scenario understanding. To realize this, a scene-dependent self-evaluation method is proposed (Figure 4.1). For a given scene, an evaluation is conducted to select the most suitable trajectory prediction model, i. e. the most suitable predictor. If none of the predictors is assessed as suitable, the scene is classified *invalid*. By this, the prediction is adapted to the current circumstances if possible, and high prediction errors, which could lead to safety-critical scenes, are a priori detected and can be prevented.

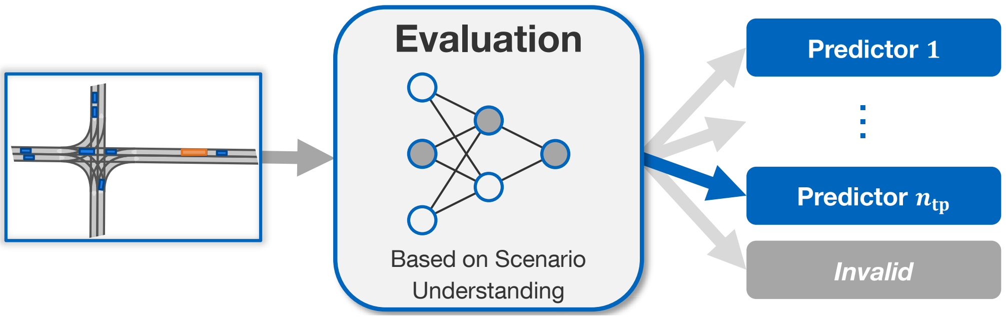


Figure 4.1: Overview of the proposed method: Self-evaluation of trajectory predictors [304]. The encoded scene comprising the objects and semantic information is input to the evaluation. The evaluation selects the best prediction method for the given scene. If none of the prediction models is expected to predict with the required accuracy, the scene is classified *invalid* to prevent high prediction errors.

The method is validated in public road traffic, which offers the following advantages to prove adaptivity. Due to the variety of road geometries and object types, diverse behavior and object movement must be encoded and predicted. In addition, objects' intentions can strongly vary among surrounding objects, and individual driving styles complicate their understanding. It is unknown which goals the objects in a scene have, which induces additional uncertainty. Based on observations, the prediction method has to estimate the intentions to derive the respect motion pattern. Furthermore, manifold interactions are expected to occur due to the varying number of traffic participants. These interactions exacerbate the prediction because concatenated effects occur between the behavior of the objects.

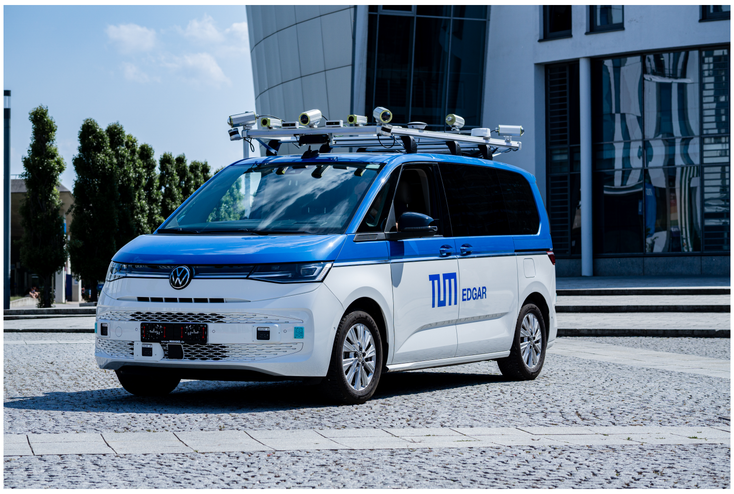
The chapter is organized as follows. At first, the environment around the research vehicle *EDGAR* is presented, in which the development and validation of the method takes place (Section 4.1). Section 4.2 introduces the hybrid prediction approach that integrates multiple predictors, which is the first part of the method. The hybrid approach combines physics-based and pattern-based prediction algorithms. The motivation for choosing these prediction algorithms and their implementation is presented in this section. The second part of the method is the self-evaluation (Section 4.3). It comprises a metric to evaluate scenes and a model to choose the respective predictor based on the metric. Next, the data and training procedures of the method are presented (Section 4.4), which build the foundation for deriving the results. The method's validation on the introduced data occurs in Section 4.5. It comprises the presentation of the quantitative results of the hybrid prediction method and the selection behavior and of qualitative results to give further insights into the method's behavior. Besides these results, two further aspects are of interest. The first aspect is the model's network architecture, which is analyzed in a comprehensive ablation study (Section 4.6). Second, influencing factors on the prediction errors are analyzed (Section 4.7). Their investigation allows insights to be used for the integration and overall software tuning process. The chapter closes by discussing the presented method and results (Section 4.8).

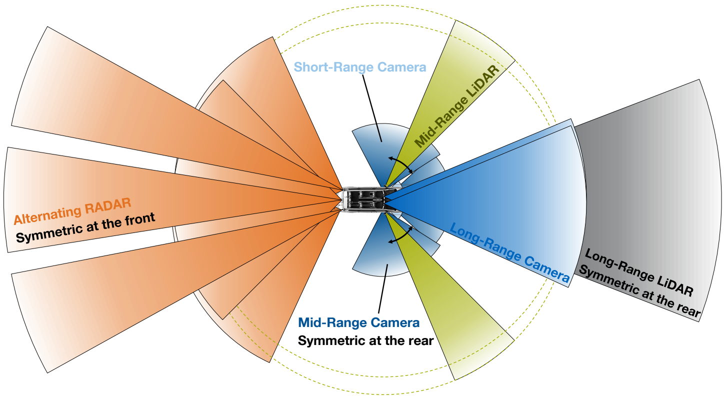
The setup of the research platform *EDGAR* is published as a preprint by the author [305]. The vehicle's digital twin is made publicly available [306]. The presented method and quantitative results are published in the author's prior work [304]. The related software is open-source available [307].

## 4.1 Research Platform EDGAR

For research in autonomous driving for public road traffic, a comprehensive development and test platform is built up [305]. The platform's core is the research vehicle *EDGAR* (*Excellent Driving GARching*, Figure 4.2a), which is embedded in a holistic simulation and data environment.

Based on a Volkswagen T7 Multivan, a multi-modal sensor setup is built with the requirements to be time-synchronized by Precision Time Protocol and ROS2 compatible. The positioning of the sensors is optimized to balance small blind spots close to the car and large FoVs at angles and distances. Figure 4.2b outlines the FoVs of the sensors in birds-eye-view. Six cameras of type Basler acA1920-50gc are used for mid-range detection and teleoperation, while two further Basler cameras to the front with a focal length of $16\,\mathrm{mm}$ are used for long-range detection with depth completion. For the short-range area, two additional stereo camera setups of type RAMOS D455E are installed, which are used for gesture detection. The LiDAR setup comprises two $360\,°$-rotating Ouster OS1-128 for localization and near-field detection and two Innovusion Falcon for long-range detection to the front and the rear. The setup is complemented by six Continental ARS430 RADAR sensors with alternating FoVs. In addition, an inertial measurement unit, multiple differential global positioning systems, and SDR-Transceivers are installed for ego-state estimation and external communication. The hardware components are listed in Appendix B.


(a) The research vehicle *EDGAR*. Image Credits: Technical University of Munich



(b) Field of View of *EDGAR*'s multi-modal sensor setup [305].

Figure 4.2: The research vehicle *EDGAR* and its sensor setup.

To enable a consistent simulation and real-world testing strategy, the vehicle's sensor setup and dynamic behavior are duplicated in a digital twin [306]. By this, the virtual validation process is aligned with the real-world tests. Despite that, the digital twin facilitates the software development process on valid vehicle and environment models.

A central element for developing and evaluating prediction and planning methods is the benchmark framework *CommonRoad* [308]. *CommonRoad*'s scenario library is used during the feature development process and for reproducible evaluation of prediction and planning methods prior to integration into the full software stack. With a mix of real-world and synthetic scenarios from different sources, it offers a wide range of road geometries and object behaviors to benchmark the developed methods. The scenario library of *CommonRoad* is also used to validate the proposed self-evaluation method.

## 4.2 Hybrid Trajectory Prediction

This section is based on the author's publication [304] and introduces the hybrid prediction approach. Since the accuracy of an individual prediction model depends on the object type and traffic scene [253], it is assumed that an ensemble of prediction approaches from different classes with heterogeneous prediction behavior complement each other to improve the total prediction accuracy. While pattern-based methods dominate the prediction competitions on vehicle-focussed datasets [15, 81, 137], physics-based models are useful to model steady-state behavior and show the best accuracy for pedestrian prediction [253, 309]. Therefore, a combination of these two classes is proposed. Due to the superior performance of GNN-based methods, a second pattern-based model is constructed with a Graph Convolution Network (GCN). This specific type of GNN-network is chosen because of the discussed advantages of its computational efficiency in aggregating features from a node's neighbors in a single step, which is beneficial in dense graphs where each node has many neighbors (Subsection 2.3.2). An inherent bottleneck of ensemble methods is the computational effort required to run multiple models. To counteract this, an integrative architecture of the proposed prediction models, which reuses particular branches of the architecture in multiple models, is implemented (Figure 4.3). It comprises a scene image encoding and the three trajectory prediction models, the predictors ($n_{\text{tp}} = 3$):

- Constant Velocity (CV)
- Linear LSTM (L_LSTM)
- Proximity-Dependent Graph-LSTM (DG_LSTM)

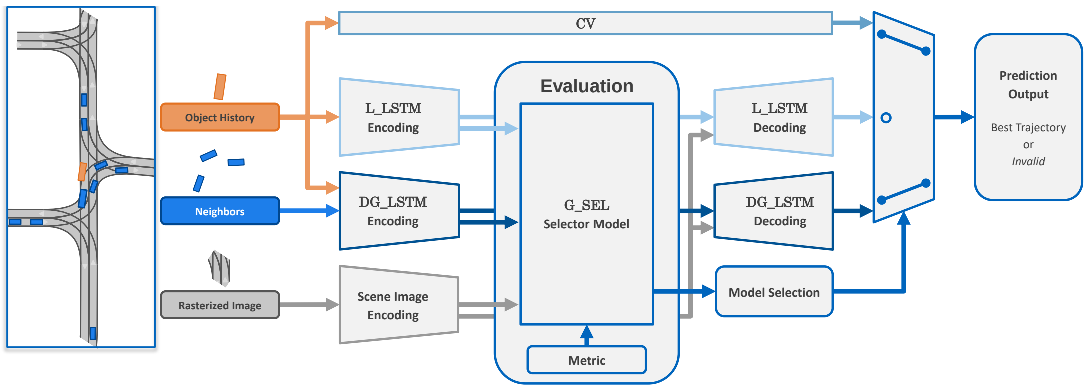


Figure 4.3: The network architecture of the self-evaluation method for trajectory predictors comprises a scene image encoding, three trajectory predictors (CV, L_LSTM, DG_LSTM), and the selector model [304]. Inputs are the target and surrounding objects' history and a rasterized image of the road network. All information is encoded and input into the evaluation. As a result of the evaluation, the selector model (G_SEL) chooses the best trajectory predictor for the present scene based on a specified metric, which is then executed to output a trajectory prediction. In the case that none of the predictors achieves the required accuracy, the prediction scene is classified *invalid* to avoid erroneous predictions.

The inputs to the method are a rasterized scene image of the road geometry and the objects' history, which are the tracked object states $(x, y, \psi)$ of the target and the surrounding objects. The time span of the object history is set to $3\,\text{s}$. Besides the objects' history, the object class $c_{\text{o}}$ is added to the input vector, which results in an input feature size of $d_{\text{in}} = 4$. The object states are transformed into the local coordinate system of the target object's current pose $(x_0, y_0, \psi_0)$. The transformation into the local coordinate system of the target object's pose yields a normalized input for the prediction models. Based on these inputs, the models' outputs are the predicted trajectory of the target object $\boldsymbol{x}_{\text{pred}}$ with its $x$-, $y$-positions over the prediction horizon of $5\,\text{s}$. Thus, the output feature size $d_{\text{out}}$ is 2. Object history and future are sampled with $10\,\text{Hz}$, which results in an observation length of $n_{\text{hist}} = 30$ and a prediction length $n_{\text{pred}} = 50$. In the following, the scene image encoding is introduced at first because it is used in multiple predictors and for the self-evaluation (Figure 4.3). Afterward, the three individual predictors of CV, L_LSTM and DG_LSTM are described. The models' parameter optimization is described in Subsection 4.4.2. The parameter settings are analyzed in the ablation study (Section 4.6).

### 4.2.1 Scene Image Encoding

A scene image encoding based on [163, 164], which extracts road geometry information from a rasterized image, is implemented to enhance semantic understanding. The input pixel image visualizes driveable areas and center lines of the road network's lanes in dedicated colors. The advantage of this representation is the independence from the road geometry and the number of roads because the input size remains unchanged compared to vector representations. For each object to be predicted, the scene image is cut in a square around the object's current position with a size of $d_{\text{map}} \times d_{\text{map}}$. The extracted pixel image is processed through CNN-layers. At first, $l_{\text{sc,in}}$ filters are applied to encode the semantic information. Then, the number of filters changes to $l_{\text{sc,out}}$ to create the output vector. A max-pooling operator is applied after every CNN-layer as an activation function. A latent 1D-vector results, which is used in the different decoders of the predictors and the selector model (Figure 4.3).

### 4.2.2 Constant Velocity Model

A CV-model is implemented with the target to cover steady-state object behavior. It extrapolates the object's state with the assumption of constant speed and constant heading (Figure 2.4). Because of the transformation into the local coordinate system, the CV-prediction simplifies to:

$$x_m = v_0\, t_m \quad \text{with } m \in \mathbb{Z} \mid 0 < m \leq n_{\text{pred}} \tag{4.1}$$

The equation reads that the $m$-th longitudinal prediction step $x_m$ is a 1D-extrapolation of the current object speed $v_0$ up to the prediction length $n_{\text{pred}}$. All lateral values in the local coordinate system $y_i$ are zero. The advantages of the CV-model are that it does not require any labeled data, no training, and has a low computational effort during inference. Its drawback is sensitivity to noisy input data because only the current object state is used to determine the future motion by linear extrapolation. Especially heading oscillations influence the lateral prediction accuracy. Figure 4.4 depicts this issue in a steady-state driving scene with a heading deviation of $\Delta\psi = 2.58\,°$, which corresponds to heading residual of the presented tracking method (Table 3.1). Therefore, the CV-model depends more on the tracking's ability to reduce input noise than pattern-based models.

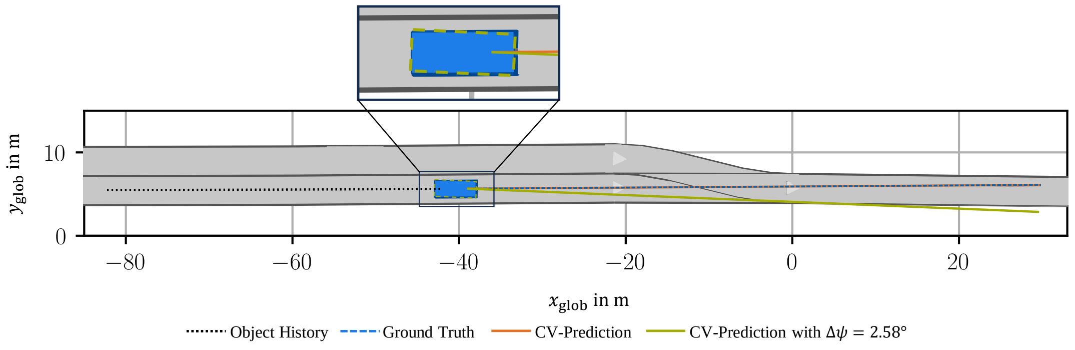


Figure 4.4: Scene with a heading error of $\Delta\psi = 2.58\,°$ at an object speed of $13.74\,\text{m}\,\text{s}^{-1}$. An increase in the RMSE of $1.91\,\text{m}$ and in the lateral deviation of $3.25\,\text{m}$ at the end of the trajectory result for the noisy prediction compared to the original prediction.

### 4.2.3 Linear LSTM Model

An RNN-based encoder-decoder with linear embedding is implemented to model low interactive scenarios. Since it does not consider the surrounding objects as input, it bases its prediction only on the encoded road geometry and object history (Figure 4.3). The encoder of the L_LSTM-model is constructed out of a linear embedding and LSTM-layers. The embedding transforms the input features into the embedding vector by means of $l_{\text{li}}$ hidden linear dense layers of size $h_{\text{li}}$ in combination with the ReLU activation function and an output layer of size $h_{\text{li,emb}}$. The embedding vector is processed sequentially by the LSTM, which contains $l_{\text{enc}}$ layers of size $h_{\text{enc}}$. The hidden state of the final LSTM layer represents the model's latent space. This latent space is embedded by a linear dense layer (size $l_{\text{lat}}$) and, concatenated with the scene image encoding, input to the LSTM-decoder. The one-layered decoder enrolls the latent space into a temporal vector through an LSTM, followed by a linear layer, which transforms the LSTM's output of size $h_{\text{dec}}$ into the output feature size. The LSTM-decoder is an adjusted version of the decoder used in the physics-constrained prediction method to obtain the acceleration profile (Section 3.3). Within the `L_LSTM`-model, the architecture's integrative characteristic takes place by reusing the model's latent space in the selector model (Figure 4.3).

### 4.2.4 Proximity-Dependent Graph-LSTM Model

The implementation of the `DG_LSTM` [310] is made to cover interactive scenarios. To achieve this, the model considers all objects in the scene as input, which are processed into an undirected graph $\mathcal{G} = \{\mathcal{N}, \mathcal{E}\}$. Each object is represented by a node $v \in \mathcal{N}$ with a node feature vector $z$ consisting of the object's historical positions $(x, y)$, heading $\psi$, and class $c_{\text{o}}$. Between the nodes, the edges $\epsilon \in \mathcal{E}$ are established. To improve the graph's focus on relevant interactions, only nodes with proximity below a threshold $\delta$ throughout the sampled history and future are connected. The distance-based limitation of the graph connections has been shown to improve the prediction behavior [156, 311]. An example of the proximity-dependent graph representation is shown in Figure 4.5.

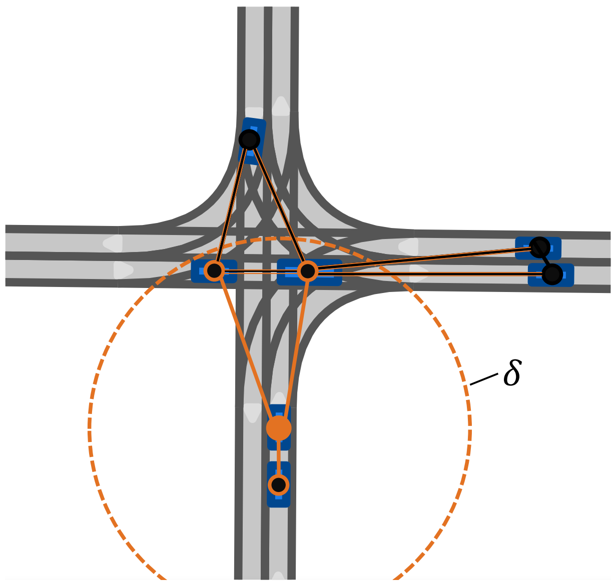


Figure 4.5: Schematic depiction of the proximity-dependent graph representation for a traffic scene [304]. Only objects with proximity below the threshold $\delta$ (orange edges) are connected and considered in the one-layered GCN of the target object (orange dot). Alternatively, by means of a two-layered GCN, second-order interactions are considered (orange-black edges).

The target object (orange dot) has direct connections to the three objects inside the proximity (orange edges), which are considered in the GCN-embedding if one layer is used. If the number of GCN-layers is increased to a value of $2$, second-order connections (orange-black edges) are additionally considered. In the current implementation, it is set to $l_{\text{dg,emb}} = 1$. The constructed graph is processed by GCN-embedding. It contains $l_{\text{dg}}$ layers of GCNs of hidden size $h_{\text{dg}}$ and output size $h_{\text{dg,emb}}$. The update of the node feature vector $z$ of an object $i$ at time $t$ is formulated as follows:

$$z'_{i,t+1} = \chi\left(\varphi_i\left(z_{i,t}\right) + \Gamma_{j\in\mathcal{N}|j\neq i}\ \varphi_j\left(z^i_{j,t}\right)\right) \tag{4.2}$$

The update of the node feature vector comprises three steps:

1. **Transformation**:
   All connected node feature vectors of the surrounding objects are transformed into the target object's coordinate system. The modified feature vectors $z^i_{j,t}$ are obtained.
2. **Message Passing**:
   The modified node feature vectors $z^i_{j,t}$ are processed by the common message function $\varphi_j$. The node feature vector of the target object $z_i$ is processed by the message function $\varphi_i$. A linear-dense layer of size $m_{\text{dg}}$ with ReLU activation defines the two message functions.
3. **Aggregation**:
   The messages from all surrounding objects are aggregated via the aggregation function $\Gamma$. This aggregation is added to the embedded node feature vector. The sum is passed through the update function $\chi$, which is a linear dense layer with the output size $h_{\text{dg,emb}}$.

Out of these three steps, the update node feature vector $z'_{i,t+1}$ results. After the GCN-embedding, an LSTM-based encoder-decoder, which has the same architecture as used in the `L_LSTM`, follows. To limit the number of parameters of the model, the respective values of the encoder ($l_{\text{enc}}$, $h_{\text{enc}}$) and the decoder ($l_{\text{lat}}$, $h_{\text{dec}}$) are the same as for the `L_LSTM`. The scene image encoding is fused in the latent space to consider semantic information. By this, non-Euclidean interaction awareness and geometric map knowledge are combined. The `DG_LSTM`-encoding is additionally used in the selector model for the self-evaluation.

## 4.3 Self-Evaluation

This section is based on the author's publication [304]. It introduces the a priori self-evaluation based on the current semantics whereby scenario understanding is incorporated into the prediction task. The self-evaluation determines the best prediction model for each scene and detects when the available predictors cannot handle the scene. The latter case of an *invalid* scene occurs if no predictor is expected to output a trajectory prediction below a certain error threshold of a specific metric. The same metric defines the best prediction. Thus, the metric definition and *invalid*-threshold are crucial factors for the evaluation. With this specification and the given semantic information, the Selector Model (G_SEL) evaluates the scene (Figure 4.3).

The evaluation does not include an analysis of the latent scene representation. Only the resulting error of the predicted trajectory is the target value of the evaluation. Former studies revealed that no distinct relation between the latent representation and the output error could be found [312]. Even though correlations between the latent representation and the output error could be determined if the data is synthetically split in dedicated maneuvers [313], no scalable solution was achieved. The missing correlation can be explained by the fact that the latent representation is overdetermined in state-of-the-art models as the latent variables correlate with each other [314]. For these reasons, the `G_SEL` is entirely optimized on the model's output.

### 4.3.1 Evaluation Metric

The evaluation metric used is the mean RMSE over the prediction length $n_{\text{pred}}$ per prediction sample $l$, which corresponds to the average displacement error used in prediction challenges [15, 81, 136, 137]. The RMSE is defined between the predicted trajectory $\boldsymbol{x}_{\text{pred}}$ and the ground truth $\boldsymbol{x}_{\text{GT}}$ by:

$$\text{RMSE}_l = \sqrt{\frac{\sum_{m=1}^{n_{\text{pred}}} \|\boldsymbol{x}_{m,l,\text{pred}} - \boldsymbol{x}_{m,l,\text{GT}}\|_2^2}{n_{\text{pred}}}} \tag{4.3}$$

By means of this metric, the best prediction can be unambiguously defined for each scene by the lowest RMSE. Alternatively, the final displacement error could be used, which is another common metric in the state of the art [15, 81, 136, 137]. This metric determines the $\mathcal{L}_2$-distance between the endpoint of the predicted trajectory and the ground truth. Thus, the final displacement does not reflect the course of the trajectory along the prediction horizon compared to the RMSE, and deviations at the beginning of the prediction are not reflected. For these reasons, it is not considered further.

The definition of an *invalid* scene requires the specification of an error threshold $\varepsilon$. Two variants are proposed. One option is to set a threshold relative to the predictors' error distribution $\varepsilon_{\text{rel}}$. During training, this threshold is created dynamically as a percentile of the achieved prediction accuracy. This option is useful in unknown scenes without knowledge about the required absolute prediction accuracy. The other option is to set the threshold to an absolute RMSE-value $\varepsilon_{\text{abs,RMSE}}$. This is useful to base the ego behavior on an expected minimum of the prediction accuracy for a defined full-stack application in known ODDs.

### 4.3.2 Selector Model

The second part of the evaluation is the selector model G_SEL. It uses the three encodings of the scene image, L_LSTM, and DG_LSTM as input (Figure 4.3). Thus, all information given to the predictors is available to the G_SEL. In addition, reusing the encoders is beneficial as the network size grows only by the selector head. The G_SEL is constructed out of linear-dense layers and followed by a SoftMax activation function to normalize the output for the classification task. The output is of dimension $n_{\text{tp}} + 1 = 4$ as the model has to classify between the best among the $n_{\text{tp}}$ predictors given and the additional option of an *invalid* scene if none of the predictions models is expected to output a trajectory below the error threshold $\varepsilon$.

The execution during inference is realized modularly. At first, the input data are passed through the three encoders. Then, the G_SEL is executed to evaluate the scene and to select the best predictor, or in case none is suitable, to classify it *invalid*. In the case that one of the predictors is chosen, the respective model (CV) or decoder (L_LSTM, DG_LSTM) is executed and the predicted trajectory is outputted. If the scene is classified *invalid*, no model is executed, and no trajectory is outputted.

## 4.4 Data and Training

This section presents the data processing and training procedure utilized for the self-evaluation method. Subsection 4.4.1 introduces the data used for the development and outlines the data processing. Appendix C provides a complementary analysis of the training and test data distribution. The training and optimization process of the model is described in Subsection 4.4.2.

### 4.4.1 Data Processing

The data utilized in this study is taken from the scenario library provided by *CommonRoad*. The scenario list taken from the library is first randomly split into three subsets to ensure the separation of training, validation, and test data. By this, there are no scenes from different subsets corresponding to the same scenario. In total, $339\,051$ samples of prediction scenes are extracted. These samples are composed of $3\,s$ object history and the road map as input, along with $5\,s$ ground truth data to be predicted. Additionally, the object history of all surrounding objects is added to each scene to enable the usage of interaction-aware models. The minimal scenario length is defined through the definition of object history length and prediction horizon. Thus, short scenarios within the scenario library are not considered during the data processing. Due to this design decision, $6.98\,\%$ of the scenarios from the library are neglected, which is expected to not influence the method's validity due to the adequate number of extracted prediction scenes. Since some objects do not exist throughout the whole scenario, only the surrounding objects, which are presented during the whole history and future of the target object, are considered for the respective scene. In contrast, surrounding objects, which are not present during the full history and future of the target object, are not considered because it is assumed that these objects show fewer interactions. The assumption is justified by the following. At first, objects that are not present within the full object future are expected to show fewer interactions because they are at the border of the scenarios. Second, objects not present within the full object history and just entered at the edge of the scenario have less impact on the target object than objects in the scenario's center. Lastly, the presence of the unconsidered objects can be interpreted as additional noise in the data to augment the learning process.

Besides the object states, the graph creation is required to model the interactions between the road users through the `DG_LSTM`. Initially, no restrictions on the connections are assumed. It applies:

$$\epsilon_{ij} = \begin{cases} 0 & \text{for } i = j \\ 1 & \text{for } i \neq j \end{cases} \quad \text{for } i, j \in z_{\mathcal{N}} \tag{4.4}$$

with $\epsilon_{ij}$ being the index vector from node $j$ to node $i$ for all nodes $z_{\mathcal{N}}$ in a scene. The restriction of the `DG_LSTM` to shrink the connection by means of the threshold $\delta$ (Subsection 4.2.4) is applied during inference. By this, the distance threshold $\delta$ can be dynamically set.

### 4.4.2 Training and Optimization

Due to the integrative character of the self-evaluation model (Figure 4.3), which involves multiple connections between the encoders and decoders, an adjustment of the trainable parameters becomes necessary to train the corresponding predictors separately. A strategy is deployed to freeze certain network branches while others are trained. Via this method, a specific optimization of the predictors is possible despite the nested model architecture. The scene image encoding is always trained alongside the first predictor. During the training of subsequent predictors, the CNN-layers remain frozen. The training sequence of the self-evaluation method begins with the `L_LSTM`, followed by the training of the `DG_LSTM`. This results in an advantage to the `L_LSTM` as the scene image encoding is specifically tailored to it. However, the `DG_LSTM` benefits less from the scene image encoding due to its additional interaction knowledge from the GNN-embedding. Thus, this training order is expected to result in the optimal overall performance of both predictors. The training of the respective predictors is solely focused on minimizing their Euclidean prediction errors. Hence, their training processes do not consider the other predictors' existence or the `G_SEL`. The MSE-loss is used for the predictors' training. The selector model is trained last and is provided access to all the trained encoders. During the `G_SEL`-training, the encoders remain static, and only the parameters of the `G_SEL` itself are optimized. To address the classification problem during training, the NLL-loss function is employed after converting the probabilities in logarithmic representation. It was revealed to show better classification performance than the alternative option of the cross-entropy loss.

The network architecture and training parameters are optimized through Bayesian optimization [315]. The focus of the optimization process is multi-modal. The optimization objectives are a low overall RMSE in the prediction model's output and a high selection rate of the G_SEL. The selection rate is defined as the rate of correct selections by the selector model among the three predictors and the *invalid* option (Figure 4.3). The combined optimization is motivated by cases where the choice is between approximately equally accurate predictors. In these cases, a wrong selection is considered acceptable since it results in only a minor increase in the output's prediction error. In contrast, when the choice is between two divergent predictors, this decision greatly influences the overall prediction error in the output. Therefore, it becomes crucial to consider RMSE and selection rate in the optimization goal $\phi$:

$$\phi = \frac{\Phi - \Phi_{\mathrm{min}}}{1 - \Phi_{\mathrm{min}}} + \frac{\mathrm{RMSE}_{\mathrm{trg}}}{\mathrm{RMSE}_{\mathrm{val}}} \tag{4.5}$$

The equation displays the relationship between the optimization goal $\phi$, the selection rate of the selector model $\Phi$, and the RMSE of the model's output $\mathrm{RMSE}_{\mathrm{val}}$. The optimization objectives are balanced against each other using the two parameters of the minimal selection rate $\Phi_{\mathrm{min}}$ and the target RMSE $\mathrm{RMSE}_{\mathrm{trg}}$. By setting these two variables, the optimization function can be tuned. Two optimization steps are conducted:

1. **Network Optimization**
   The networks' parameters and the training parameters of learning rate, decay rate, and decay step size are optimized. The tuning parameters for the optimization goal (Equation (4.5)) are set to $\Phi_{\mathrm{min}} = 0.7$ and $\mathrm{RMSE}_{\mathrm{trgt}} = 0.3\,\mathrm{m}$ and a relative threshold $\varepsilon_{\mathrm{rel}} = 0.8$ is used. It corresponds to the $80\,\%$-quantile of the model's output RMSE and considers the dynamic improvement of the predictors' RMSE during the optimization.
2. **Behavior Optimization**
   The learning rate, decay rate, and decay step size are optimized, the networks' parameters remain unchanged. Three pairs of tuning parameters ($\Phi_{\mathrm{min}}$, $\mathrm{RMSE}_{\mathrm{trg}}$) with the goals of a high selection rate (index: $\mathrm{sel}$), a low overall RMSE (index: $\mathrm{rmse}$) and a balanced behavior between selection rate and low overall RMSE (index: $\mathrm{bl}$) are defined. For this optimization step, the RMSE-threshold for an *invalid* scene is set to an absolute value $\varepsilon_{\mathrm{abs,RMSE}} = 0.6221\,\mathrm{m}$, which is the $80\,\%$-quantile of the best individual predictor on the test data after the first optimization step.

The tuning parameters for the optimizations, the resulting parameters of the network from step one, and the training parameters determined by step two are listed in Appendix A.3.

## 4.5 Results

The self-evaluation method is validated on the dataset derived from *CommonRoad*. The Wale-Net [164] serves as a benchmark model. It is a pattern-based encoder-decoder model, using a SOT to encode interactions among road users and incorporates the same scene image encoding as the self-evaluation model (Table 2.2). Wale-Net's base architecture [106] achieved the first place on the Argoverse dataset [136] at the time of its release. For the validation, it is retrained on the same scenario split, using the parameters from Wale-Net's open-source repository [316]. Moreover, the three individual predictors from the hybrid prediction architecture are employed for comparison. The scope of the quantitative validation includes the prediction errors (Subsection 4.5.1) and the selection behavior (Subsection 4.5.2). In addition, qualitative results are presented to analyze the method's behavior (Subsection 4.5.3). The presented results are derived from the optimization with the goal of a high selection rate (Subsection 4.4.2). Subsection 4.5.1 and 4.5.2 are based on the author's publication [304].

### 4.5.1 Hybrid Prediction

The RMSE over the prediction horizon is depicted in Figure 4.6. The self-evaluation method, utilizing the G_SEL, the three individual predictors, and the benchmark model WaLe-Net are shown. Both pattern-based predictors surpass the benchmark model. Only the physics-based CV-model shows a worse prediction behavior. This model maintains high accuracy for short-term horizons up to $t_{\mathrm{pred}} = 0.3\,\mathrm{s}$, but its accuracy decreases as the prediction horizon extends. Among the individual predictors, the L_LSTM-model demonstrates the best performance and exceeds the DG_LSTM-model, despite not accounting for interactions between surrounding objects. It suggests that the specialized training of the scene image encoding, in conjunction with the L_LSTM, compensates for the lack of graph interaction knowledge inherent in the DG_LSTM. The self-evaluation method, which incorporates the three predictors and detects *invalid* predictions using the G_SEL, exceeds all individual predictors, achieving an average RMSE of $0.44\,\mathrm{m}$ on the subset of *valid* predictions. This subset contains $28\,291$ of the $34\,063$ test samples. Thus, with the error treshold of *invalid* prediction set to $\varepsilon_{\mathrm{abs,RMSE}} = 0.6221\,\mathrm{m}$, $16.95\,\%$ of the scenes are classified *invalid*. The final displacement error is reduced to $1.24\,\mathrm{m}$, in contrast to $5.84\,\mathrm{m}$ for the benchmark model. However, the proportion between the final and mean RMSE, $\mathrm{RMSE}_{\mathrm{final}}/\mathrm{RMSE}_{\mathrm{mean}}$, shows only a marginal improvement to $2.43$, compared to $2.56$ for the benchmark model. It indicates that the progressive increase in prediction error over the prediction horizon remains, which is reasonable because the G_SEL selects the entire trajectory from the single predictors and does not change its temporal expansion.

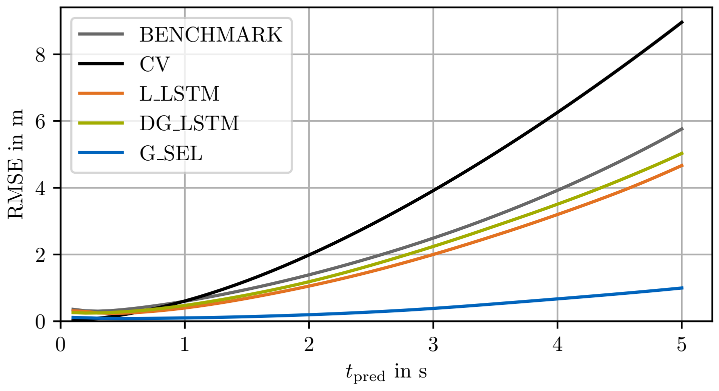


Figure 4.6: RMSE over the prediction horizon of the benchmark model Wale-Net, the three individual predictors (CV, L_LSTM, DG_LSTM), and the self-evaluation method by means of the selector model G_SEL [304]. The error threshold for an *invalid* prediction is $\varepsilon_{\mathrm{abs,RMSE}} = 0.6221\,\mathrm{m}$.

The capability of the self-evaluation method to prevent inaccurate predictions is examined through the analysis of the RMSE-distribution (Figure 4.7). It can be seen that the benchmark model is outperformed by the two individual predictors. Compared to the individual predictors, the error distribution is further reduced by the self-evaluation method with the G_SEL. The $90\,\%$-quantile can be decreased to $q90_{\mathrm{G\_SEL}} = 0.33\,\mathrm{m}$ for the given error threshold $\varepsilon_{\mathrm{abs,RMSE}} = 0.6221\,\mathrm{m}$, which means a reduction of $78.8\,\%$ compared to the benchmark model. The effectiveness of the self-evaluation method is further confirmed by the MissRate21 $\mathrm{MR}_{2_1}$ ($k = 1$, Equation (2.1)). The best predictor, the L_LSTM, achieves a value of $\mathrm{MR}_{2_1} = 14.53\,\%$, which is reduced to $2.00\,\%$ through the selector model. In comparison, the benchmark model's MissRate21 is at $21.83\,\%$.

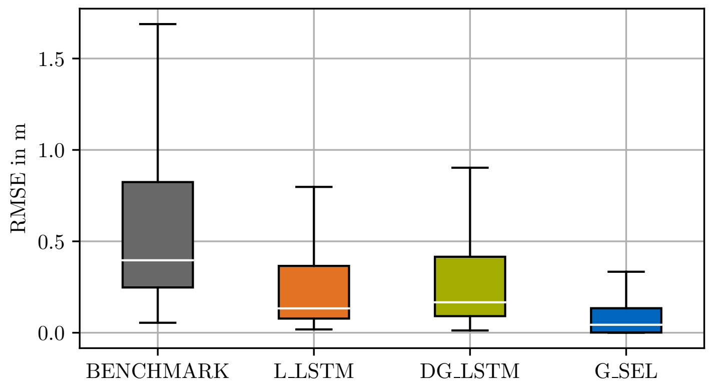
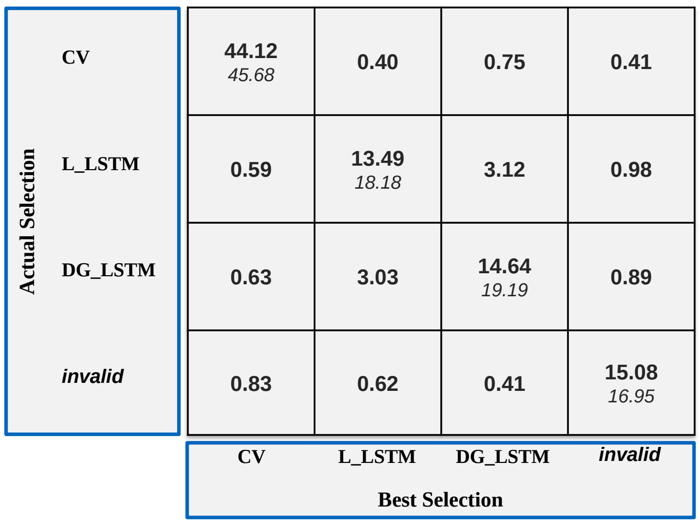


Figure 4.7: Box plot (median in white) of the mean RMSE over the prediction samples of the two best individual predictors (L_LSTM, DG_LSTM) and the self-evaluation method by means of the selector model G_SEL compared the benchmark model Wale-Net ($\varepsilon_{\text{abs,RMSE}} = 0.6221\,\text{m}$) [304].

### 4.5.2 Selection Behavior

The G_SEL's classification performance is analyzed using the confusion matrix (Figure 4.8). Four classes are represented: The three predictors and an additional category for *invalid* predictions ($n_{\text{tp}} + 1 = 4$). The columns represent the ground truth of the best selection for every scene. The rows show the actual selections made by the model.



| Actual Selection \ Best Selection | CV | L_LSTM | DG_LSTM | *invalid* |
|---|---|---|---|---|
| CV | 44.12 *45.68* | 0.40 | 0.75 | 0.41 |
| L_LSTM | 0.59 | 13.49 *18.18* | 3.12 | 0.98 |
| DG_LSTM | 0.63 | 3.03 | 14.64 *19.19* | 0.89 |
| *invalid* | 0.83 | 0.62 | 0.41 | 15.08 *16.95* |

Figure 4.8: Confusion matrix of the selector model ($\varepsilon_{\text{abs,RMSE}} = 0.6221\,\text{m}$) [304]. The optimal selection behavior is given in *italic*.

The confusion matrix reveals that every predictor has a share of more than 18 % of the most accurate predictions. The CV has the highest share of most accurate predictions, which shows that beyond the high prediction accuracy for pedestrians, the physics-based approach can be a *valid* choice for vehicle prediction. However, it must be considered that the CV has the highest mean RMSE (Figure 4.6). Even though the L_LSTM has the lowest mean RMSE, it has a smaller share of best predictions compared to the DG_LSTM. Regarding the selection rate, the G_SEL achieves a value of $\Phi = 87.3\,\%$ of correct selections. A limitation of the G_SEL in differentiating between the L_LSTM and the DG_LSTM, with more than 3 % incorrect selections in both directions, can be observed. The FP-rate of *valid* predictions that are incorrectly classified as *invalid* is 2.3 %. Conversely, the FN-rate for *invalid* predictions wrongly classified as *valid* is 13.1 %. The selector model demonstrates higher FP-rates toward the two pattern-based models compared to the CV-model. This could be attributed to the complexity of trajectories predicted by these two models, posing a challenge for the selector model in accurately understanding the scene and reliably choosing the *invalid* option. The tuning of the selector model toward specificity can be adjusted during training by weighting the *invalid* option in the NLL-loss.

The classification problem based on the defined evaluation metric is unambiguous. However, the extent of the consequences arising from an incorrect selection can vary, depending on the magnitude of deviation between the actual and the correct choice. Analyzing this aspect is crucial for understanding the model's selection behavior and its applicability in a full AV stack. In Figure 4.9, the sensitivity of the G_SEL toward *valid* predictions with varying tolerances for the correct selection $\Delta_{c\%}$ is depicted. A tolerance of $\Delta_{c\%}$ implies that a selection is considered correct if the RMSE error of the selected predictor is maximal $c\,\%$ above the RMSE of the best predictor ($\mathrm{RMSE}_{\mathrm{sel}}/\mathrm{RMSE}_{\mathrm{best}} - 1 \leq \Delta_{c\%}$). This study is performed across a spectrum of relative error thresholds $\varepsilon_{\mathrm{rel}}$ to examine their impact on the selection process. The relative error threshold specifies the ratio of *valid* predictions among the predictors' error distribution (Subsection 4.3.1).

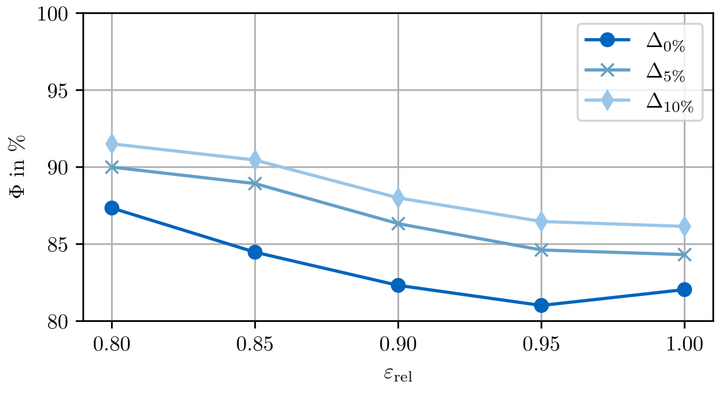


Figure 4.9: Analysis of the selector model's sensitivity toward *valid* predictors with varying tolerances $\Delta_{c\%}$ for the correct selection. The analysis is conducted over the relative error threshold $\varepsilon_{\mathrm{rel}}$ [304].

The analysis indicates that an average increase of $3.4\,\%$ in the selection rate is observed across all thresholds when a $5\,\%$ tolerance $\Delta_{5\%}$ is defined. Under the consideration that the model achieves selection rates between $81.0\,\%$ ($\varepsilon_{\mathrm{rel}} = 0.95$) and $87.3\,\%$ ($\varepsilon_{\mathrm{rel}} = 0.80$), the observed increase suggest that the discrepancy from $100\,\%$ correct selections is primarily due to unambiguous choices. The remaining samples to $100\,\%$ show a clear difference $> \Delta_{5\%}$ between the best prediction and the other predictors. This is further accentuated when examining the selection rate increase with $\Delta_{10\%}$. The increased tolerance leads to a $5.08\,\%$ rise in average compared to the baseline, indicating that around two-thirds of the remaining incorrect selections exhibit a relative deviation exceeding $10\,\%$ between the best and other predictors. Along the error threshold $\varepsilon_{\mathrm{rel}}$, the absolute increase of the selection rate with $\Delta_{c\%}$ is constant for $0.85 \leq \varepsilon_{\mathrm{rel}} \leq 0.95$. In contrast, for $\varepsilon_{\mathrm{rel}} = 0.8$ and $\varepsilon_{\mathrm{rel}} = 1.0$, the increase diminishes. In these cases, the absolute increase is less than $2.7\,\%$ for $\Delta_{5\%}$ and smaller than $4.2\,\%$ for $\Delta_{10\%}$. Thus, for a strict filter of *invalid* scenes ($\varepsilon_{\mathrm{rel}} = 0.80$) or the consideration of all scenes ($\varepsilon_{\mathrm{rel}} = 1.00$), the choice is clearer compared to the filter thresholds in between. The assumption that most evaluated scenes are an unambiguous choice for the best predictor can be further validated by the error distribution of the predictors on test data. When analyzing this, an average difference of $0.32\,\mathrm{m}$ between the best and second-best RMSE of the three predictors can be observed (standard deviation of $0.86\,\mathrm{m}$). Hence, there is a clear difference compared to the mean RMSE (Figure 4.6).

Lastly, the actual effectiveness of the self-evaluation method to enhance the accuracy of prediction outputs is examined (Figure 4.10). The analysis compares the RMSE of the self-evaluation method across various error thresholds with those of an optimal selector, the best individual predictor, and a random selector. Compared to the optimal selector (yellow), the RMSE of the G_SEL is, on average, $0.19\,\mathrm{m}$ higher. At thresholds of $\varepsilon_{\mathrm{rel}} = 0.8$ and $\varepsilon_{\mathrm{rel}} = 0.85$, the G_SEL's performance closely matches that of the optimal selector, but a decline in performance is noted at higher thresholds. Relative to the best individual predictor, the G_SEL demonstrates an improvement in the output RMSE even at a threshold of $\varepsilon_{\mathrm{rel}} = 1.0$ with no *invalid* predictions considered. Compared to a random selector, the results confirm that G_SEL method effectively identifies the model with the lowest RMSE.

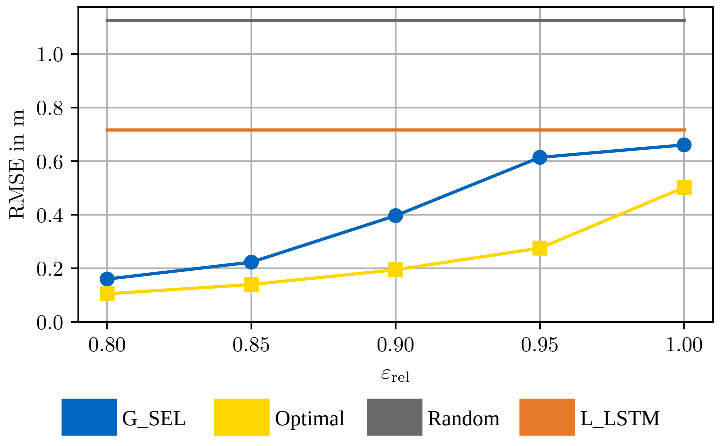


Figure 4.10: Mean RMSE over the prediction samples for varying error thresholds of the self-evaluation method compared to an optimal selector and a random selector [304]. In addition, the best individual predictor without self-evaluation, the L_LSTM, is shown.

## 4.5.3 Qualitative Evaluation

The qualitative evaluation aims to highlight the method's characteristics, which comprise the prediction behavior of the individual predictors and the selection behavior, on specific scene samples.

Figure 4.11 depicts the advantages of the CV-model. The target object drives in a steady state, which can be derived from its past states (black dotted). No objects are in front on the straight road, and the vehicles behind drive at similar speeds. Thus, low interaction is expected. The CV-prediction (orange) extrapolates the current state and matches the ground truth (black solid) in this scene with high accuracy. The two pattern-based predictions of the L_LSTM (green) and the DG_LSTM (gray) fall short in the longitudinal direction and drift to the right boundary of the road. The latter can be explained by the given road geometry, which these two models consider through scene image encoding. The lateral drift results from the merging second lane. Thus, the two models can not resolve that the target object stays in the lane, it currently drives on. The selector model (blue dashed) correctly selects the CV-model as the best prediction.

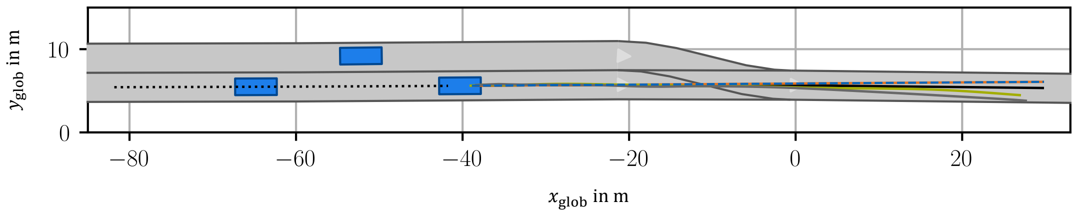


Figure 4.11: A scene with steady-state driving and the most accurate prediction by the CV-model, which is correctly selected by the self-evaluation method. The object's history (black dotted), future ground truth (black solid), the three predictors (CV in orange, L_LSTM in green, DG_LSTM in gray) and the output of the self-evaluation method (blue dashed) are shown.

In contrast, the CV-model (orange) fails when the object is turning (Figure 4.12). Without semantic knowledge, the predicted trajectory is outputted straight and exits the road. Alternatively, considering the object's turn rate using a CTRV-model would also be unsuitable because an overshoot toward the approaching traffic would result. The pattern-based models benefit from the input information of the full object history and the road map. Both output trajectories have low lateral errors. The DG_LSTM (gray) overestimates the influence of the approaching traffic and predicts larger longitudinal progress than the L_LSTM (green) for a high safety distance between the target object and the surroundings towards the end of the prediction horizon. The L_LSTM without interaction awareness predicts steady longitudinal progress, which matches the ground truth (black solid). The selector model correctly identifies this trajectory as the most accurate.

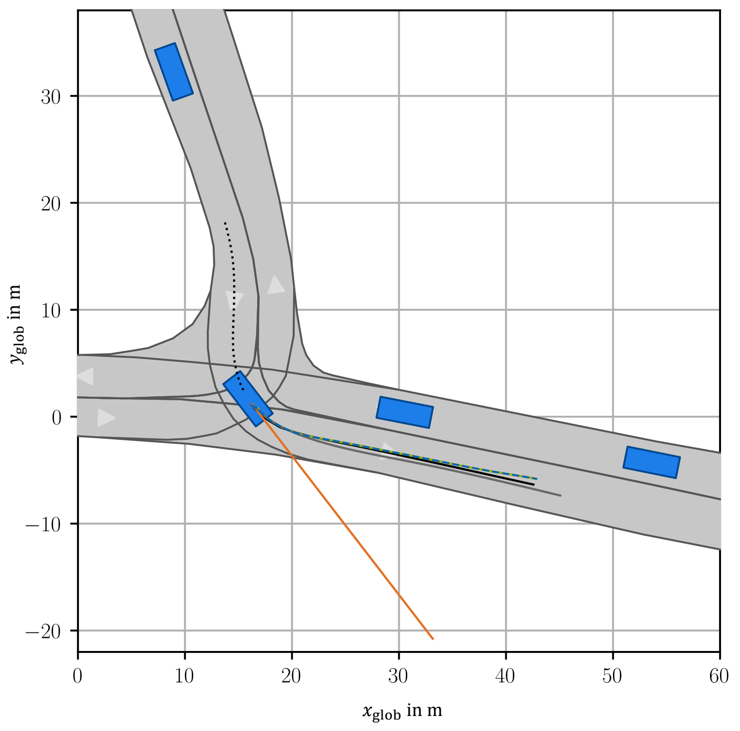


Figure 4.12: A scene with the most accuaurate prediction by the L_LSTM model, correctly selected by the G_SEL. The constant velocity assumption does not hold. The object's history (black dotted), future ground truth (black solid), the three predictors (CV in orange, L_LSTM in green, DG_LSTM in gray) and the output of the self-evaluation method (blue dashed) are shown.

The scene, depicted in Figure 4.13, highlights the benefits of the interaction-aware DG_LSTM-model. The target object is a stopping cyclist at the right road boundary. To the cyclist's left on the same lane and in front on the merging lane are motorcycles. Behind the target object is a convoy of cars. Since the stopping cyclist's speed is zero (black dotted object history not visible), the prediction of the CV is that the object remains still. In contrast, the L_LSTM outputs a short forwarded prediction because it expects the objects to move along the road. Since this assumption is of high uncertainty without further knowledge of the surrounding objects, the L_LSTM's trajectory falls short in the longitudinal direction and is noisy in the first steps with an overshoot in the opposite direction. The DG_LSTM derives a suitable assumption from the neighboring and trailing objects. It assumes that the target object can move forward in parallel to the motorcyclist on the left and is obligated to do so because of the trailing traffic. The longitudinal progress matches the ground truth. However, in the lateral direction, the prediction drifts to the lane's centerline. This can be explained by the second motorcyclist at the merging road from the right. Without the constraints of external traffic rules, the DG_LSTM bases its prediction on the learned pattern that objects keep a safety distance between each other and the shift to the centerline results. The selector model is able to determine the correct predictor. The DG_LSTM is the best choice for this dense and transient scene.

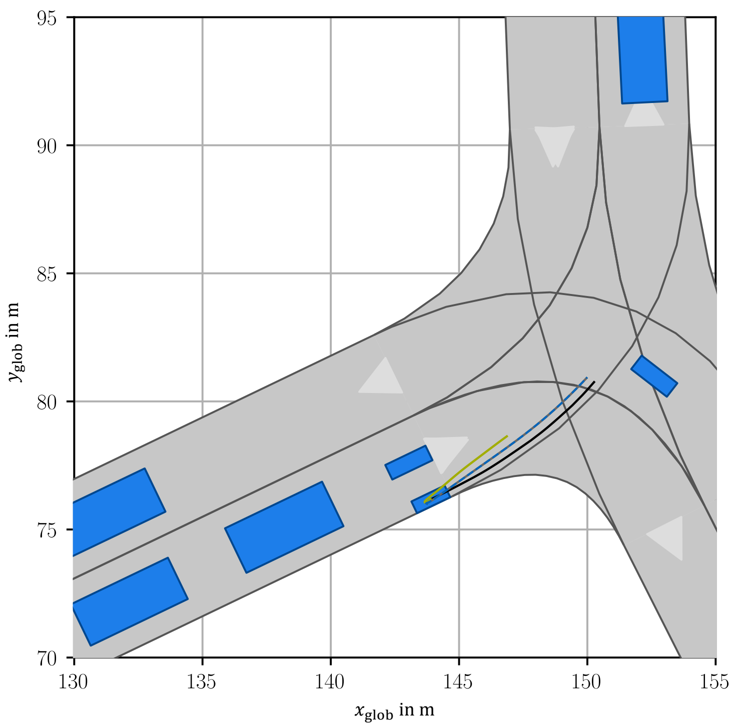


Figure 4.13: A scene with the most accurate prediction by the DG_LSTM model, correctly selected by the G_SEL. Interaction-awareness benefits can be seen by the standing object being predicted forward because of the trailing traffic. The object's history (black dotted), future ground truth (black solid), the three predictors (CV in orange, L_LSTM in green, DG_LSTM in gray) and the output of the self-evaluation method (blue dashed) are shown.

Interaction awareness can also lead to conservative prediction behavior in dense traffic situations. In Figure 4.14, the target object is driving toward a junction and hasn't started decelerating. The GCN-based interaction model considers in total six surrounding objects, of which five are approaching the junction. Based on the current steady state driving, the semantic knowledge of the upcoming junction, and the approaching objects, the DG_LSTM predicts an abrupt standstill of the target object (gray). Thus, it fails to resolve the interactive behavior in the situation and violates the dynamic limits of the target object. However, the ground truth movement of accelerating and turning to the left is hard to predict as there are other options of turning to the right and stopping at the junction entry. Considering traffic rules could remedy the latter issue to distinguish between standstill and accelerating. However, the reliance on these rules is risky in case of incorrect behavior of the further road users. Thus, considering rules has to be balanced against the objects' dynamic behavior. The longitudinal acceleration can not be derived from the target object's history at this time step because it is in a steady state. The L_LSTM outputs the best prediction by assuming a small acceleration. The selector model chooses this model. However, in this case, the prediction should be classified as *invalid* because all three predictors output trajectories above the predefined error threshold. The G_SEL fails to understand this scene.

Another wrong classified scene is shown in Figure 4.15. The target object is on a straight road in a convoy of vehicles with similar speeds. The G_SEL selects the L_LSTM. However, excluding the interaction model leads to a trajectory that exceeds the ground truth in the longitudinal direction and predicts the object to yield close to the leading object. The selector's wrong choice can be explained by the simple road geometry, in which the linear encoding is expected to enable an accurate prediction. The interaction-aware DG_LSTM considers the leading vehicle and outputs a short trajectory, which matches the ground truth with high accuracy. The linear extrapolation using the CV outputs a trajectory that is too short because the acceleration can not be modeled.

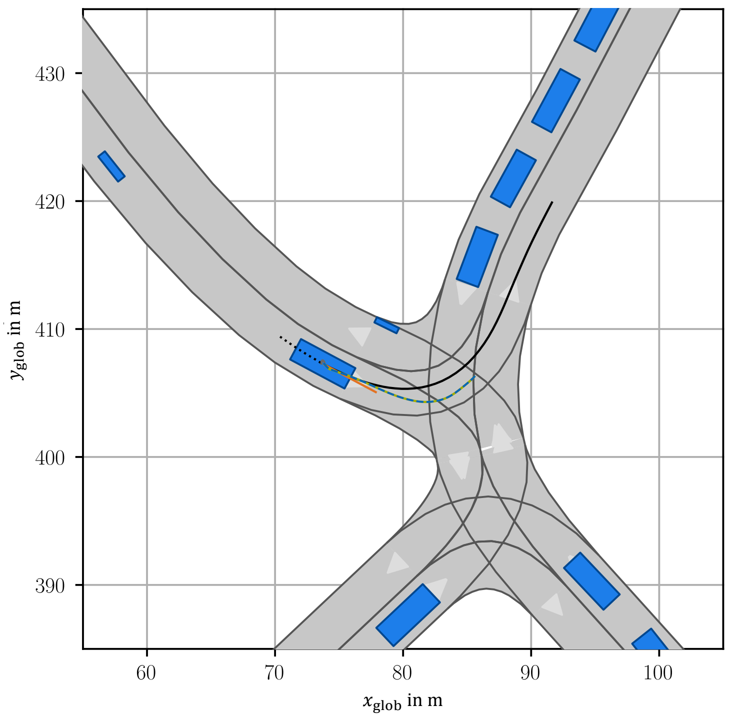


Figure 4.14: A scene with dense traffic and complex road geometry. The interaction awareness fails to resolve the scene and outputs a dynamically unfeasible trajectory (gray). None of the three predictors is able to determine an accurate solution. The selector model wrongly classifies this scene as *valid*. The object's history (black dotted), future ground truth (black solid), the three predictors (CV in orange, L_LSTM in green, DG_LSTM in gray) and the output of the self-evaluation method (blue dashed) are shown.

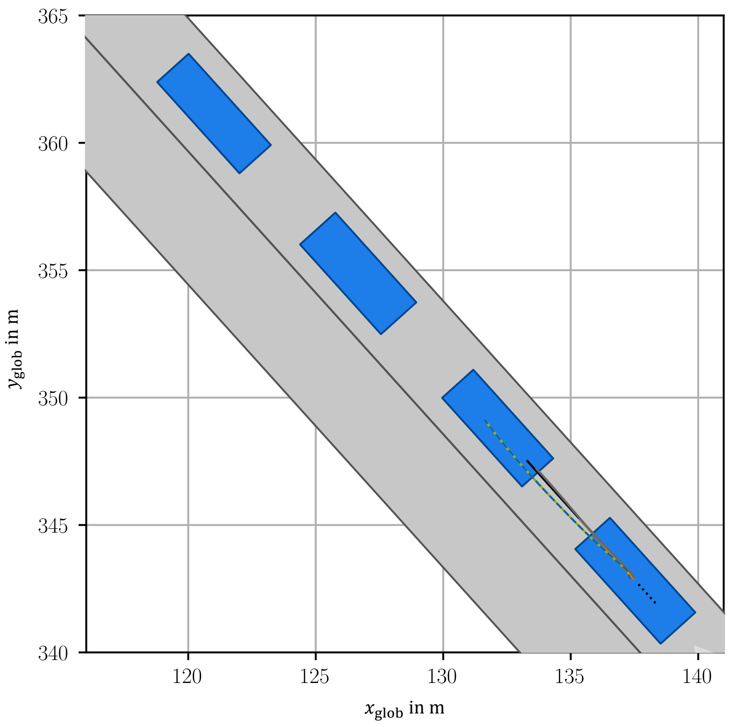


Figure 4.15: A scene with a convoy of vehicles. The selector model wrongly chooses the L_LSTM (green) because of the straight road geometry. Instead, the DG_LSTM (gray) is the correct choice because it considers the leading vehicle and outputs a shorter prediction. The object's history (black dotted), future ground truth (black solid), the three predictors (CV in orange, L_LSTM in green, DG_LSTM in gray) and the output of the self-evaluation method (blue dashed) are shown.

## 4.6 Ablation Study

Besides the presented results with parameters obtained from the Bayesian optimization (Appendix A.3), an ablation study on multiple parameters is conducted. At first, the parameters of the scene image encoding are varied (Subsection 4.6.1), which both pattern-based predictors and the selector model use. Afterward, the parameters of the `L_LSTM` (Subsection 4.6.2) and the `DG_LSTM` (Subsection 4.6.3) are permuted and the resulting prediction accuracy is investigated. At last, the self-evaluation method under varying constitutions of the hybrid prediction model is analyzed (Subsection 4.6.4).

### 4.6.1 Scene Image Encoding

The ablation study of the scene image encoder comprises the variation of the two parameters of the input image filters $l_{\mathrm{sc,in}}$ and output image filters $l_{\mathrm{sc,out}}$ (Table 4.1). In addition, the case of neglecting the scene image encoding in the trajectory predictors is analyzed. The parameter variation is evaluated by means of the RMSE-values of the `L_LSTM` $\mathrm{RMSE}_{\mathrm{li}}$ and `DG_LSTM` $\mathrm{RMSE}_{\mathrm{dg}}$. To consider the influence of the network's non-deterministic behavior due to random initialization of the weights and random shuffle of the training samples, the default network is trained eight times with the same parameterization. From these trainings, a standard deviation of $\sigma_{\mathrm{li}} = 0.03\,\mathrm{m}$ for the `L_LSTM` results, which are $1.88\,\%$ deviation in relation to the mean RMSE value of $1.76\,\mathrm{m}$. In case of the `DG_LSTM`, a standard deviation of $\sigma_{\mathrm{dg}} = 0.04\,\mathrm{m}$ around the mean value of $1.80\,\mathrm{m}$ is observed, which are $2.32\,\%$.

Table 4.1: Ablation study of the scene image encoding. Variations of the number of input image filters $l_{\mathrm{sc,in}}$ and the number of output image filters $l_{\mathrm{sc,out}}$ are conducted. The default configuration is given as benchmark ($l_{\mathrm{sc,in}} = 32$, $l_{\mathrm{sc,out}} = 16$). In addition, the prediction accuracy without a scene image encoding is examined. The RMSEs of the `L_LSTM` and the `DG_LSTM` are investigated. Minimal values for each column are given in bold.

| **Parameter** | **Value** | **$\mathrm{RMSE}_{\mathrm{li}}$ in m** | | **$\mathrm{RMSE}_{\mathrm{dg}}$ in m** | |
|---|---|---|---|---|---|
| | | **abs** | **rel** | **abs** | **rel** |
| Default | - | 1.76 | - | 1.80 | - |
| Number of input image filters $l_{\mathrm{sc,in}}$ | 16 | 1.75 | −0.40 % | 1.81 | 0.17 % |
| | 48 | 1.78 | 0.97 % | 1.81 | 0.50 % |
| | 64 | 1.84 | 4.38 % | 1.87 | 3.72 % |
| Number of output image filters $l_{\mathrm{sc,out}}$ | 8 | 1.84 | 4.60 % | 1.94 | 7.77 % |
| | 24 | 1.75 | −0.45 % | **1.78** | **−1.11** % |
| | 32 | **1.65** | **−6.37** % | 1.86 | 3.33 % |
| | 40 | 1.66 | −5.51 % | 1.82 | 0.94 % |
| No scene image | - | 2.52 | 43.21 % | 2.65 | 47.22 % |

The variation of the number of input image filters around the default value of 32 shows low variation in the RMSE for values of 16 and 48. Thus, the encoding capabilities of the network are agnostic within this range. A further increase to $l_{\mathrm{sc,in}} = 64$ results in a higher RMSE for both predictors. Besides the higher number of network parameters correlating with higher computational effort, the prediction accuracy diminishes. Under consideration that there is a low change in the predictors' RMSE between 16 and 48, the deterioration at $l_{\mathrm{sc,in}} = 64$ can be explained by overfitting due to the large network size. The large scene image encoder overfits the low-complex task of extracting the centerlines.

The number of output filters is varied around the default value of 16. Halving the number of filters increases the RMSE of both predictors. Thus, the network's capability to extract map information is reduced due to the low number of parameters. The increase of $l_{\mathrm{sc,out}}$ results in a low variation of the `DG_LSTM`'s RMSE. It can be interpreted that the model extracts less information from the scene image encoding. In contrast, the

L_LSTM's RMSE improves with a higher number of output image filters. Hence, the scene image encoding is able to extract more relevant information for the L_LSTM.

Lastly, neglecting the scene image encoding and relying the prediction only on the extraction of the objects' past states is investigated. While this architecture is beneficial because of the smaller amount of parameters, the RMSEs increases by more than $43\,\%$ for both predictors. It highlights the relevance of semantic knowledge to the prediction models.

## 4.6.2 Linear LSTM Model

The ablation study of the L_LSTM comprises the parameter variations of the linear encoder embedding, LSTM encoder, latent embedding, and LSTM decoder (Table 4.2). The study is evaluated by the RMSE of the prediction model and its miss rate $\mathrm{MR}_{2_1}$. The initial analysis of the model's variance by eight trainings with the default configuration results in a standard deviation of $\sigma_{\mathrm{li}} = 0.03\,\mathrm{m}$. In relation to the mean RMSE value of $1.76\,\mathrm{m}$, it is a relative variation of $1.88\,\%$.

Table 4.2: Ablation study of the L_LSTM. Variations of the linear embedding, LSTM encoder, latent embedding, and LSTM decoder are conducted. The default configuration is given as benchmark ($l_{\mathrm{li}} = 1$, $h_{\mathrm{li}} = 50$, $h_{\mathrm{li,emb}} = 50$, $l_{\mathrm{enc}} = 1$, $h_{\mathrm{enc}} = 50$, $h_{\mathrm{lat}} = 48$, $h_{\mathrm{dec}} = 128$). The RMSE and the miss rate are investigated. Minimal values for each column are given in bold.

| **Parameter** | **Value** | RMSE in m | | $\mathrm{MR}_{2_1}$ in % | |
|---|---|---|---|---|---|
| | | **abs** | **rel** | **abs** | **rel** |
| Default | - | 1.76 | - | 14.30 | - |
| Number of linear hidden layers $l_{\mathrm{li}}$ | 2 | 1.73 | −1.82 % | 14.13 | −1.19 % |
| | 3 | 1.79 | 1.53 % | 16.08 | 12.45 % |
| | 4 | 1.79 | 1.53 % | 14.00 | −2.1 % |
| Size of linear hidden layers $h_{\mathrm{li}}$ | 30 | 1.70 | −3.47 % | 14.09 | −1.47 % |
| | 40 | 1.83 | 4.32 % | 15.19 | 6.22 % |
| | 60 | 1.72 | −1.99 % | **13.36** | **−6.57** % |
| | 70 | 1.82 | 3.24 % | 16.20 | 13.29 % |
| Size of linear input embedding $h_{\mathrm{li,emb}}$ | 30 | **1.67** | **−5.00** % | 15.67 | 9.58 % |
| | 40 | 1.73 | −1.53 % | 14.41 | 0.77 % |
| | 60 | 1.78 | 1.08 % | 14.61 | 2.17 % |
| | 70 | 1.72 | −2.16 % | 16.26 | 13.71 % |
| Number of hidden layers in LSTM encoder $l_{\mathrm{enc}}$ | 2 | 1.69 | −4.21 % | 13.72 | −4.06 % |
| | 3 | 1.72 | −1.99 % | 14.41 | 0.77 % |
| | 4 | 1.72 | −2.39 % | 14.15 | −1.05 % |
| Size hidden layers in LSTM encoder $h_{\mathrm{enc}}$ | 30 | 1.74 | −1.31 % | 15.72 | 9.93 % |
| | 40 | 1.74 | −0.97 % | 14.45 | 1.05 % |
| | 60 | 1.81 | 2.96 % | 15.63 | 9.3 % |
| | 70 | 1.74 | −1.25 % | 13.78 | −3.64 % |
| Size of latent embedding $h_{\mathrm{lat}}$ | 16 | 1.76 | −0.11 % | 14.87 | 3.99 % |
| | 32 | 1.80 | 2.44 % | 14.18 | −0.84 % |
| | 64 | 1.72 | −2.33 % | 16.12 | 12.73 % |
| | 80 | 1.73 | −1.76 % | 15.64 | 9.37 % |
| Size hidden layers in LSTM decoder $h_{\mathrm{dec}}$ | 32 | 1.86 | 5.63 % | 15.83 | 10.7 % |
| | 64 | 1.78 | 1.31 % | 15.46 | 8.11 % |
| | 196 | 1.79 | 2.05 % | 16.33 | 14.2 % |

The first varied parameter is the number of hidden layers in the linear input embedding $l_{\mathrm{li}}$. Increasing this value from the default setting of $l_{\mathrm{li}} = 1$ up to 4 results in RMSE-changes, which are within the variation of the model due to non-deterministic behavior. Thus, no significant changes can be observed. Regarding the miss rate, an increase by $12.45\,\%$ can be observed for $l_{\mathrm{li}} = 3$. However, this trend can not be confirmed for the other values, and no distinct tendency can be derived regarding the model behavior.

Varying the size of linear embedding's hidden layers $h_{\text{li}}$ from $30$ to $70$ (default value: $50$) shows a volatile behavior of the RMSE. The miss rate shows the same direction of improvement as the related RMSE. However, there is no proportional correlation between the value pairs. With $h_{\text{li}} = 60$, the lowest miss rate of all samples is achieved. Since the related RMSE is around the model's training variance, the conclusion that the model's prediction capabilities and reliability improve can not be drawn.

The variation of the linear input embedding size $h_{\text{li,emb}}$ achieves the best RMSE for $h_{\text{li,emb}} = 30$. Since the miss rate increases by $9.58\,\%$ for this sample, the overall improvement of the prediction model is not proven. Between $h_{\text{li,emb}} = 40$ and $h_{\text{li,emb}} = 70$, the RMSE varies within the magnitude of $1\sigma_{\text{li}}$, thus no clear change of the prediction accuracy can be observed. A remarkable result is that the miss rates increase by more than $9\,\%$ for $h_{\text{li,emb}} = 30$ and $h_{\text{li,emb}} = 70$ while the related RMSEs decrease by more than $2\,\%$. Hence, a qualitatively inverse correlation between the RMSE and the miss rate can be observed.

The variation of the LSTM encoder's hidden layers reveals a decrease for all samples with $l_{\text{enc}} > 1$ but stay within $\pm 3\sigma_{\text{li}}$ range. Despite the model's accuracy improvement, the growth of the total network size by $10\,\%$ per additional layer additionally opts against additional hidden layers. Besides the number of the encoder layers, their size $h_{\text{enc}}$ is varied around the default value of $50$. An inverse correlation between the RMSE and the miss rate can be seen for $h_{\text{enc}} < 50$. Increasing the layer size above $50$ results in ambiguous behavior of the prediction accuracy.

The encoded object states are embedded to prepare the latent space for the enrollment of the object's future motion in the decoder. The increase of the size of the linear latent embedding $h_{\text{lat}} > 48$ displays a decrease of the RMSE but an increase in the miss rate. Thus, an overall improvement in the prediction performance can not be identified.

Lastly, the size of the one-layered decoder $h_{\text{dec}}$ is analyzed. The variation of this parameter around the default value of $128$ results in a deterioration of the RMSE and the miss rate for all investigated parameter ranges. Hence, within the examined parameters, the setting of $h_{\text{dec}} = 128$ is the best choice.

### 4.6.3 Proximity-Dependent Graph-LSTM Model

The ablation study of the `DG_LSTM`'s prediction behavior is focussed on the GNN-embedding and the decoder (Table 4.3). Eight trainings with the default setting are conducted to put the study in the context of the model's variance. Around the mean value of $1.80\,\text{m}$, a standard deviation of $\sigma_{\text{dg}} = 0.04\,\text{m}$ results, which are $2.32\,\%$ in relation to the mean value.

The first investigated value is the distance threshold $\delta$, which limits the graph connections between surrounding objects (Figure 4.5). An increase $\delta > 20$ reveals a continuous RMSE-decrease up to a threshold of $\delta = 40$. However, this trend is stopped for $\delta = 50$ with an RMSE close to the default ones. In addition, the clear increase in the miss rate for $\delta = 50$ indicates that the range between 20 and 40 suits best the given model validation.

The message passing between the nodes in the graph is realized by a linear dense layer of size $m_{\text{dg}}$. The variation around the default value of $64$ exhibits that the RMSE's deviation is smaller than $2\sigma_{\text{dg}}$ for $32 \leq m_{\text{dg}} \leq 80$. Thus, no correlation between the RMSE and the message size could be found in this range. A further increase of the message size to $m_{\text{dg}} = 96$ leads to a deterioration of the RMSE and the miss rate, which is expected to result from overfitting due to the larger message size.

The number of hidden layers in the graph defines the order of connections between surrounding objects (Figure 2.5). For the default setting of $l_{\text{dg,emb}} = 1$, first-order connections result. Due to the high sensitivity to vanishing gradients of GCNs, a low number of layers is recommended [317]. This statement can be confirmed for the given variation. For $l_{\text{dg}} > 1$, an increase in the miss rate can be monitored. While the RMSE stays

Table 4.3: Ablation study of the DG_LSTM. Variations of the GCN-features (distance, message size, embedding size, hidden layers), latent embedding, and LSTM decoder are conducted. The default configuration is given as benchmark ($\delta = 20$, $m_{\mathrm{dg}} = 64$, $l_{\mathrm{dg}} = 1$, $h_{\mathrm{dg}} = 48$, $h_{\mathrm{dg,emb}} = 128$, $h_{\mathrm{lat}} = 48$, $h_{\mathrm{dec}} = 128$). The RMSE and the miss rate are investigated. Minimal values for each column are given in bold.

| **Parameter** | **Value** | RMSE in m | | $\mathrm{MR}_{2_1}$ in % | |
|---|---|---|---|---|---|
| | | **abs** | **rel** | **abs** | **rel** |
| Default | - | 1.80 | - | 14.28 | - |
| Distance threshold $\delta$ | 10 | 1.83 | 1.78 % | 16.03 | 12.25 % |
| | 30 | 1.71 | −5.11 % | **13.77** | **−3.57** % |
| | 40 | **1.66** | **−7.61** % | 13.92 | −2.52 % |
| | 50 | 1.78 | −1.17 % | 16.59 | 16.18 % |
| Size of graph node message $m_{\mathrm{dg}}$ | 32 | 1.83 | 1.44 % | 16.64 | 16.53 % |
| | 48 | 1.84 | 2.22 % | 17.03 | 19.26 % |
| | 80 | 1.76 | −2.44 % | 15.23 | 6.65 % |
| | 96 | 2.06 | 14.1 % | 18.06 | 26.47 % |
| Number of graph hidden layers $l_{\mathrm{dg}}$ | 2 | 1.86 | 3.22 % | 17.58 | 23.11 % |
| | 3 | 1.79 | −0.44 % | 16.41 | 14.92 % |
| | 4 | 3.08 | 70.74 % | 30.59 | 114.22 % |
| Size of graph hidden layers $h_{\mathrm{dg}}$ | 16 | 1.79 | −0.56 % | 16.90 | 18.35 % |
| | 32 | 1.82 | 1.17 % | 16.88 | 18.21 % |
| | 64 | 1.86 | 3.11 % | 17.17 | 20.24 % |
| | 80 | 1.81 | 0.39 % | 16.58 | 16.11 % |
| Size of graph embedding $h_{\mathrm{dg,emb}}$ | 32 | 1.84 | 2.0 % | 16.58 | 16.11 % |
| | 64 | 1.96 | 8.88 % | 17.41 | 21.92 % |
| | 196 | 1.84 | 2.0 % | 16.43 | 15.06 % |
| Size of latent embedding $h_{\mathrm{lat}}$ | 16 | 1.78 | −1.39 % | 15.58 | 9.10 % |
| | 32 | 1.82 | 1.17 % | 14.79 | 3.57 % |
| | 64 | 1.84 | 2.33 % | 16.28 | 14.01 % |
| | 80 | 1.86 | 3.28 % | 15.89 | 11.27 % |
| Size of LSTM decoder hidden layers $h_{\mathrm{dec}}$ | 32 | 1.85 | 2.83 % | 17.96 | 25.77 % |
| | 64 | 1.87 | 3.94 % | 16.43 | 15.06 % |
| | 196 | 1.86 | 3.11 % | 15.93 | 11.55 % |

within $2\sigma_{\mathrm{dg}}$ for $l_{\mathrm{dg}} < 4$, it worsens by more than 70 % for $l_{\mathrm{dg}} = 4$. The size of the hidden layers is the second parameter to adjust the aggregation function besides the number of hidden layers (Equation (4.2)). Similar to the previously investigated values of $m_{\mathrm{dg}}$ and $l_{\mathrm{dg,emb}}$, an increase in the miss rate for all analyzed parameters can be seen. In addition, the RMSE shows no improvement.

The ablation studies on the latent embedding size $h_{\mathrm{lat}}$ and the LSTM decoder size $h_{\mathrm{dec}}$ reveal variations of the RMSE smaller than $2\sigma_{\mathrm{dg}}$. Deteriorations of the miss rate up to 25 % can be observed, which suggests that the default setting ($h_{\mathrm{lat}} = 48$, $h_{\mathrm{dec}} = 128$) is a valid choice on the investigated data.

### 4.6.4 Hybrid Prediction

Beyond optimizing the particular predictors, combining multiple predictors for the best prediction behavior of the self-evaluation method is of interest. The term best, in this case, refers to multiple objectives. Primarily, a low average Euclidean prediction error, quantified by the RMSE, is desired. However, an optimization of the mean error value does not ensure the prevention of particular mispredictions. For that reason, the MissRate21 $\mathrm{MR}_{2_1}$ must be evaluated. Another aspect is the method's selection behavior. Thus, two further metrics are of interest. The selection rate is considered to reflect the self-evaluation method's capability to select the correct option in a given scene. This rate indicates a reliable understanding of traffic scenes to adjust the prediction behavior to the given circumstances. In addition, the rate of *invalid* predictions is of interest. It is desired to have a low number of *invalid* predictions, indicating a high generalizability of the method because many scenes can be accurately predicted. The variation of the hybrid prediction composition is analyzed by these four metrics (Table 4.4). The listed RMSEs and the miss rates are the values for the *valid* classified predictions. The denoted invalid rates correspond to the ground truth of the evaluation to not induce a bias by the model's selection capabilities. An *invalid* prediction is defined by the absolute RMSE threshold $\varepsilon_{\mathrm{abs,RMSE}} = 0.6221\,\mathrm{m}$. The study's benchmarks are the three conducted optimization runs of a high selection rate (index: $\mathrm{sel}$), a low overall RMSE (index: $\mathrm{rmse}$), and the balanced setup (index: $\mathrm{bl}$) (Subsection 4.4.2). The latter one is used to calculate the relative changes (Table 4.4). The varied network configurations are trained with the parameters from the balanced setup (Appendix A.3).

Table 4.4: Ablation study of the hybrid network configuration. The composition of the hybrid prediction model is varied between the three models of CV, L_LSTM and DG_LSTM. The procedure is repeated with a reduced selector model, whose encoder branches are limited to the given ones from the predictors ('red.'). The default configuration (CV + L_LSTM + DG_LSTM with the full selector model) with three different optimization goals is given as a benchmark. The RMSE, the miss rate $\mathrm{MR}_{2_1}$, the rate of *invalid* predictions with $\varepsilon_{\mathrm{abs,RMSE}} = 0.6221\,\mathrm{m}$, and the rate of correct selections $\Phi$ are investigated. The best values, direction indicated by the arrow, for each column are given in bold.

| Parameter | RMSE in m ↓ | | $\mathrm{MR}_{2_1}$ in % ↓ | | Invalid in % ↓ | | $\Phi$ in % ↑ | |
|---|---|---|---|---|---|---|---|---|
| | abs | rel | abs | rel | abs | rel | abs | rel |
| Balanced bl | 0.38 | - | 1.72 | - | **16.64** | - | 83.50 | - |
| High selection sel | 0.44 | 14.62 % | 2.00 | 16.28 % | 16.95 | 1.86 % | 87.34 | 4.60 % |
| Low RMSE rmse | **0.28** | **−25.84** % | **1.32** | **−23.25** % | 17.76 | 6.73 % | 84.35 | 1.02 % |
| CV + L_LSTM | 0.34 | −9.92 % | 1.52 | −11.63 % | 17.93 | 7.78 % | 93.18 | 11.53 % |
| CV + DG_LSTM | 0.41 | 6.01 % | 1.52 | −11.63 % | 18.55 | 11.49 % | 92.46 | 10.66 % |
| L_LSTM + DG_LSTM | 0.39 | 1.83 % | 1.77 | 2.91 % | 17.77 | 6.81 % | 80.99 | −3.06 % |
| CV | 0.36 | −6.01 % | 1.56 | −9.3 % | 37.91 | 127.88 % | **98.17** | **17.50** % |
| L_LSTM | 0.32 | −16.19 % | 1.59 | −7.56 % | 18.64 | 12.07 % | 95.40 | 14.18 % |
| DG_LSTM | 0.40 | 4.96 % | 1.41 | −18.02 % | 21.46 | 29.01 % | 95.64 | 14.47 % |
| CV + L_LSTM, red. | 0.32 | −17.23 % | 1.56 | −9.3 % | 19.47 | 17.03 % | 93.62 | 12.05 % |
| CV + DG_LSTM, red. | 0.42 | 9.14 % | 1.56 | −9.3 % | 19.23 | 15.6 % | 91.90 | 9.99 % |
| CV, red. | 0.62 | 62.37 % | 2.07 | 20.35 % | 37.91 | 127.88 % | 95.10 | 13.89 % |
| L_LSTM, red. | 0.35 | −7.31 % | 1.77 | 2.91 % | 18.90 | 13.62 % | 95.16 | 13.9 % |
| DG_LSTM, red. | 0.40 | 5.48 % | 1.70 | −1.16 % | 21.60 | 29.8 % | 95.43 | 14.22 % |

Within the three variations of the default configuration (CV + L_LSTM + DG_LSTM with the selector model), it can be seen that with different optimization goals, the RMSE and miss rate can be influenced. The balanced setup achieves the lowest *invalid* rate indicating the highest degree of finding *valid* solutions for a scene, while the selection rate $\Phi$ is the worst among the three optimizations. The high-selection optimization fulfills its goal with a selection rate $\Phi = 87.34\,\%$. Since the RMSE is less prioritized, it increases by $14.62\,\%$. The low-RMSE optimization reduces the RMSE by $25.84\,\%$ but shows an increase of *invalid* predictions. Thus, the improvement is in parts achieved by excluding more scenes.

The next three variations comprise the usage of two predictors while maintaining all three encoders of the scene image, L_LSTM and DG_LSTM available to the selector model (Figure 4.3). The combination of CV + L_LSTM shows an improvement in RMSE and miss rate. In addition, the selection rate $\Phi$ improves to $93.18\,\%$. However, the rate of *invalid* prediction increases. Thus, the model loses its ability to adjust to various scenes. The increase in the selection rate and rate of *invalid* predictions can also be observed for the combination CV + DG_LSTM. In comparison to the combination of L_LSTM + DG_LSTM with a selection rate of $80.99\,\%$, it can be concluded that a physics-based and a pattern-based model are easier to distinguish. The selection rate is additionally negatively influenced because combining two pattern-based models yields a worse differentiation between *valid* and *invalid* predictions since the two models are less interpretable regarding their prediction behavior in a scene.

The correlation between the available prediction models, rate of *invalid* predictions, and selection rate also applies to the next three variations, which use one predictor in combination with the full selector model. In these configurations, the selector model's task is distinguishing between *valid* and *invalid*. If only the CV-model is used, the highest selection rate of 98.17 % is achieved. The selector model is able to differentiate between steady-state driving and transient scenes. However, the rate of *invalid* predictions is more than twice as much compared to the balanced setting. It can be seen that using only the CV-model is insufficient to cover the motion patterns in the data. The usage of only one pattern-based model results in similar behavior. The RMSE and miss rate can be kept low by an increase of *invalid* predictions compared to the setups with multiple predictors. The selection rates of these configurations are above 95 %.

In the remaining studies, the variations of reducing the number of predictors to two and one are repeated with the difference that the encoders of the discarded predictors are also removed. Exemplary, in case of the combination CV + L_LSTM, red. the entire DG_LSTM model is deleted, and the selector model uses only the available encoders of the scene image and the L_LSTM (Figure 4.3). This procedure enables estimating the available encoders' influence on the selection behavior. Compared to the non-reduced cases, it can be observed that the selection rates for the dual- and single-model setups achieve similar scores above 91 %. Hence, the selector model can also with fewer encoders available distinguish between physics-based and pattern-based prediction scenes and the *invalid* case. The model is additionally able to determine *invalid* predictions without the encoding of any object movement only through the scene image encoding, which is the case for CV, red.

## 4.7 Analysis of Prediction Errors

For a reliable application, the knowledge about the prediction error in dependency on semantic features is relevant. In the following, the influence of the object-specific features of type and speed and the number of surrounding objects on the miss rate is analyzed. Out of special interest is additionally the miss rate over the prediction horizon because a direct adjustment can be made to a shorter horizon if a scene is classified *invalid*.

The relative distribution of the miss rate over the object types for the three individual predictors and the G_SEL is depicted in Figure 4.16. In comparison with the total ratio of object types in the test dataset (Figure C.1), the CV's miss rate distribution is similar to the total data distribution. The model does not consider the object type. No bias is induced. The pattern-based models (L_LSTM, DG_LSTM) have a share of 79.76 % and 80.23 % of miss rates by predicting cars, which is more than 10 % less than the occurrence of cars in the test dataset. This implies that for the remaining object types of truck, bus, bicycle, and motorcycle, their share is doubled, and by this, their object-specific probability of a miss rate is around twice their average miss rate. This bias toward minor road users is amplified for the fusion model G_SEL. While the share of miss rates for motorcycles remains at the same level as for the pattern-based predictors, the shares of miss rates for trucks, buses, and bicycles are around three times bigger than their share in the total dataset. Thus, the probability of a miss rate of these objects is three times higher than the average miss rate over all objects.

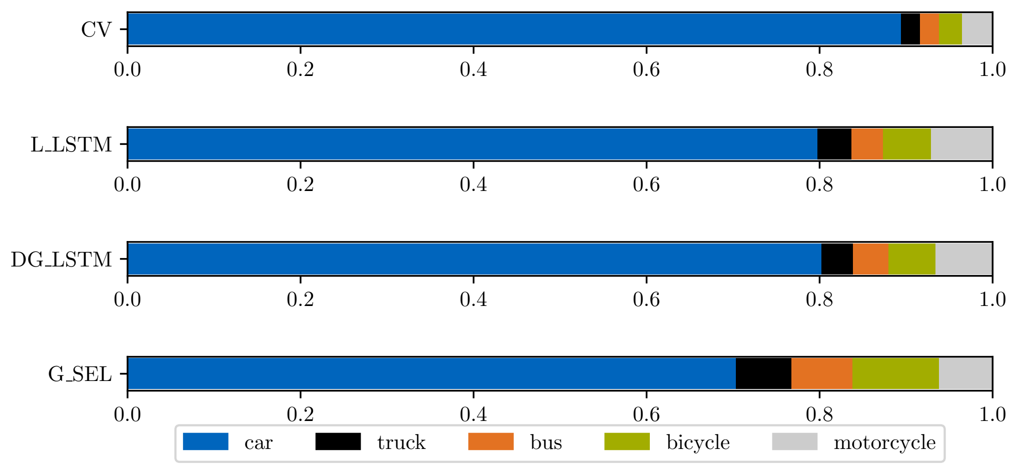


Figure 4.16: Miss rate distribution over the object types of the three predictors (CV, L_LSTM, DG_LSTM) and the combined self-evaluation method (G_SEL).

Figure 4.17 depicts the relative distribution of miss rates over the target object's speed. The CV-model shows a peak at $6\,\mathrm{m\,s^{-1}}$, which is the average speed of the objects in the dataset (Figure C.2). Besides, the CV-model's miss rate distribution shows the share of objects that are at a standstill but accelerate during the future prediction horizon, which is not predictable by this model. This particular motion behavior has the highest share of miss rates for the three remaining models. In case of the G_SEL, the share of miss rates at $v_{\mathrm{obj}} = 0$ is $56.31\,\%$. Thus, the selector model is not able to understand these scenes correctly. In addition, it can be concluded that the selector model shows high scenario understanding in the further speed range. It can be seen that the distribution of the G_SEL between $3\,\mathrm{m\,s^{-1}}$ and $18\,\mathrm{m\,s^{-1}}$ is below the ones of the individual predictors, except the peak at $5\,\mathrm{m\,s^{-1}}$. Hence, the limitations of the three predictors are detected in this speed range by the G_SEL.

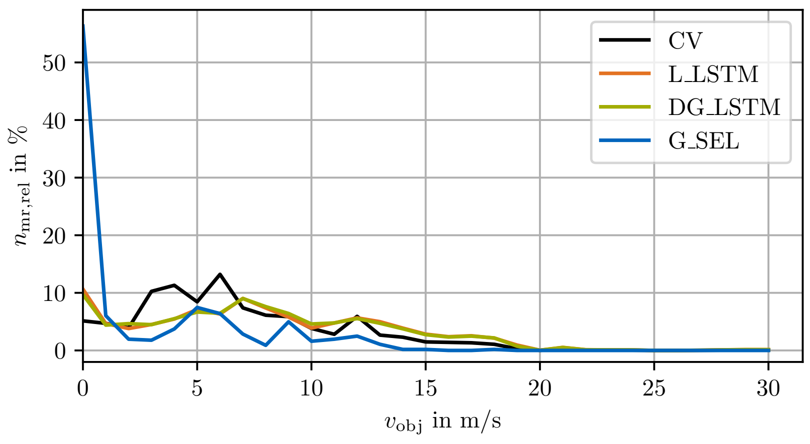


Figure 4.17: Miss rate distribution over the target object's speed of the three predictors (CV, L_LSTM, DG_LSTM) and the combined self-evaluation method (G_SEL).

The analysis of the misprediction distribution over the number of objects is visualized in Figure 4.18. Under consideration of the object distribution in the dataset (Figure C.2), no bias can be determined for the CV-model. In contrast, the pattern-based models and the fusion model have their maxima at $n_{\mathrm{obj}} = 61$. Together with the further peaks in the range of $25 \leq n_{\mathrm{obj}} \leq 60$, the analysis reveals that the models show higher rates of mispredictions in dense traffic scenarios compared to the majority of scenes with $n_{\mathrm{obj}} \leq 10$.

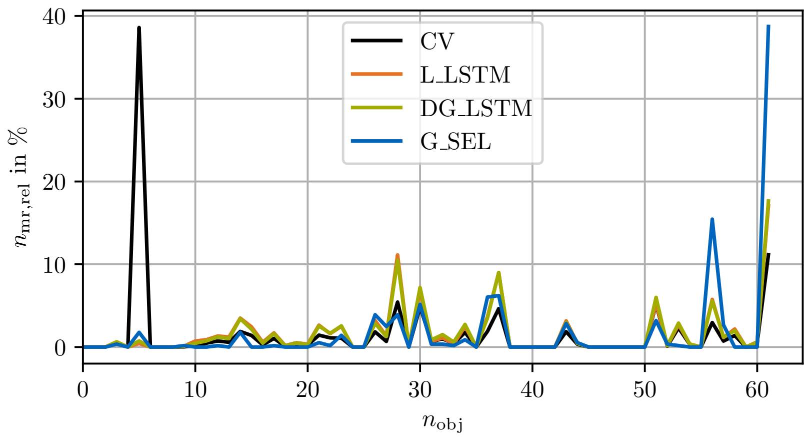


Figure 4.18: Miss rate distribution over the number of objects of the three predictors (`CV`, `L_LSTM`, `DG_LSTM`) and the combined self-evaluation method (`G_SEL`).

Lastly, the miss rate over the prediction horizon is investigated (Figure 4.19). Since the RMSE correlates with the prediction horizon because of the error summation (Equation (4.3)), the previous definition of an *invalid* through a fixed error threshold $\varepsilon_{\mathrm{abs,RMSE}}$ is not suitable for varying prediction horizons. Instead, the miss rate $\mathrm{MR}_{2_1}$ is used for the analysis. Except the range of $t_{\mathrm{pred}} < 1\,\mathrm{s}$, the `CV`-model's miss rate increases by a factor between $6.3\,\mathrm{s}^{-1}$ and $10.80\,\mathrm{s}^{-1}$. It can be seen that this model is on a similar level as the pattern-based model for short-term ($t_{\mathrm{pred}} < 1\,\mathrm{s}$) but diverges for longer prediction horizons. The `L_LSTM`'s curve has an average incline of $3.38\,\mathrm{s}^{-1}$ while the `DG_LSTM`'s incline is $3.22\,\mathrm{s}^{-1}$ in average. The `G_SEL`'s curve shows the most progressive behavior over the prediction horizon. Thus, reducing the prediction horizon is most effective for this model. Exemplary, the restriction of the prediction horizon to $t_{\mathrm{pred}} = 4\,\mathrm{s}$ would reduce the miss rate to $1.25\,\%$, which is a relative decrease of $37.30\,\%$.

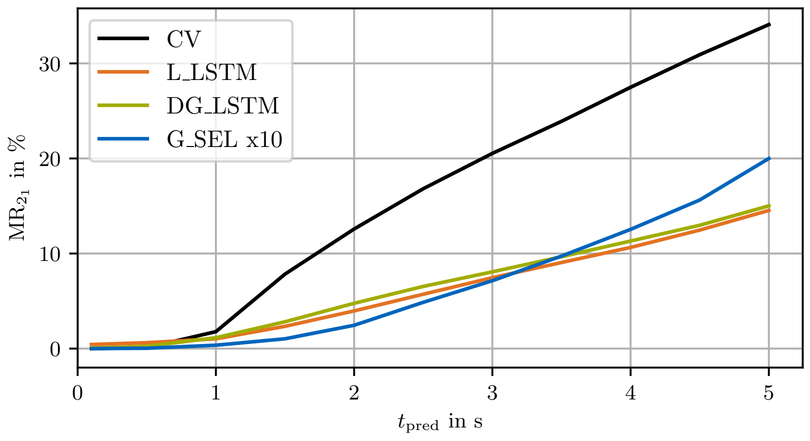


Figure 4.19: Miss rate distribution against the prediction horizon of the three predictors (`CV`, `L_LSTM`, `DG_LSTM`) and the combined self-evaluation method (`G_SEL`). The `G_SEL`'s miss rate is scaled by factor 10.

## 4.8 Discussion

The presented self-evaluation method aims to create a scene-dependent prediction behavior by the composition of multiple predictors. In addition, the method evaluates its algorithmic limitations and detects scenes that the predictors are not capable of accurately predicting. In the following, the validation of the results (Subsection 4.8.1), a weighted summation instead of the selection approach (Subsection 4.8.2) and the integration in a full AV software stack (Subsection 4.8.3) are discussed.

### 4.8.1 Validation of the Method

Reviewing the quantitative results, the benchmark model and all individual predictors are outperformed by the self-evaluation method in terms of RMSE and final displacement error (Figure 4.6). In addition, a reduction of the miss rate to $2.00\,\%$ is achieved compared to the benchmark model's miss rate of $21.83\,\%$. This is accomplished by the selection rate of $\Phi = 87.3\,\%$ and specificity of $97.7\,\%$ by the selector model (Figure 4.8). Thus, *invalid* predictions are detected at a high rate, erroneous predictions are avoided. Besides, the analyzed selection behavior indicates that each of the three predictors is used in a relevant ratio to output the best predictions, validating the choice of the individual models. An analysis of the sensitivity in selecting *valid* predictors reveals that choosing the correct predictor is a clear task (Figure 4.9). Only minor improvements in the selection rate are observed with an increased tolerance for the correct selection. Furthermore, the method's impact on the overall RMSE shows near-optimal selection behavior at error thresholds of $\varepsilon_{\text{rel}} = 0.8$ and $\varepsilon_{\text{rel}} = 0.85$ (Figure 4.10). Additionally, it is observed that even without specifying *invalid* predictions, the prediction accuracy is improved, proving the hybrid prediction approach's benefit. The two studies on the selection sensitivity (Figure 4.9) and the selection effectiveness (Figure 4.10) additionally reveal the relation between the selection rate and the specified error threshold. A continuous decline in the selection rate $\Phi$ from $\varepsilon_{\text{rel}} = 0.8$ to $\varepsilon_{\text{rel}} = 0.95$ and an increased gap to the optimal RMSE is observed. It needs further investigation to validate this observation since the shown results are conducted with the model's parameters optimized with a threshold of $\varepsilon_{\text{rel}} = 0.8$.

The presented qualitative results illustrate the prediction behavior of the individual predictors and the composed self-evaluation method (Subsection 4.5.3). Regarding the CV-model, its effectiveness in simple steady-state scenarios but limitations in complex, non-linear scenarios are depicted. This observation is confirmed by the model having the highest share of best predictions in the test dataset (Figure 4.8) but exhibiting the highest mean RMSE (Figure 4.6). The impact of interaction awareness is two-fold. On the one hand, considering surrounding road users enables a more comprehensive trajectory generation, which can surpass missing dynamic patterns in the object's motion history. On the other hand, the knowledge about surrounding objects leads to more conservative prediction behavior in dense traffic scenarios. In the quantitive evaluation, the interaction-aware DG_LSTM has a lower RMSE than the L_LSTM but a higher share of best predictions. Exemplary scenes in traffic convoys show that the selector model falls short in resolving the complexity of the scenes and distinguishing between the pattern-based predictors. This finding is confirmed by the confusion matrix, which shows a FP-rate of $18.78\,\%$ and $17.15\,\%$ if just the $2 \times 2$-matrix between these two models is analyzed (Figure 4.8).

The ablation study shows at first the importance of considering semantic encoding (Subsection 4.6.1). Despite the simple creation of the map in rastering the centerline of the road network around the target object, including this information improves the prediction performance. The study on the networks' parameters of the L_LSTM (Subsection 4.6.2) and the DG_LSTM (Subsection 4.6.3) reveal that the given parameterization, resulting from an optimization process, is suitable. The results demonstrate improvements by the variation of distinct layers, but no specific parameterization bottleneck is identified. Especially considering both the improvement of the RMSE and the miss rate has shown to be difficult because there is no consistent correlation between the two metrics. The occurrence of a negative correlation between these metrics could even be interpreted as a trade-off between a balanced prediction behavior with low miss rates or a more specific setup with lower RMSE in average but higher miss rates overall. A key finding for the `DG_LSTM` (Subsection 4.6.3) is the influence of the distance threshold $\delta$, which defines the upper limit for considering surrounding objects in the interaction model. Up to a distance of $\delta = 40$, an improvement in the prediction accuracy can be observed. The ablation study on the hybrid prediction configuration focuses on the network composition and the encoders available to the selection model (Subsection 4.6.4). In addition, the variation of the self-evaluation method's optimization goal emphasizes its individual adjustment possibilities. Focus points can either be set on the minimization of the RMSE, which implies an increase in the rate of *invalid* predictions, or on optimizing the selection rate, which results in an increase of the RMSE. The reduction of the number of used predictors yields an improved selection rate. The highest selection rate can be achieved by using only one predictor. In this case, the selector model has to distinguish solely whether the scene is *valid* or not. However, using only one model induces an increase in the *invalid* rate. It can be concluded that the adaptivity diminishes when attempting to predict large data accurately. To analyze the influence of the encoding branches on the selector model, a further study is conducted, in which the removal of a predictor implies the removal of its encoder to be used for the selection process. It reveals that the selector model is still able to fulfill its task. Thus, distinguishing two different prediction classes and detecting *invalid* scenes can be achieved with less encoded information. A limitation of the ablation study is the uni-modal variation. Dependencies by varying multiple parameters simultaneously are not depicted. In the multi-dimensional solution space, the movement is just along one axis, respectively. Thus, multi-dimensional gradients can not be found, which leaves room for further optimization.

Lastly, an analysis of the prediction error is realized (Section 4.7). A distinct bias of the pattern-based predictors and the fusion model toward lower-represented objects can be seen. Data augmentation and specific training constraints could remedy the bias against these object types. Regarding the application, the ego-motion behavior can be adjusted through this knowledge if these object types occur. In this context, it must be noted that the prediction behavior on object types not given in the data is not analyzed. In particular, the prediction behavior on pedestrians, which are not given in the dataset (Figure C.1), is of interest for the application in urban environments. Another influencing factor on the miss rate is the target object's speed. The analysis reveals a bias toward stillstanding objects, especially in the case of the `G_SEL`. While their future intention is not determinable by their motion history, semantic features must be included to determine the future movement. The analysis of the miss rates against the number of objects shows shortcomings for scenes with $n_{\text{obj}} > 10$. Thus, even though an interaction-aware model is considered, and employing the proximity threshold, the interaction is focussed on a few objects, the method fails to prevent miss rates in dense traffic scenes. The final analysis examines the correlation between the miss rate and the prediction horizon. With the knowledge of the incline of the miss rate along the horizon, an adjustment can be made to the desired average miss rate. The progressive shape of the `G_SEL`'s curve enables an effective reduction of the miss rate if the prediction horizon is decreased.

### 4.8.2 Weighted Summation

The proposed method exclusively takes the maximum of the selector model's output to select a model to predict the object's future motion. This differs from ensemble methods, which combine multiple models by a weighted sum. It also differs from the introduced MixNet (Section 3.3), which uses a weighted sum of base curves, enabling reliable prediction on closed road geometries.

An argument that supports a weighted summation instead of an exclusive selection is an increase in the temporal consistency of the prediction because no intermediate switches between the predictors occur. Furthermore, the comparison of multiple prediction models can be used to increase reliability. The divergence between multiple models can be evaluated as an indicator of high motion uncertainty. Thus, the models can be used to validate each other. However, the inversion that the output of three similar predictions yields an accurate prediction might not hold in complex scenes where all used predictors fail to encode the object's movement.

A drawback of ensemble methods is the calculation effort because multiple models must be executed. So the number of models or their complexity must be restricted. IMMs commonly use physics-based models for low calculation effort but are limited to short-term prediction. Computation times of ensemble methods with pattern-based models [20, 69] are not reported.

An ambiguous issue is the influence of a weighted summation of multiple predictors on the prediction accuracy. The merge of multiple models could enable a higher degree of adaptivity and motion behavior being modeled compared to the exclusive selection of one predictor. However, compared to MixNet's concept to define a set of base curves for a closed road geometry, which results in a convex solution space for the prediction output, the combination of multiple models and the application to open road geometries do not allow this quality guarantee (Figure 3.18). In addition, the predictors' behavior can strongly vary in contrast to the usage of base curves. Thus, the weighting of individual predictors does not guarantee a common motion behavior for an object. The variation of the predictors' behavior can be reduced by training specialized models for dedicated motion patterns or by the usage of base curves derived from the road geometry. However, in both cases, the summation does not necessarily yield a prediction within the driveable area of junctions or roundabouts (Figure 4.20). Also, the assumption that the prediction lies within the spanned solution space of the predictors does not hold for these road geometries.

In summary, the increased calculation effort and the ambiguous effect of merging multiple models are points of further investigation for the usage of a weighted sum approach.

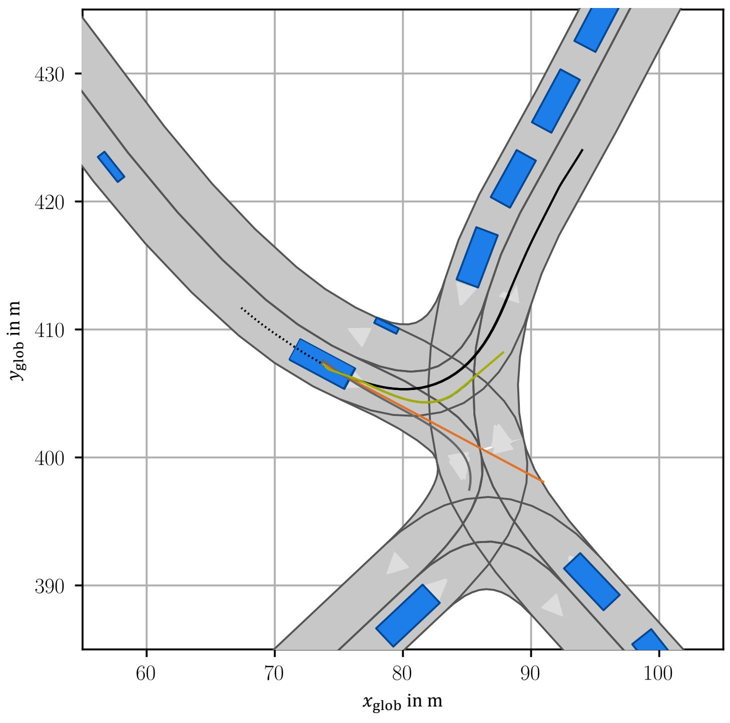


Figure 4.20: Synthetic modification of a scene with dense traffic and complex road geometry. The three predictors (CV in orange, L_LSTM in green, DG_LSTM in gray) show diverse behavior at the junction. All models show low accuracy in predicting the ground truth (black solid).

### 4.8.3 Full-Stack Integration

The developed method is validated in the benchmark framework *CommonRoad*, which is part of the introduced research environment (Section 4.1). The intended full-stack integration and real-world application of the self-evaluation method on *EDGAR* shall be discussed in the following.

The first point of discussion are the used predictors under the consideration of the target ODD. The used predictors must be tailored to the expected motion behavior, object types, data and semantic information availability, and degree of interactive and transient scenes. Alternatively, the same prediction method could be used multiple times, and specialized predictors could be implemented to cover the solution space. For example, multiple pattern-based models could be selected and trained on specific data clusters of the ODD.

Besides that, the selector model's classification behavior, particularly its specificity and sensitivity, and the error threshold must be fine-tuned in conjunction with the ego-motion planner. The FN-rate must align with the ego-motion planner behavior to compensate for undetected *invalid* predictions. In addition, understanding the planning performance enables setting an absolute error threshold for *invalid* predictions, ensuring the necessary prediction accuracy for safe ego-motion behavior.

A third aspect is handling *invalid* predictions when encountered during inference in real-world traffic scenarios. Selecting an *invalid* option does not necessarily trigger an emergency state. By detecting an *invalid* prediction, the AV software can adaptively adjust its strategy to prevent hazardous situations. For instance, deploying a different set of planner parameters for *invalid* prediction scenes, switching to a shorter prediction horizon, or applying deterministic approaches like reachable sets can be considered. This adaptive motion prediction and planning strategy can be likened to human drivers' natural reactions in unfamiliar scenarios, such as reducing speed.

Lastly, the full-stack integration comprises the consideration of varying input data quality and computational requirements. The first aspect must be analyzed on respective datasets and recorded data after the method is trained for this ODD. The aspect of the computational requirements is outlined in Figure 4.21. The calculation time for preprocessing and model execution over the number of objects on an exemplary hardware setup is depicted. For each bin, $30$ runs are conducted. The preprocessing comprises the graph generation between the tracked objects and the transfer of the batches to the GPU. It can be seen that the preprocessing's calculation time correlates with the number of objects. It increases from $0.74\,\mathrm{ms}$ for $n \leq 10$ objects to $10.10\,\mathrm{ms}$ for $n > 50$ objects. In contrast, the model execution does not correlate with the number of objects and stays around $60\,\mathrm{ms}$ with a standard deviation below $5\,\mathrm{ms}$ for all investigated numbers of objects. Thus, executing the objects in one batch makes the calculation time approximately constant. The usage of the VRAM on the GPU is kept below $1\,\mathrm{GB}$ for all investigated object numbers. Thus, there is no bottleneck in the storage to process that number of objects with state-of-the-art GPUs. In total, the calculation time ranges between $61\,\mathrm{ms}$ and $69\,\mathrm{ms}$ for the given hardware. An additional analysis of the calculation time on the successor generation of GPU architectures, the Nvidia A100, and an AMD EPYC 9474F CPU with $4.10\,\mathrm{GHz}$ results in an average model execution of $27\,\mathrm{ms}$ and a preprocessing time below $7\,\mathrm{ms}$ for the given range of object numbers. For real-world applications, the target latency of the software, the required update frequency from the succeeding planning module, and the available hardware resources must be considered to deploy the method with the respective calculation times.

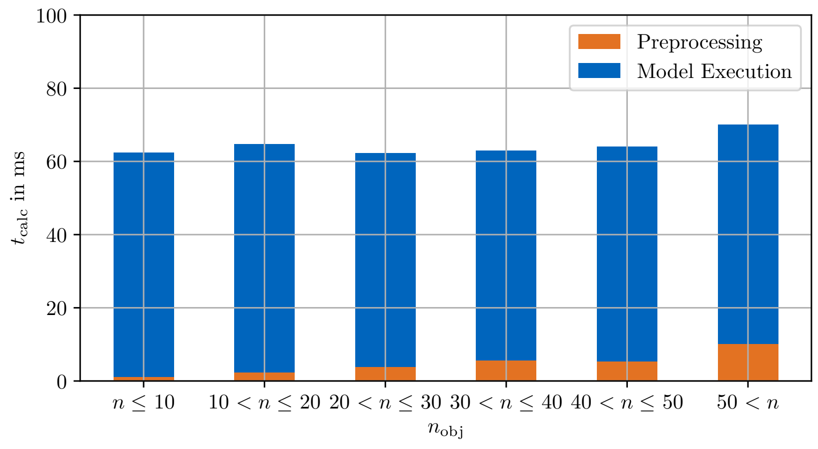


Figure 4.21: Calculation time of the object list processing and the model execution over the number of objects (CPU: Intel Xeon Gold 6148 2.40 GHz, 380 GB; GPU: Nvidia Tesla V100, 16 GB).

# 5 Discussion

In the following chapter, the presented methods contributing to the establishment of the generic framework are discussed. The discussion is based on the identified research gap (Section 2.4). The related research items are analyzed in Section 5.1. Next, the combination of the two research streams of reliability and adaptivity and their application in a specific ODD are discussed (Section 5.2). Finally, the research question, to which the research items contribute, is reviewed in Section 5.3. The chapter closes with an outlook on future work based on this research (Section 5.4).

## 5.1 Review of Research Items

The five proposed research items result from the formulation of the thesis' research question as identified topics to be investigated (Section 2.4). In the following, the research items are reviewed regarding their fulfillment by the proposed methods.

### 5.1.1 Detection Input Optimization

The first research item is formulated as follows:

**RI1** How can the detection input be improved to enable reliable application?

This investigation takes place in the scope of autonomous racing, a challenging real-world application because of high ego and relative speeds. In Section 3.2, a multi-modal sensor fusion and object tracking method is introduced, which targets this research item. Two features are proposed to improve the detection input. At first, a modular late fusion algorithm is implemented, which is able to fuse heterogeneous detection input with differences in measured features, frequency, and measurement accuracy. It allows the modular creation of multiple parallel perception pipelines and the derivation of optimized information about the surrounding objects' states. Second, an EKF with incorporated delay compensation is presented. Besides estimating the objects' states by the filter, implementing a backward-forward integration to consider delayed detection input but still output an updated object list is a key feature to reduce the overall software latency. While this feature is essential for performant high-speed racing, it is also a safety feature for public road traffic. The related results show a low-biased tracking behavior on real-world data and consistent compensation for delay in updating the object states. The research item is assessed as successfully addressed by the proposed method.

### 5.1.2 Constrained Black-Box Models

The second research item is identified as follows:

**RI2** How can black-box models be constrained to increase their reliability?

The method addressing this item is a combination of physics constraints and a NN-based scene encoding and developed for the ODD of autonomous racing (Section 3.3). The proposed method ensures a non-oscillating trajectory by superpositioning smooth base curves. The superposition additionally ensures that the trajectory stays inside the driveable area, improving the feasibility. Despite these constraints, the validation of the method shows that it is able to surpass the accuracy of the unconstrained method. The method successfully addresses the research item within the context of autonomous racing. The transfer to further ODDs is discussed in Section 3.4 with the conclusion that the transfer to a closed-road environment such as a highway is possible while the shift to urban environments requires further research effort to adjust the base curve selection to open road geometries.

### 5.1.3 Combination of Multiple Methods

The research item, which targets the aspect of hybrid prediction methods, is specified as:

**RI3** How can multiple models be combined for adaptive prediction behavior?

This research item targets the adaptivity of motion prediction methods. The item is motivated by the review of the state of the art (Subsection 2.3.5), which outlines that there is no census about a universal prediction method that fits all ODDs. Based on this review, a hybrid prediction method is proposed to enable adaptive behavior (Section 4.2). It combines multiple prediction models from different classes. The models are fused in an integrative approach to ensure computational efficiency and are scene-dependently applied to output an adjusted trajectory prediction. The validation of the method shows that each prediction model has shares of the most accurate predictions and that the combination of multiple methods is able to generate a more general and scene-adaptive prediction behavior (Subsection 4.5.2). The related ablation study (Section 4.6) offers further insights into the relation between the used prediction model to create the hybrid prediction method and the achieved accuracy. Summarising this research item, the hybrid prediction model and the feature to adaptively select the most suitable are assessed to fulfill this research item.

### 5.1.4 Evaluation of Prediction Performance

The evaluation of the prediction performance is incorporated in the fourth research item, which states:

**RI4** How can scenes be evaluated regarding the expected prediction performance?

It is the second research item that focuses on the aspect of adaptivity. Based on the hybrid prediction method, which defines the available predictors to use, a selector model is implemented to evaluate scenes based on a predefined metric. Besides the selection of the scene-dependent best predictor, the evaluation identifies the predictors' limitations. This evaluation involves detecting non-predictable, *invalid* scenes. In these scenes, it is expected that none of the existing prediction models can achieve the required accuracy. Hence, scenes are binary evaluated either as *valid*, which means that a prediction is determined, or as *invalid* if a specified prediction performance is not reached. The evaluation of scenes and their rating as *invalid* builds an essential foundation for future, edge-case-driven improvement of the prediction models. Under the assumption that the prediction performance is assessed by the specified metric of the RMSE, the research item is fulfilled by the proposed method.

### 5.1.5 Factors for Inaccurate Predictions

The last research item is focused on understanding the influencing factors of prediction errors:

**RI5** Which factors influence inaccurate predictions?

From the system application's PoV, the reasons for the inaccuracy of the predictions are of interest to handle and prevent them. To fulfill this research item, an analysis of the correlation between the semantics in a scene and the prediction performance is conducted (Section 4.7). The results show a bias toward specific object types, speed ranges, and number of objects. This knowledge is essential to base future research on and adjust the ego behavior in case these features occur during application. The analysis additionally shows the influence of the prediction horizon, which is an intermediate option to constrain the prediction. Through the analysis, the research item is addressed regarding these specific features. However, further investigations, such as details about road geometries and multi-modal variations, are required to extend the understanding of erroneous predictions.

## 5.2 Combination in a Generic Prediction Framework

Developing a reliable and adaptive prediction method is identified as a research gap. However, the proposed research items focus on one of the features respectively. Beyond the isolated development of these, the research question points to the combination of these features in one generic framework (Section 2.4). The following section focuses on this and discusses the incorporation of reliability and adaptivity in one framework. A concept of combining these features is analyzed in Subsection 5.2.1. Based on this concept, the specific integration of the methods proposed in this thesis and application in an ODD is discussed (Subsection 5.2.2).

### 5.2.1 Concept

The concept of the generic prediction framework based on the modular software stack (Figure 2.1) is depicted in Figure 5.1. The tracking is incorporated as an integral part because the handling of varying data quality of the detection input shall be covered. It is an essential prerequisite to enable motion prediction [263] and enhances the reliability. In a straightforward way, one prediction method outputs forecasted trajectories for all tracked objects based on their object history.

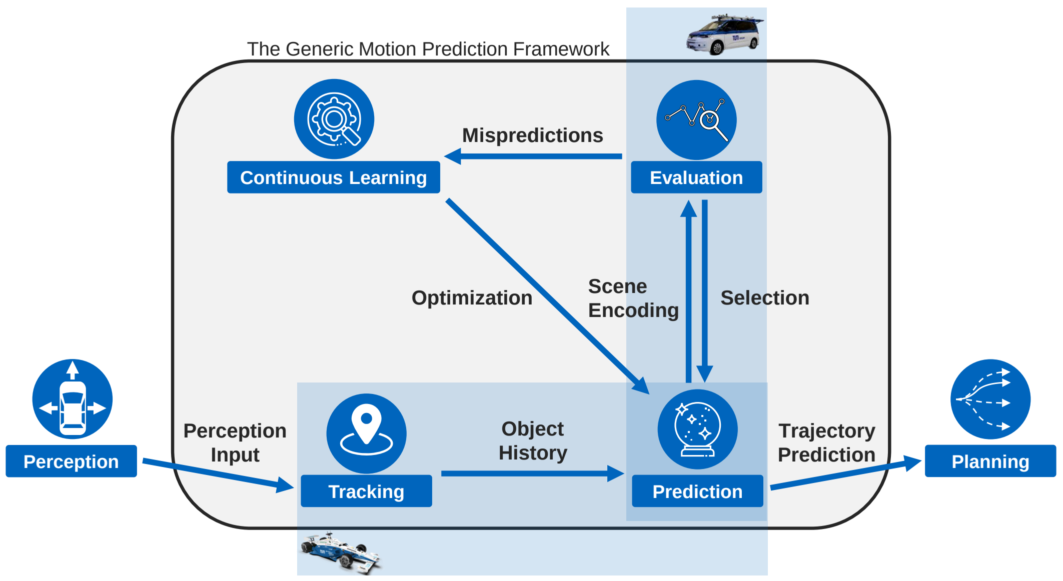


Figure 5.1: Concept of the generic framework for motion prediction in autonomous driving. The two aspects of reliability, realized in autonomous racing, and adaptivity, shown for public road traffic, are embedded into the framework.

While the concept of tracking and one fixed prediction method is suitable for specific ODDs with a limited set of scenes, adaptivity is required when it comes to more diverse scenes with varying road geometries, object types, and intentions. For this purpose, a scene evaluation and a continuous learning method are proposed. The evaluation's task is to assess the scene and available predictors and determine the most suitable prediction approach to ensure accurate forecasts. The continuous learning method is incorporated based on the assumption that the available data during development can not cover all scenes occurring during the application. The idea is that the evaluation detects scenes that would result in erroneous prediction, and the prediction models are optimized through continuous learning from these scenes. By this, the prediction models constantly improve based on detected mispredictions.

### 5.2.2 Integration of the Proposed Methods

The two research streams of this thesis for reliability in autonomous racing and adaptivity in public road traffic are embedded into the framework as symbolized by the vehicles of the AV-21 and *EDGAR* (Figure 5.1).

Since the preceding tracking module is unidirectional and does not interact with the prediction algorithms, the proposed late fusion and tracking method (Section 3.2) is integrable into the framework. It does not depend on the number of predictors or usage of the evaluation method. It is rather the case that the required input quality for the prediction algorithms must be ensured or the other way around that the prediction algorithms have to be tuned to base their output on the given tracking accuracy. The physics-constrained prediction model (Section 3.3) is also integrable into the framework. The only dependency regarding the evaluation method is the share of the encoded scene. The encoder-decoder architecture of the MixNet (Subsection 3.3.1) is suitable for this purpose.

The crucial point of these two methods for generic usage is the ODD they are developed for and the made assumptions. The specific discussion on transferring the two methods to further ODDs is conducted in Subsection 3.4.2. Regarding the tracking method, the filters to merge two objects and for out-of-track detections and the assumptions that all objects have the same driving direction are distinct limitations. However, the state estimation with delay compensation and the modular late fusion of the detection pipelines are features that are beneficial for various ODDs. Reviewing the made assumptions for the physics-constrained prediction method and their transferability to public road traffic, the main issue is choosing suitable base curves to create a superposed path prediction. In contrast to closed race tracks, the existence of junctions, multi-lane roads, and one-way routes prevent an intermediate transfer to ODDs in public road traffic. Nevertheless, the basic idea of constraining a neural network's output to given semantic geometries is valuable for public road traffic. The concept can enhance reliability by introducing constraints through external knowledge about the limits of an object's behavior and possible driving options.

The additional usage of the evaluation method does not imply requirements for the used prediction models. As shown in the conducted ablation study (Subsection 4.6.4), the selector model can be based on various encoders. In addition, the type and number of used predictors combined in the evaluation method are flexible and not limited to the proposed ones within this thesis. Regarding the usage in a specific ODD, the self-evaluation is applicable because it does not rely on ODD-specific assumptions. A crucial constraint, however, is the data availability to train and optimize the predictors and the selector model. Integrating the self-evaluation method in the framework for the usage in the ODD of autonomous racing is also realizable. The combination of one predictor and the selector model could be used to evaluate whether a scene is predictable by the model or not. This could be beneficial when the track or race format is changed, and there is not enough ODD-specific data to train the predictor on. In this case, the self-evaluation method could be used to avoid inaccurate mispredictions. The usage of multiple predictors for the racing ODD should be considered when it comes to more advanced motion patterns with multiple vehicles, less strict rules, and more complex track geometries.

To conclude the discussion, the following can be stated: The overall bottleneck for the creation of a generic framework for accurate motion prediction is the realization of adaptivity while maintaining reliability. The two investigated research streams reveal this trade-off. On the one hand, reliable methods for the ODD of autonomous racing with known object types, a closed road geometry, and defined race rules are achieved. While the reliability is achieved by including assumptions derived from ODD-specific knowledge, it implies limited adaptivity. On the other hand, an adaptive method is proposed, which selects the scene-dependent most suitable prediction method but is not proven reliable. Additionally, the predictors used in the current implementation of the evaluation method are unconstrained, which limits their reliability.

## 5.3 Review of Research Question

Based on the preceding discussions, the focus shall shift to the review of the research question, which is formulated as:

> *How can reliable and adaptive prediction methods for autonomous driving be developed and combined in a generic framework?*

It comprises two focal points. The first one is the development of reliable and adaptive prediction methods, which is detailed in the research items. To relate the proposed methods to the state of the art, Figure 5.2 reveals the graphical classification along these two axes.

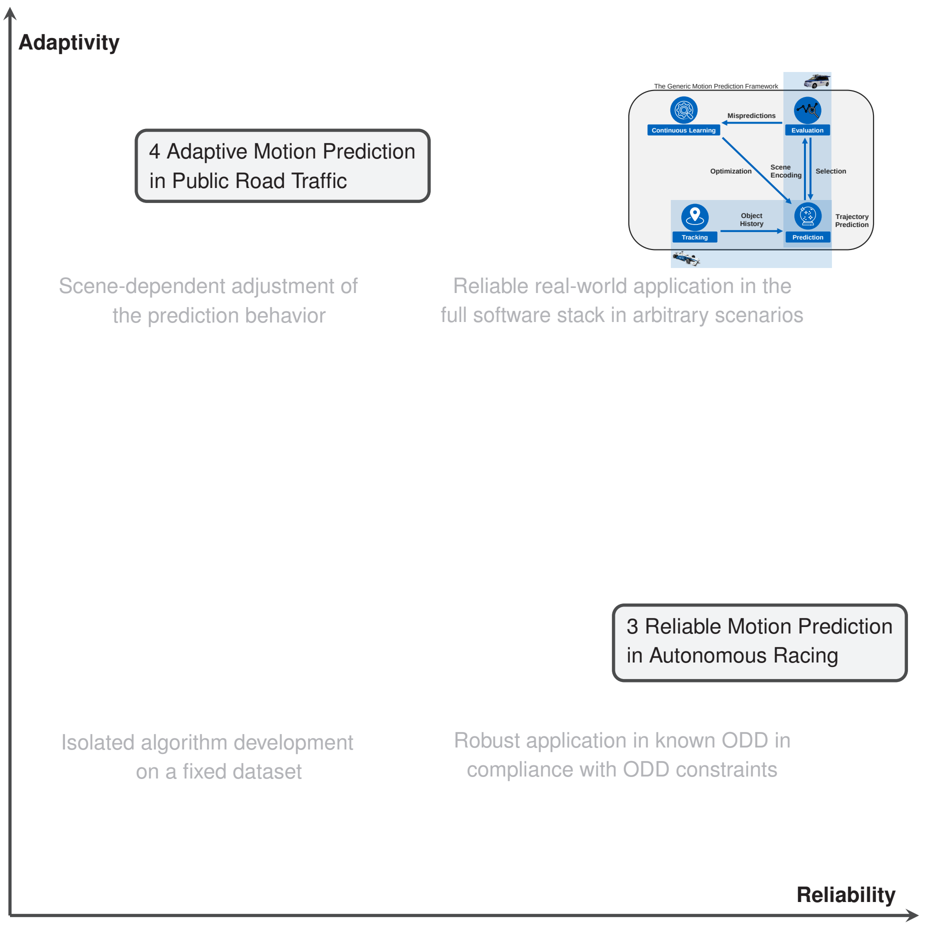

Adaptivity
4 Adaptive Motion Prediction in Public Road Traffic
The Generic Motion Prediction Framework
Continuous Learning
Mispredictions
Evaluation
Optimization
Scene Encoding
Selection
Object History
Tracking
Prediction
Trajectory Prediction
Scene-dependent adjustment of the prediction behavior
Reliable real-world application in the full software stack in arbitrary scenarios
3 Reliable Motion Prediction in Autonomous Racing
Isolated algorithm development on a fixed dataset
Robust application in known ODD in compliance with ODD constraints
Reliability

Figure 5.2: Categorization of the thesis' research along the two axes of reliability and adaptivity.

The proposed reliable prediction method is assessed to be placed on the far right along the reliability axis (Chapter 3). Both modules of tracking and prediction are integrated into the full software stack of TUM autonomous motorsport, satisfying the computational requirements to run on the target hardware with the respective run time [264]. Beyond the presented results on real-world data (Subsection 3.2.4) with low-biased tracking errors and successful compensation of the perception delay, the reliability of the tracking method is confirmed through its usage at the events of the AC@CES in Las Vegas from 2022 – 2024. The method is deployed with continuously increasing speeds, fewer constraints regarding overtaking maneuvers, and varying detection pipelines used since the team's software stack constantly grows. It was an essential component for the fastest autonomous overtakes at the time of its publication at $270\,\mathrm{km\,h^{-1}}$. The constrained prediction method's reliability is shown by the presented results on interactive high-speed data featuring quality guarantees besides accurate prediction behavior (Subsection 3.3.5). The real-world application of the method was not achieved because of the cancellation of the multi-vehicle race at the inaugural event of the IAC in Indianapolis in October 2021. For the later event at the AC@CES in Las Vegas in January 2022, the method could not be deployed due to missing data for the specific race track, which differs from the one in Indianapolis by its size and shape (Figure 3.2). The rail-based prediction, which is presented in Section 3.3 as a benchmark, was used. However, MixNet's superior performance on the multi-vehicle HiL and on the simulation race data justifies the proposed classification of the framework. In terms of adaptivity, the transfer to further ODDs (Subsection 3.4.2) and the integration in the concept of the generic framework are analyzed (Subsection 5.2.2). While fundamentals of the methods such as the delay compensation and EKF-based tracking or the concept to superpose base curves are adaptive, the development has its focus on the autonomous racing ODD with distinct assumptions. Thus, the classification for adaptivity is in the lower part of the plot.

The proposed self-evaluation method for adaptive prediction behavior is classified at the top along the vertical axis. The grading can be justified by the validated ability to adaptively select the scene-dependent most suitable prediction method and the idea of an integral hybrid prediction composition. The evaluation is realized by semantic encoding to create scenario understanding, which selects the most suitable prediction approach for the current scene-dependent circumstances. A hybrid prediction model provides diverse prediction approaches. The ODD of public road traffic is chosen because varying behavior of object types, road geometry, and motion patterns are expected, which all require adaptive prediction. In addition, the self-evaluation method is capable of detecting scenes in which high prediction errors occur. The feature to detect the algorithmic limitations and the conducted study about the reasons for prediction errors (Section 4.7) can be used to adjust the prediction behavior dynamically. Thus, the respective scenes can be handled, e. g through an adjustment of the prediction horizon. Regarding reliability, the method does not fulfill specific ODD constraints, nor is it applied in a full software stack on real-world data. A discussion to realize this is given in Subsection 4.8.3. In summary, low reliability is assessed.

The second focal point of the research question is the integration of the methods into one generic framework. A concept to combine the given methods is discussed (Section 5.2), which fulfills all requirements of reliability and adaptivity, defined in Section 2.1, justifying its placement at the top right of the graphical classification (Figure 5.2). Varying input data quality and latency is handled by the tracking module. A constrained prediction method can ensure compliance with the ODD. An evaluation can switch between several predictors and detect algorithmic limitations. Additionally, continuous learning is proposed for extended adaptivity to optimize the predictors based on mispredictions. The crucial point of this concept is its validation, including real-world application in a specific ODD. The two specific issues of the method's extended adaptivity and validation are discussed as an outlook of this work (Subsection 5.4.2 and 5.4.3).

In summary, the research question can be assessed as fulfilled in terms of developing reliable and adaptive methods. Regarding the combination of these methods, a fulfillment on a conceptual level is given, which must be validated through its implementation.

## 5.4 Outlook

In the following, an outlook on future research directions shall be given. At first, developing a more comprehensive evaluation metric, a central element in assessing scenarios, is discussed (Subsection 5.4.1). In Subsection 5.4.2, continuous learning to expand the adaption capabilities and to complete the generic prediction framework is analyzed. Lastly, the validation of the discussed generic prediction framework is outlined (Subsection 5.4.3).

### 5.4.1 Evaluation Metric

The first future research direction is a comprehensive metric to quantify the performance of motion prediction methods. Within the scope of this work, metrics from the state of the art of mean Euclidean distance errors are used to quantify the methods' accuracy. The criticism of the existing metrics is two-fold. At first, current metrics are not able to evaluate the performance of prediction methods for the integration in a full AV software stack. Especially, the overall driving performance and behavior of the ego-motion planning based on the motion prediction method can not be evaluated with current metrics [318, 319]. Aspects of temporal consistency and estimation of uncertainty must be considered. While the latter aspects can be quantified through the NLL, the quantification of the former aspect, temporal consistency, needs to be investigated. It focuses on the deviations of predictions of the same object in two successive scenes. It is required to allow stable ego-motion planning without abrupt changes. The proposed self-evaluation benefits from a metric that considers the impact on the ego-motion planner in advance. In contrast to the current assumption that high Euclidean prediction errors negatively impact ego-motion behavior, the evaluation would be more efficient in preventing critical situations if the metric reflects the full impact of the prediction on the driving performance.

The second point of criticism is the gap between datasets and real-world scenarios [254]. The metrics do not consider the robustness of the algorithms against noisy observation. Besides the gap in data quality, the prediction behavior must be considered when facing arbitrary object behavior during real-world application, which the dataset does not contain. In the scope of this thesis, the qualitative rating of the two features of reliability and adaptivity is used to cover these aspects and to get a more comprehensive evaluation of a prediction method's behavior. However, quantifying these features and their balancing in combination with a method's accuracy is subject to future research. A holistic metric to evaluate motion prediction methods, which covers and quantifies the discussed aspects, is desired.

### 5.4.2 Extended Adaptivity

The second aspect of future work is further progress regarding adaptivity. Besides a scene evaluation, continuous learning to constantly improve the predictors is proposed as part of a generic prediction framework (Figure 5.1). It is expected that due to deteriorated input data, abnormal object behavior, or to continuously extend the ODD, new scenes occur, which the framework has to cope with. To apply continuous learning to learn from new occurring scenes, several points of investigation exist.

The first issue is the training strategy. The amount of required data to retrain the algorithms has to be determined. Online learning is possible after just one or a few samples up to the collection of samples by a fleet and the creation of an edge case dataset. It must be analyzed how the amount of data to retrain the methods influences the overall prediction accuracy compared to the ability to adapt to edge cases. Besides the amount of data, their imbalance is a crucial aspect. Under the assumption that the scene evaluation specifically detects scenes that the predictors are not capable of handling, a data bias emerges. In combination with aggressive learning strategies, a loss of generalizability is expected. The blend of edge case data with training data is a possible remedy but could lead to missing adaptation to the edge case data.

A second point of investigation is the method's scalability. It must be analyzed how many new ODDs and edge cases can be covered with the learning process. While the process of offline training, ODD test rides, identification of edge cases, and retraining might be possible once, an iteration through multiple ODDs is questionable with existing methods in the field of artificial intelligence.

Third, the missing stability of current state-of-the-art methods hinders the usage. In the scope of AVs, the required level of safety does not allow an unstable behavior. Particular data augmentation strategies, self-supervised, and one-shot learning to realize continuous learning are the subject of ongoing research. These advances must be analyzed to realize the missing piece of the framework.

Lastly, with a focus on automotive safety standards, new safeguarding methods are required because current methods are not able to consider adaptive behavior [320]. The potentially unsafe behavior resulting from continuous learning cannot be caught with the existing methods. Thus, there is a requirement for new safety standards to be developed to deploy these kinds of adaptive methods.

### 5.4.3 Validation of the Framework

Following the discussion about a concept for a generic prediction framework and the integration of the proposed methods (Subsection 5.2.2), its validation is a central point of interest for future research. Besides the interaction between the different parts of the framework, its deployment allows further insights into its behavior because of the dependencies on the adjacent modules of perception and planning and the necessity of handling real-world input data.

The usage in the racing ODD should comprise the usage of one predictor and the evaluation method in the form of a supervisor, which distinguishes between *valid* and *invalid* scenes. As a first step, data from past races can be used and complemented by simulation data for diverse object behavior causing *invalid* scenes to validate the evaluation capabilities.

To achieve real-world application with *EDGAR*, data of the target ODD (Figure 5.3) are an essential prerequisite for the usage of pattern-based predictors and the self-evaluation method. Prior training on public datasets [73, 81, 137] is recommended to reduce the efforts in data mining. With the knowledge of the specific ODD, it should be reviewed which predictors are deployed to optimize the calculation efforts. In addition, the specific tuning of the planning algorithm and the handling of *invalid* predictions must be conducted as discussed in Subsection 4.8.3. The identified limitations of the two methods of tracking and physics-constrained prediction if used in an urban ODD must be considered, too (Subsection 3.4.2). The made assumptions to achieve reliable behavior must be adjusted to the new ODD.

Figure 5.3: TUM's research vehicle *EDGAR* in the city of Munich.

# 6 Conclusion

This thesis investigates the development of reliable and adaptive prediction methods and their combination in a generic framework for motion prediction in autonomous driving. The initial motivation is the proficient human driver who has anticipatory driving skills based on driving experience to solve interactive traffic scenarios. The transfer of these skills into the AV software stacks requires implementing methods for motion prediction, which features scenario understanding to handle diverse scenarios and to provide holistic prediction behavior.

The survey of the state of the art (Chapter 2) reveals that the field of motion prediction is actively researched and accurate methods exist. A review of these methods, however, identifies their shortcomings. None of them reliably predicts the motion of surrounding vehicles and is capable of adapting the prediction behavior to diverse scenes. This research gap is investigated by five research items that focus on reliable or adaptive prediction methods, contributing to the formulated research question to establish a generic prediction framework.

The feature of reliability is investigated in the ODD of autonomous racing, which offers a full-stack application at high speeds, dynamic object behavior, and interaction between the opponents. A finding from this application is that the input data received from the perception must be optimized, and consistent object tracks are crucial to enable profound motion prediction. To tackle this aspect, a modular multi-modal late fusion and object-tracking method with delay compensation is developed (Section 3.2). Its quantitative validation on real-world data and achievement of the fastest autonomous overtake at the time of its publication at the AC@CES 2022 prove the reliability. Another essential factor for reliable prediction is the guarantee of quality constraints. Performant ego-motion planning requires non-oscillating trajectories and low prediction errors. Thus, a model is demanded that ensures these constraints and offers profound scene encoding to achieve accurate prediction behavior. The trade-off is solved by proposing a physics-constrained deep neural motion prediction method, which achieves both a low prediction error and the assurance of quality guarantees (Section 3.3).

A method for adaptive prediction behavior is researched within the diverse ODD of public road traffic (Chapter 4). Adaptivity is achieved by a hybrid prediction model and a self-evaluation of the present scene based on semantic information. The hybrid prediction model combines trajectory prediction approaches from multiple classes to cover various motion patterns. The evaluation is realized by a selector model, which determines based on a predefined metric, the best out of the available prediction approaches for a given scene. In addition, the evaluation considers the case that no model can accurately predict the scene and high prediction errors are expected. By this, an adaptive prediction behavior is realized and scene-dependent algorithmic limitations are detected. The method's validation, conducted on the comprehensive scenario library *CommonRoad*, shows a decrease of the Euclidean prediction error by $81.0\,\%$ and a $90.8\,\%$ reduction in mispredictions compared to the benchmark model. The results additionally reveal the benefits of using different prediction classes that complement each other. Besides, the variation of the hybrid network composition indicates that tailored configurations for specific use cases can be realized.

The discussion (Chapter 5) highlights that the proposed methods for autonomous racing and autonomous driving in public road traffic successfully address the five proposed research items. Focussing on the overall research question to achieve a generic prediction framework, it is discussed how the research streams of reliability and adaptivity can be combined. A concept is discussed, which needs to be implemented and applied in order to validate it. In addition, the discussion reveals ODD-specific assumptions that must be solved and outlines how to realize a merge of the presented methods to be used in one ODD. A central finding of the discussion is the trade-off between reliability by incorporating additional constraints and adaptivity through the adjustment of the prediction behavior. The discussion closes with naming future research directions contributing to the realization of the generic framework. These are the definition of a comprehensive metric to evaluate a model's behavior and its impact on the overall driving performance and the extension of adaptivity by methods for continuous learning. Besides that, the validation of the generic framework is named as a future research direction.

# List of Figures

# List of Tables

# Bibliography

[1] T. Litman, „Autonomous vehicle implementation predictions,“ Victoria Transport Policy Institute, Victoria, BC, 2017.

[2] D. J. Fagnant and K. Kockelman, „Preparing a nation for autonomous vehicles: opportunities, barriers and policy recommendations,“ *Transportation Research Part A: Policy and Practice*, vol. 77, pp. 167–181, 2015, DOI: 10.1016/j.tra.2015.04.003.

[3] U.S. Access Board. „Inclusive Design of Autonomous Vehicles: A Public Dialogue – Summary Report,“ [Online]. Available: https://www.access-board.gov/files/usab-av-forum-summary-report.pdf.

[4] Ó. Silva, R. Cordera, E. González-González and S. Nogués, „Environmental impacts of autonomous vehicles: A review of the scientific literature,“ *Science of The Total Environment*, vol. 830, p. 154615, 2022, DOI: 10.1016/j.scitotenv.2022.154615.

[5] A. F. Williams and O. Carsten, „Driver age and crash involvement,“ *American Journal of Public Health*, vol. 79, no. 3, pp. 326–327, 1989, DOI: 10.2105/ajph.79.3.326.

[6] M. A. Rahman, M. M. Hossain, E. Mitran and X. Sun, „Understanding the contributing factors to young driver crashes: A comparison of crash profiles of three age groups,“ *Transportation Engineering*, vol. 5, p. 100076, 2021, DOI: 10.1016/j.treng.2021.100076.

[7] C. C. McDonald, A. E. Curry, V. Kandadai, M. S. Sommers and F. K. Winston, „Comparison of teen and adult driver crash scenarios in a nationally representative sample of serious crashes,“ *Accident Analysis & Prevention*, vol. 72, pp. 302–308, 2014.

[8] T. Seacrist, E. C. Douglas, E. Huang, J. Megariotis, A. Prabahar, et al., „Analysis of near crashes among teen, young adult, and experienced adult drivers using the SHRP2 naturalistic driving study,“ *Traffic Injury Prevention*, vol. 19, no. 1, pp. 89–96, 2018, DOI: 10.1080/15389588.2017.1415433.

[9] A. J. McKnight and A. S. McKnight, „Young novice drivers: careless or clueless?,“ *Accident Analysis & Prevention*, vol. 35, no. 6, pp. 921–925, 2003, DOI: 10.1016/s0001-4575(02)00100-8.

[10] S. E. Lee, S. G. Klauer, E. C. B. Olsen, B. G. Simons-Morton, T. A. Dingus, et al., „Detection of road hazards by novice teen and experienced adult drivers,“ *Transportation Research Record*, vol. 2078, no. 1, pp. 26–32, 2008, DOI: 10.3141/2078-04.

[11] Q. Li, H. Yao and X. Li, „A matched case-control method to model car-following safety,“ *Transportmetrica A: transport science*, vol. 19, no. 3, p. 2055198, 2023, DOI: 10.1080/23249935.2022.2055198.

[12] N. M. Negash and J. Yang, „Driver Behavior Modeling Toward Autonomous Vehicles: Comprehensive Review,“ *IEEE Access*, vol. 11, pp. 22788–22821, 2023, DOI: 10.1109/ACCESS.2023.3249144.

[13] W. Münst, „Prediction of Driver Behavior and Decision Strategies for Autonomous Driving: Using Machine Learning and Decision Theory,“ Hagen, 2020, DOI: 10.18445/20201122-144857-0.

[14] W. Wang, L. Wang, C. Zhang, C. Liu, L. Sun, et al., „Social interactions for autonomous driving: A review and perspectives,“ *Foundations and Trends in Robotics*, vol. 10, no. 3, pp. 198–376, 2022, ISBN: 9781638281290.

[15] S. Ettinger, S. Cheng, B. Caine, C. Liu, H. Zhao, et al., „Large Scale Interactive Motion Forecasting for Autonomous Driving: The Waymo Open Motion Dataset,“ in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 9710–9719, DOI: 10.1109/ICCV48922.2021.00957.

[16] R. Qian, X. Lai and X. Li, „3D object detection for autonomous driving: A survey,“ *Pattern Recognition*, vol. 130, p. 108796, 2022, DOI: 10.1016/j.patcog.2022.108796.

[17] European Commission, „Commission Implementing Regulation (EU) 2022/1426,“ 2022.

[18] A. Jain, L. Del Pero, H. Grimmett and P. Ondruska, „Autonomy 2.0: Why is self-driving always 5 years away?,“ *arXiv preprint arXiv:2107.08142*, 2021.

[19] D. Bogdoll, M. Nitsche and J. M. Zöllner, „Anomaly detection in autonomous driving: A survey,“ in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 4488–4499, DOI: 10.1109/CVPRW56347.2022.00495.

[20] W. Shao, B. Li, W. Yu, J. Xu and H. Wang, „When Is It Likely to Fail? Performance Monitor for Black-Box Trajectory Prediction Model,“ *IEEE Transactions on Automation Science and Engineering*, pp. 1–15, 2024, DOI: 10.1109/TASE.2024.3356070.

[21] K. Xu, X. Xiao, J. Miao and Q. Luo, „Data driven prediction architecture for autonomous driving and its application on apollo platform,“ in *2020 IEEE Intelligent Vehicles Symposium (IV)*, 2020, pp. 175–181, DOI: 10.1109/IV47402.2020.9304810.

[22] P. Karle and M. Lienkamp. „*Autonomous Driving Software Engineering*,“ Institute of Automotive Technology, Technical University of Munich. 2021. DOI: 10.13140/RG.2.2.10596.76165.

[23] SAE International On-Road Automated Driving Committee, „Vehicles J3016 201806 Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor – Revised,“ DOI: 10.4271/J3016_202104.

[24] M. Maurer, „EMS-vision: knowledge representation for flexible automation of land vehicles,“ in *Proceedings of the IEEE Intelligent Vehicles Symposium 2000*, 2000, pp. 575–580, DOI: 10.1109/IVS.2000.898409.

[25] S. Geyer, M. Baltzer, B. Franz, S. Hakuli, M. Kauer, et al., „Concept and development of a unified ontology for generating test and use-case catalogues for assisted and automated vehicle guidance,“ *IET Intelligent Transport Systems*, vol. 8, no. 3, pp. 183–189, 2014, DOI: 10.1049/iet-its.2012.0188.

[26] S. Ulbrich, T. Menzel, A. Reschka, F. Schuldt and M. Maurer, „Defining and substantiating the terms scene, situation, and scenario for automated driving,“ in *2015 IEEE 18th international conference on intelligent transportation systems*, 2015, pp. 982–988, DOI: 10.1109/ITSC.2015.164.

[27] C. Luo, X. Yang and A. Yuille, „Exploring Simple 3D Multi-Object Tracking for Autonomous Driving,“ in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 10488–10497, DOI: 10.1109/ICCV48922.2021.01032.

[28] S. Mozaffari, O. Y. Al-Jarrah, M. Dianati, P. Jennings and A. Mouzakitis, „Deep Learning-Based Vehicle Behavior Prediction for Autonomous Driving Applications: A Review,“ *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 1, pp. 33–47, 2022, DOI: 10.1109/TITS.2020.3012034.

[29] Y. Huang, J. Du, Z. Yang, Z. Zhou, L. Zhang, et al., „A Survey on Trajectory-Prediction Methods for Autonomous Driving,“ *IEEE Transactions on Intelligent Vehicles*, vol. 7, no. 3, pp. 652–674, 2022, DOI: 10.1109/TIV.2022.3167103.

[30] C. Katrakazas, M. Quddus, W.-H. Chen and L. Deka, „Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions,“ *Transportation Research Part C: Emerging Technologies*, vol. 60, pp. 416–442, 2015, DOI: 10.1016/j.trc.2015.09.011.

[31] J. Wei, J. M. Snider, J. Kim, J. M. Dolan, R. Rajkumar, et al., „Towards a viable autonomous driving research platform,“ in *2013 IEEE Intelligent Vehicles Symposium (IV)*, 2013, pp. 763–770, DOI: 10.1109/IVS.2013.6629559.

[32] M. Buechel, M. Schellmann, H. Rosier, T. Kessler and A. Knoll. „*Fortuna: Presenting the 5G-Connected Automated Vehicle Prototype of the Project PROVIDENTIA*,“ 2019. DOI: 10.13140/RG.2.2.24402.91842.

[33] T. Kessler, J. Bernhard, M. Buechel, K. Esterle, P. Hart, et al., „Bridging the Gap between Open Source Software and Vehicle Hardware for Autonomous Driving,“ in *2019 IEEE Intelligent Vehicles Symposium (IV)*, 2019, pp. 1612–1619, DOI: 10.1109/IVS.2019.8813784.

[34] J. Haselberger, M. Pelzer, B. Schick and S. Müller, „JUPITER – ROS based Vehicle Platform for Autonomous Driving Research,“ in *2022 IEEE International Symposium on Robotic and Sensors Environments (ROSE)*, 2022, pp. 1–8, DOI: 10.1109/ROSE56499.2022.9977434.

[35] G. Mehr, P. Ghorai, C. Zhang, A. Nayak, D. Patel, et al., „X-CAR: An Experimental Vehicle Platform for Connected Autonomy Research Powered by CARMA,“ *IEEE Intelligent Transportation Systems Magazine*, 2022, DOI: 10.1109/MITS.2022.3168801.

[36] J. Fayyad, M. A. Jaradat, D. Gruyer and H. Najjaran, „Deep learning sensor fusion for autonomous vehicle perception and localization: A review,“ *Sensors*, vol. 20, no. 15, p. 4220, 2020, DOI: 10.3390/s20154220.

[37] P. Kolar, P. Benavidez and M. Jamshidi, „Survey of Datafusion Techniques for Laser and Vision Based Sensor Integration for Autonomous Navigation,“ *Sensors*, vol. 20, no. 8, 2020, DOI: 10.3390/s20082180.

[38] Y. Wang, S. Chen, L. Huang, R. Ge, Y. Hu, et al., „1st Place Solutions for Waymo Open Dataset Challenges–2D and 3D Tracking,“ *arXiv preprint arXiv:2006.15506*, 2020.

[39] F. Sauerbeck, S. Huch, F. Fent, P. Karle, D. Kulmer, et al., „Learn to See Fast: Lessons Learned From Autonomous Racing on How to Develop Perception Systems,“ *IEEE Access*, vol. 11, pp. 44034–44050, 2023, DOI: 10.1109/ACCESS.2023.3272750.

[40] D. J. Yeong, G. Velasco-Hernandez, J. Barry and J. Walsh, „Sensor and Sensor Fusion Technology in Autonomous Vehicles: A Review,“ *Sensors*, vol. 21, no. 6, 2021, DOI: 10.3390/s21062140.

[41] L. Xiao, B. Dai, D. Liu, T. Hu and T. Wu, „CRF-based road detection with multi-sensor fusion,“ in *2015 IEEE intelligent vehicles symposium (IV)*, 2015, pp. 192–198, DOI: 10.1109/IVS.2015.7225685.

[42] L. Xiao, R. Wang, B. Dai, Y. Fang, D. Liu, et al., „Hybrid conditional random field based camera-LIDAR fusion for road detection,“ *Information Sciences*, vol. 432, pp. 543–558, 2018, DOI: 10.1016/j.ins.2017.04.048.

[43] I. Sobh, L. Amin, S. Abdelkarim, K. Elmadawy, M. Saeed, et al., „End-to-end multi-modal sensors fusion system for urban automated driving,“ in *Advances in Neural Information Processing Systems (NeurIPS) Workshop*, 2018.

[44] D. Xu, D. Anguelov and A. Jain, „PointFusion: Deep Sensor Fusion for 3D Bounding Box Estimation,“ *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 244–253, 2018, DOI: 10.1109/CVPR.2018.00033.

[45] C. R. Qi, W. Liu, C. Wu, H. Su and L. J. Guibas, „Frustum PointNets for 3D Object Detection from RGB-D Data,“ in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 918–927, DOI: 10.1109/CVPR.2018.00102.

[46] M. Liang, B. Yang, Y. Chen, R. Hu and R. Urtasun, „Multi-Task Multi-Sensor Fusion for 3D Object Detection,“ in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, DOI: 10.1109/CVPR.2019.00752.

[47] V. A. Sindagi, Y. Zhou and O. Tuzel, „MVX-Net: Multimodal VoxelNet for 3D Object Detection," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 7276–7282, DOI: 10.1109/ICRA.2019.8794195.

[48] J. Ku, M. Mozifian, J. Lee, A. Harakeh and S. L. Waslander, „Joint 3D Proposal Generation and Object Detection from View Aggregation," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 1–8, DOI: 10.1109/IROS.2018.8594049.

[49] M. Liang, B. Yang, S. Wang and R. Urtasun, „Deep Continuous Fusion for Multi-Sensor 3D Object Detection," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, DOI: 10.1007/978-3-030-01270-0_39.

[50] S. Chadwick, W. Maddern and P. Newman, „Distant Vehicle Detection Using Radar and Vision," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 8311–8317, DOI: 10.1109/ICRA.2019.8794312.

[51] K. Kowol, M. Rottmann, S. Bracke and H. Gottschalk, „YOdar: Uncertainty-based Sensor Fusion for Vehicle Detection with Camera and Radar Sensors," in *Proceedings of the 13th International Conference on Agents and Artificial Intelligence*, 2021, pp. 177–186, ISBN: 978-989-758-484-8. DOI: 10.5220/0010239301770186.

[52] F. Nobis, E. Shafiei, P. Karle, J. Betz and M. Lienkamp, „Radar Voxel Fusion for 3D Object Detection," *Applied Sciences*, vol. 11, no. 12, 2021, DOI: 10.3390/app11125598.

[53] F. Nobis, M. Geisslinger, M. Weber, J. Betz and M. Lienkamp, „A Deep Learning-based Radar and Camera Sensor Fusion Architecture for Object Detection," *Sensor Data Fusion: Trends, Solutions, Applications (SDF)*, pp. 1–7, 2019, DOI: 10.1109/SDF.2019.8916629.

[54] J. Nie, J. Yan, H. Yin, L. Ren and Q. Meng, „A Multimodality Fusion Deep Neural Network and Safety Test Strategy for Intelligent Vehicles," *IEEE Transactions on Intelligent Vehicles*, vol. 6, no. 2, pp. 310–322, 2021, DOI: 10.1109/TIV.2020.3027319.

[55] F. Leon and M. Gavrilescu, „A Review of Tracking and Trajectory Prediction Methods for Autonomous Driving," *Mathematics*, vol. 9, no. 6, 2021, DOI: 10.3390/math9060660.

[56] D. M. Jiménez-Bravo, Álvaro Lozano Murciego, André Sales Mendes, Héctor Sánchez San Blás and Javier Bajo, „Multi-object tracking in traffic environments: A systematic literature review," *Neurocomputing*, vol. 494, pp. 43–55, 2022, DOI: 10.1016/j.neucom.2022.04.087.

[57] Chahinez Ounoughi and Sadok Ben Yahia, „Data fusion for ITS: A systematic literature review," *Information Fusion*, vol. 89, pp. 267–291, 2023, DOI: 10.1016/j.inffus.2022.08.016.

[58] S. Jagannathan, M. Mody, J. Jones, P. Swami and D. Poddar, „Multi-sensor fusion for Automated Driving: Selecting model and optimizing on Embedded platform," *Electronic Imaging*, vol. 2018, no. 17, pp. 256–1, 2018, DOI: 10.2352/ISSN.2470-1173.2018.17.AVM-256.

[59] R. E. Kalman, „A New Approach to Linear Filtering and Prediction Problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960, DOI: 10.1115/1.3662552.

[60] H. W. Sorenson, *Kalman filtering: Theory and application*, IEEE Computer Society Press, 1985, ISBN: 0780304217.

[61] A. H. Jazwinski, *Stochastic processes and filtering theory*, (Mathematics in science and engineering). vol. 64, 1st ed., San Diego, Acad. Press, 1970, ISBN: 0123815509.

[62] E. Wan and R. Van Der Merwe, „The unscented Kalman filter for nonlinear estimation," *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium*, pp. 153–158, 2000.

[63] P. Del Moral, „Nonlinear filtering: Interacting particle resolution,“ *Comptes Rendus de l’Académie des Sciences-Series I-Mathematics*, vol. 325, no. 6, pp. 653–658, 1997, DOI: 10.1016/S0764-4442(97)84778-7.

[64] K. Granstrom, M. Baum and S. Reuter, „Extended Object Tracking: Introduction, Overview and Applications,“ *arXiv preprint arXiv:1604.00970*, 2016.

[65] R. O. Chavez-Garcia and O. Aycard, „Multiple Sensor Fusion and Classification for Moving Object Detection and Tracking,“ *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 2, pp. 525–534, 2016, DOI: 10.1109/TITS.2015.2479925.

[66] F. Garcia, D. Martin, A. de La Escalera and J. M. Armingol, „Sensor fusion methodology for vehicle detection,“ *IEEE Intelligent Transportation Systems Magazine*, vol. 9, no. 1, pp. 123–133, 2017, DOI: 10.1109/MITS.2016.2620398.

[67] W. Farag, „Kalman-filter-based sensor fusion applied to road-objects detection and tracking for autonomous vehicles,“ *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, vol. 235, no. 7, pp. 1125–1138, 2021, DOI: 10.1177/0959651820975523.

[68] E. Andert and A. Shrivastava, „Accurate Cooperative Sensor Fusion by Parameterized Covariance Generation for Sensing and Localization Pipelines in CAVs,“ in *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, 2022, pp. 3595–3602, DOI: 10.1109/ITSC55140.2022.9922598.

[69] M. P. Muresan, I. Giosan and S. Nedevschi, „Stabilization and Validation of 3D Object Position Using Multimodal Sensor Fusion and Semantic Segmentation,“ *Sensors*, vol. 20, no. 4, 2020, DOI: 10.3390/s20041110.

[70] S. Jahromi, Babak, T. Tulabandhula and S. Cetin, „Real-Time Hybrid Multi-Sensor Fusion Framework for Perception in Autonomous Vehicles,“ *Sensors*, vol. 19, no. 20, 2019, DOI: 10.3390/s19204357.

[71] S. Verma, Y. H. Eng, H. X. Kong, H. Andersen, M. Meghjani, et al., „Vehicle Detection, Tracking and Behavior Analysis in Urban Driving Environments Using Road Context,“ in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 1413–1420, DOI: 10.1109/ICRA.2018.8460951.

[72] R. Kunjumon and G. S. Sangeetha Gopan, „Sensor Fusion of Camera and Lidar Using Kalman Filter,“ *Intelligent Systems*, pp. 327–343, 2021, ISBN: 978-981-16-2248-9.

[73] A. Geiger, P. Lenz and R. Urtasun, „Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite,“ in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 3354–3361, ISBN: 978-1-4673-1226-4. DOI: 10.1109/CVPR.2012.6248074.

[74] S. K. Pal, A. Pramanik, J. Maiti and P. Mitra, „Deep learning in multi-object detection and tracking: state of the art,“ *Applied Intelligence*, vol. 51, no. 9, pp. 6400–6429, 2021, DOI: 10.1007/s10489-021-02293-7.

[75] Wenhan Luo, Junliang Xing, Anton Milan, Xiaoqin Zhang, Wei Liu, et al., „Multiple object tracking: A literature review,“ *Artificial Intelligence*, vol. 293, p. 103448, 2021, DOI: 10.1016/j.artint.2020.103448.

[76] M. J. Shafiee, A. Jeddi, A. Nazemi, P. Fieguth and A. Wong, „Deep Neural Network Perception Models and Robust Autonomous Driving Systems: Practical Solutions for Mitigation and Improvement,“ *IEEE Signal Processing Magazine*, vol. 38, no. 1, pp. 22–30, 2021, DOI: 10.1109/MSP.2020.2982820.

[77] A. Rangesh and M. M. Trivedi, „No Blind Spots: Full-Surround Multi-Object Tracking for Autonomous Vehicles Using Cameras and LiDARs,“ *IEEE Transactions on Intelligent Vehicles*, vol. 4, no. 4, pp. 588–599, 2019, DOI: 10.1109/TIV.2019.2938110.

[78] H. W. Kuhn, „The Hungarian method for the assignment problem,“ *Naval Research Logistics Quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955, DOI: 10.1002/nav.3800020109.

[79] P. C. Mahalanobis, „On the generalized distance in statistics,“ *Proceedings of the National Institute of Sciences (Calcutta)*, vol. 2, pp. 49–55, 1936.

[80] R. de Maesschalck, D. Jouan-Rimbaud and D.L. Massart, „The Mahalanobis distance,“ *Chemometrics and Intelligent Laboratory Systems*, vol. 50, no. 1, pp. 1–18, 2000, DOI: 10.1016/S0169-7439(99) 00047-7.

[81] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, et al., „Nuscenes: A multimodal dataset for autonomous driving,“ in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11621–11631, DOI: 10.1109/CVPR42600.2020.01164.

[82] H.-K. Chiu, J. Li, R. Ambruş and J. Bohg, „Probabilistic 3D Multi-Modal, Multi-Object Tracking for Autonomous Driving,“ in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 14227–14233, DOI: 10.1109/ICRA48506.2021.9561754.

[83] X. Weng, J. Wang, D. Held and K. Kitani, „3D Multi-Object Tracking: A Baseline and New Evaluation Metrics,“ in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 10359–10366, DOI: 10.1109/IROS45743.2020.9341164.

[84] M.-Q. Dao and V. Frémont, „A Two-Stage Data Association Approach for 3D Multi-Object Tracking,“ *Sensors*, vol. 21, no. 9, 2021, DOI: 10.3390/s21092894.

[85] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, et al., „Scalability in perception for autonomous driving: Waymo open dataset,“ in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 2446–2454, DOI: 10.1109/CVPR42600.2020.00252.

[86] P. Karle, M. Geisslinger, J. Betz and M. Lienkamp, „Scenario Understanding and Motion Prediction for Autonomous Vehicles —- Review and Comparison,“ *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 10, pp. 16962–16982, 2022, DOI: 10.1109/TITS.2022.3156011.

[87] S. Lefèvre, D. Vasquez and C. Laugier, „A survey on motion prediction and risk assessment for intelligent vehicles,“ *Robomech Journal*, vol. 1, no. 1, pp. 1–14, 2014, DOI: 10.1186/s40648-014-0001-z.

[88] K. Brown, K. Driggs-Campbell and M. J. Kochenderfer. „*A taxonomy and review of algorithms for modeling and predicting human driver behavior*,“ 2020.

[89] S. Grigorescu, B. Trasnea, T. Cocias and G. Macesanu, „A survey of deep learning techniques for autonomous driving,“ *Journal of Field Robotics*, vol. 37, no. 3, pp. 362–386, 2020, DOI: 10.1002/rob. 21918.

[90] A. Rudenko, L. Palmieri, M. Herman, K. M. Kitani, D. M. Gavrila, et al., „Human motion trajectory prediction: a survey,“ *The International Journal of Robotics Research*, vol. 39, no. 8, pp. 895–935, 2020, DOI: 10.1177/0278364920917446.

[91] A. Fuchs, A. Passarella and M. Conti, „Modeling, Replicating, and Predicting Human Behavior: A Survey,“ *ACM Trans. Auton. Adapt. Syst.*, vol. 18, no. 2, 2023, DOI: 10.1145/3580492.

[92] M. Ito, „The uncertainty that the autonomous car faces and predictability analysis for evaluation,“ pp. 83–95, 2019, DOI: 10.1007/978-3-030-28005-5_7.

[93] ISO, „ISO 26262 Road vehicles – Functional safety,“ 2011.

[94] ISO, „ISO 21448 Road vehicles – Safety of the intended functionality,“ 2022.

[95] S. Teng, X. Hu, P. Deng, B. Li, Y. Li, et al., „Motion Planning for Autonomous Driving: The State of the Art and Future Perspectives,“ *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 6, pp. 3692–3711, 2023, DOI: 10.1109/TIV.2023.3274536.

[96] C. Zhou, B. Huang and P. Fränti, „A review of motion planning algorithms for intelligent robots,“ *Journal of Intelligent Manufacturing*, vol. 33, no. 2, pp. 387–424, 2022, DOI: 10.1007/s10845-021-01867-z.

[97] L. Antonyshyn, J. Silveira, S. Givigi and J. Marshall, „Multiple Mobile Robot Task and Motion Planning: A Survey,“ *ACM Computing Surveys*, vol. 55, no. 10, 2023, DOI: 10.1145/3564696.

[98] R. Schubert, E. Richter and G. Wanielik, „Comparison and evaluation of advanced motion models for vehicle tracking,“ in *2008 11th International Conference on Information Fusion*, 2008, pp. 1–6.

[99] M. Tsogas, A. Polychronopoulos and A. Amditis, „Unscented Kalman filter design for curvilinear motion models suitable for automotive safety applications,“ in *2005 7th International Conference on Information Fusion*, 2005, p. 8, DOI: 10.1109/ICIF.2005.1592006.

[100] C. Barrios and Y. Motai, „Improving Estimation of Vehicle’s Trajectory Using the Latest Global Positioning System With Kalman Filtering,“ *IEEE Transactions on Instrumentation and Measurement*, vol. 60, no. 12, pp. 3747–3755, 2011, DOI: 10.1109/TIM.2011.2147670.

[101] D. Petrich, T. Dang, G. Breuel and C. Stiller, „Assessing map-based maneuver hypotheses using probabilistic methods and evidence theory,“ in *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, 2014, pp. 995–1002, DOI: 10.1109/ITSC.2014.6957818.

[102] G. Xie, H. Gao, L. Qian, B. Huang, K. Li, et al., „Vehicle Trajectory Prediction by Integrating Physics- and Maneuver-Based Approaches Using Interactive Multiple Models,“ *IEEE Transactions on Industrial Electronics*, vol. 65, no. 7, pp. 5999–6008, 2018, DOI: 10.1109/TIE.2017.2782236.

[103] S. Hoermann, D. Stumper and K. Dietmayer, „Probabilistic long-term prediction for autonomous vehicles,“ in *2017 IEEE Intelligent Vehicles Symposium (IV)*, 2017, pp. 237–243, DOI: 10.1109/IVS.2017.7995726.

[104] B. Kim, K. Park and K. Yi, „Probabilistic threat assessment with environment description and rule-based multi-traffic prediction for integrated risk management system,“ *IEEE Intelligent Transportation Systems Magazine*, vol. 9, no. 3, pp. 8–22, 2017.

[105] Y. Bar-Shalom, K. C. Chang and H. Blom, „Tracking a maneuvering target using input estimation versus the interacting multiple model algorithm,“ *IEEE Transactions on Aerospace and Electronic Systems*, vol. 25, no. 2, pp. 296–300, 1989, DOI: 10.1109/7.18693.

[106] N. Deo, A. Rangesh and M. M. Trivedi, „How Would Surround Vehicles Move? A Unified Framework for Maneuver Classification and Motion Prediction,“ *IEEE Transactions on Intelligent Vehicles*, vol. 3, no. 2, pp. 129–140, 2018, DOI: 10.1109/TIV.2018.2804159.

[107] J. S. Gill, P. Pisu, V. N. Krovi and M. J. Schmid, „Behavior identification and prediction for a probabilistic risk framework,“ in *Dynamic Systems and Control Conference*, 2019, V002T25A004.

[108] A. T. Schulz and R. Stiefelhagen, „A Controlled Interactive Multiple Model Filter for Combined Pedestrian Intention Recognition and Path Prediction,“ in *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, 2015, pp. 173–178, DOI: 10.1109/ITSC.2015.37.

[109] N. Schneider and D. M. Gavrila, „Pedestrian Path Prediction with Recursive Bayesian Filters: A Comparative Study,“ in *Pattern Recognition*, 2013, pp. 174–183, DOI: 10.1007/978-3-642-40602-7_18.

[110] M. Althoff, „Reachability Analysis and its Application to the Safety Assessment of Autonomous Cars,“ Technische Universität München, 2010.

[111] M. Althoff, O. Stursberg and M. Buss, „Computing reachable sets of hybrid systems using a combination of zonotopes and polytopes,“ *Nonlinear analysis: hybrid systems*, vol. 4, no. 2, pp. 233–249, 2010, DOI: 10.1016/j.nahs.2009.03.009.

[112] M. Althoff, C. Le Guernic and B. H. Krogh, „Reachable Set Computation for Uncertain Time-Varying Linear Systems,“ in *Proceedings of the 14th International Conference on Hybrid Systems: Computation and Control*, 2011, pp. 93–102, ISBN: 9781450306294. DOI: 10.1145/1967701.1967717.

[113] M. Koschi and M. Althoff, „SPOT: A tool for set-based prediction of traffic participants,“ in *2017 IEEE Intelligent Vehicles Symposium (IV)*, 2017, pp. 1686–1693, DOI: 10.1109/IVS.2017.7995951.

[114] S. M. LaValle, *Planning algorithms*, Cambridge university press, 2006, ISBN: 978-0521862059.

[115] S. Lloyd, „Least squares quantization in PCM,“ *IEEE transactions on information theory*, vol. 28, no. 2, pp. 129–137, 1982, DOI: 10.1109/TIT.1982.1056489.

[116] J. H. Ward Jr, „Hierarchical grouping to optimize an objective function,“ *Journal of the American Statistical Association*, vol. 58, no. 301, pp. 236–244, 1963, DOI: 10.2307/2282967.

[117] M. Nanni and D. Pedreschi, „Time-focused clustering of trajectories of moving objects,“ *Journal of Intelligent Information Systems*, vol. 27, pp. 267–289, 2006, DOI: 10.1007/s10844-006-9953-7.

[118] K. Driggs-Campbell, V. Govindarajan and R. Bajcsy, „Integrating Intuitive Driver Models in Autonomous Planning for Interactive Maneuvers,“ *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 12, pp. 3461–3472, 2017, DOI: 10.1109/TITS.2017.2715836.

[119] R. Bellman and R. Kalaba, „On adaptive control processes,“ *IRE Transactions on Automatic Control*, vol. 4, no. 2, pp. 1–9, 1959, DOI: 10.1109/TAC.1959.1104847.

[120] M. Vlachos, G. Kollios and D. Gunopulos, „Elastic translation invariant matching of trajectories,“ *Machine Learning*, vol. 58, pp. 301–334, 2005, DOI: 10.1007/s10994-005-5830-9.

[121] S. Atev, G. Miller and N. P. Papanikolopoulos, „Clustering of vehicle trajectories,“ *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 3, pp. 647–657, 2010, DOI: 10.1109/TITS.2010.2048101.

[122] W. Wang, A. Ramesh, J. Zhu, J. Li and D. Zhao, „Clustering of driving encounter scenarios using connected vehicle trajectories,“ *IEEE Transactions on Intelligent Vehicles*, vol. 5, no. 3, pp. 485–496, 2020, DOI: 10.1109/TIV.2020.2973550.

[123] J. Wiest, M. Höffken, U. Kreßel and K. Dietmayer, „Probabilistic trajectory prediction with Gaussian mixture models,“ in *2012 IEEE Intelligent vehicles symposium*, 2012, pp. 141–146, DOI: 10.1109/IVS.2012.6232277.

[124] Y. Hu, W. Zhan and M. Tomizuka, „Probabilistic prediction of vehicle semantic intention and motion,“ in *2018 IEEE Intelligent Vehicles Symposium (IV)*, 2018, pp. 307–313, DOI: 10.1109/IVS.2018.8500419.

[125] J. Bian, D. Tian, Y. Tang and D. Tao, „A survey on trajectory clustering analysis,“ *arXiv preprint arXiv:1802.06971*, 2018.

[126] C. Cortes and V. Vapnik, „Support-vector networks,“ *Machine Learning*, vol. 20, pp. 273–297, 1995, DOI: 10.1007/BF00994018.

[127] P. Kumar, M. Perrollaz, S. Lefevre and C. Laugier, „Learning-based approach for online lane change intention prediction,“ in *2013 IEEE Intelligent Vehicles Symposium (IV)*, 2013, pp. 797–802, DOI: 10.1109/IVS.2013.6629564.

[128] H. M. Mandalia and M. D. D. Salvucci, „Using support vector machines for lane-change detection,“ in *Proceedings of the human factors and ergonomics society annual meeting*, 2005, pp. 1965–1969, DOI: 10.1177/154193120504902217.

[129] H. Woo, Y. Ji, H. Kono, Y. Tamura, Y. Kuroda, et al., „Lane-change detection based on vehicle-trajectory prediction,“ *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 1109–1116, 2017, DOI: 10.1109/LRA.2017.2660543.

[130] G. S. Aoude, B. D. Luders, K. K. H. Lee, D. S. Levine and J. P. How, „Threat assessment design for driver assistance system at intersections," in *13th international ieee conference on intelligent transportation systems*, 2010, pp. 1855–1862, DOI: 10.1109/ITSC.2010.5625287.

[131] Z. Ghahramani, „An introduction to hidden Markov models and Bayesian networks," *International journal of pattern recognition and artificial intelligence*, vol. 15, no. 01, pp. 9–42, 2001, DOI: 10.1142/S0218001401000836.

[132] W. Yuan, Z. Li and C. Wang, „Lane-change prediction method for adaptive cruise control system with hidden Markov model," *Advances in Mechanical Engineering*, vol. 10, no. 9, 2018, DOI: 10.1177/1687814018802932.

[133] W. Zhan, L. Sun, Y. Hu, J. Li and M. Tomizuka, „Towards a fatality-aware benchmark of probabilistic reaction prediction in highly interactive driving scenarios," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, 2018, pp. 3274–3280.

[134] 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*, vol. 17, no. 10, pp. 2751–2766, 2016, DOI: 10.1109/TITS.2016.2522507.

[135] M. Schreier, V. Willert and J. Adamy, „Bayesian, maneuver-based, long-term trajectory prediction and criticality assessment for driver assistance systems," in *17th international IEEE conference on intelligent transportation systems (ITSC)*, 2014, pp. 334–341, DOI: 10.1109/ITSC.2014.6957713.

[136] M.-F. Chang, J. Lambert, P. Sangkloy, J. Singh, S. Bak, et al., „Argoverse: 3D Tracking and Forecasting With Rich Maps," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, DOI: 10.1109/CVPR.2019.00895.

[137] B. Wilson, W. Qi, T. Agarwal, J. Lambert, J. Singh, et al., „Argoverse 2: Next Generation Datasets for Self-Driving Perception and Forecasting," 2021.

[138] N. Harmening, M. Biloš and S. Günnemann, „Deep representation learning and clustering of traffic scenarios," *arXiv preprint arXiv:2007.07740*, 2020.

[139] S. H. Park, B. Kim, C. M. Kang, C. C. Chung and J. W. Choi, „Sequence-to-Sequence Prediction of Vehicle Trajectory via LSTM Encoder-Decoder Architecture," in *IEEE Intelligent Vehicles Symposium (IV)*, 2018, pp. 1672–1678, ISBN: 978-1-5386-4452-2. DOI: 10.1109/IVS.2018.8500658.

[140] B. Kim, C. M. Kang, J. Kim, S. H. Lee, C. C. Chung, et al., „Probabilistic vehicle trajectory prediction over occupancy grid map via recurrent neural network," in *IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, 2017, pp. 399–404, ISBN: 978-1-5386-1526-3. DOI: 10.1109/ITSC.2017.8317943.

[141] H. Ma, J. Li, W. Zhan and M. Tomizuka, „Wasserstein Generative Learning with Kinematic Constraints for Probabilistic Interactive Driving Behavior Prediction," in *IEEE Intelligent Vehicles Symposium*, 2019, pp. 2477–2483, DOI: 10.1109/IVS.2019.8813783.

[142] J. Li, H. Ma and M. Tomizuka, „Interaction-aware Multi-agent Tracking and Probabilistic Behavior Prediction via Adversarial Learning," in *International Conference on Robotics and Automation (ICRA)*, 2019, pp. 6658–6664, ISBN: 978-1-5386-6027-0. DOI: 10.1109/ICRA.2019.8793661.

[143] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese and A. Alahi, „Social GAN: Socially Acceptable Trajectories With Generative Adversarial Networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, DOI: 10.1109/CVPR.2018.00240.

[144] X. Huang, S. G. McGill, J. A. DeCastro, L. Fletcher, J. J. Leonard, et al., „DiversityGAN: Diversity-Aware Vehicle Motion Prediction via Latent Semantic Sampling," *IEEE Robotics and Automation Letters*, no. 5, pp. 5089–5096, 2020, DOI: 10.1109/LRA.2020.3005369. Available: https://arxiv.org/pdf/1911.12736.

[145] R. Chandra, T. Guan, S. Panuganti, T. Mittal, U. Bhattacharya, et al., „Forecasting trajectory and behavior of road-agents using spectral clustering in graph-lstms,“ *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4882–4890, 2020, DOI: 10.1109/LRA.2020.3004794.

[146] J. Mercat, T. Gilles, N. E. Zoghby, G. Sandou, D. Beauvois, et al., „Multi-Head Attention for Joint Multi-Modal Vehicle Motion Forecasting,“ in *IEEE International Conference on Robotics and Automation*, 2020, DOI: 10.1109/ICRA40945.2020.9197340.

[147] H. Cui, V. Radosavljevic, F.-C. Chou, T.-H. Lin, T. Nguyen, et al., „Multimodal Trajectory Predictions for Autonomous Driving using Deep Convolutional Networks,“ in *International Conference on Robotics and Automation (ICRA)*, 2019, pp. 2090–2096, ISBN: 978-1-5386-6027-0. DOI: 10.1109/ICRA.2019.8793868.

[148] T. Phan-Minh, E. C. Grigore, F. A. Boulton, O. Beijbom and E. M. Wolff, „CoverNet: Multimodal Behavior Prediction Using Trajectory Sets,“ in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, DOI: 10.1109/CVPR42600.2020.01408.

[149] S. Konev, K. Brodt and A. Sanakoyeu, „MotionCNN: a strong baseline for motion prediction in autonomous driving,“ *arXiv preprint arXiv:2206.02163*, 2022.

[150] D. Ridel, N. Deo, D. Wolf and M. Trivedi, „Scene Compliant Trajectory Forecast With Agent-Centric Spatio-Temporal Grids,“ *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2816–2823, 2020, DOI: 10.1109/LRA.2020.2974393.

[151] L. Furtner, „Development of a raster-based scene representation for motion prediction of road users via CNN,“ Semester Thesis, Lehrstuhl für Fahrzeugtechnik, Technischen Universität München, 2021.

[152] L. Fang, Q. Jiang, J. Shi and B. Zhou, „TPNet: Trajectory Proposal Network for Motion Prediction,“ in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, DOI: 10.1109/CVPR42600.2020.00683.

[153] N. Djuric, V. Radosavljevic, H. Cui, T. Nguyen, F.-C. Chou, et al., „Uncertainty-aware Short-term Motion Prediction of Traffic Actors for Autonomous Driving,“ in *The IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2020.

[154] M. Khakzar, A. Rakotonirainy, A. Bond and S. G. Dehkordi, „A Dual Learning Model for Vehicle Trajectory Prediction,“ *IEEE Access*, vol. 8, pp. 21897–21908, 2020, DOI: 10.1109/ACCESS.2020.2968618.

[155] A. Breuer, S. Elflein, T. Joseph, J.-A. Bolte, S. Homoceanu, et al., „Analysis of the Effect of Various Input Representations for LSTM-Based Trajectory Prediction,“ in *IEEE Intelligent Transportation Systems Conference (ITSC)*, 2019, pp. 2728–2735, ISBN: 978-1-5386-7024-8. DOI: 10.1109/ITSC.2019.8917373.

[156] T. Zhao, Y. Xu, M. Monfort, W. Choi, C. Baker, et al., „Multi-Agent Tensor Fusion for Contextual Trajectory Prediction,“ in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 12118–12126, DOI: 10.1109/CVPR.2019.01240.

[157] M. Schreiber, S. Hoermann and K. Dietmayer, „Long-Term Occupancy Grid Prediction Using Recurrent Neural Networks,“ in *International Conference on Robotics and Automation (ICRA)*, 2019, pp. 9299–9305, DOI: 10.1109/ICRA.2019.8793582.

[158] S. Li, N. Li, A. Girard and I. Kolmanovsky, „Decision making in dynamic and interactive environments based on cognitive hierarchy theory, Bayesian inference, and predictive control,“ in *2019 IEEE 58th Conference on Decision and Control (CDC)*, 2019, pp. 2181–2187, DOI: 10.1109/CDC40024.2019.9029646.

[159] M. Schäfer, K. Zhao, M. Bühren and A. Kummert, „Context-Aware Scene Prediction Network (CASP-Net),“ in *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, 2022, pp. 3970–3977, DOI: 10.1109/ITSC55140.2022.9921850.

[160] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. S. Torr, et al., „DESIRE: Distant Future Prediction in Dynamic Scenes With Interacting Agents,“ in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, DOI: 10.1109/CVPR.2017.233.

[161] N. Deo and M. M. Trivedi, „Convolutional Social Pooling for Vehicle Trajectory Prediction,“ in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPR)* 2018, pp. 1468–1476, DOI: 10.1109/CVPRW.2018.00196.

[162] K. Messaoud, I. Yahiaoui, A. Verroust-Blondet and F. Nashashibi, „Non-local Social Pooling for Vehicle Trajectory Prediction,“ in *IEEE Intelligent Vehicles Symposium*, 2019, pp. 975–980, DOI: 10.1109/IVS.2019.8813829.

[163] K. Messaoud, N. Deo, M. M. Trivedi and F. Nashashibi, „Trajectory Prediction for Autonomous Driving based on Multi-Head Attention with Joint Agent-Map Representation,“ pp. 165–170, 2021, DOI: 10.1109/IV48863.2021.9576054.

[164] M. Geisslinger, P. Karle, J. Betz and M. Lienkamp, „Watch-and-Learn-Net: Self-supervised Online Learning for Probabilistic Vehicle Trajectory Prediction,“ in *2021 IEEE international conference on systems, man, and cybernetics (SMC)*, 2021, pp. 869–875, DOI: 10.1109/SMC52423.2021.9659079.

[165] Y. Hu, W. Zhan and M. Tomizuka, „A Framework for Probabilistic Generic Traffic Scene Prediction,“ 30/10/2018. Available: http://arxiv.org/pdf/1810.12506v1.

[166] Y. Hu, W. Zhan, L. Sun and M. Tomizuka, „Multi-modal Probabilistic Prediction of Interactive Behavior via an Interpretable Model,“ pp. 557–563, 2019, DOI: 10.1109/IVS.2019.8813796.

[167] R. Chandra, U. Bhattacharya, A. Bera and D. Manocha, „TraPHic: Trajectory Prediction in Dense and Heterogeneous Traffic Using Weighted Interactions,“ in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 8475–8484, DOI: 10.1109/CVPR.2019.00868.

[168] F. Marchetti, F. Becattini, L. Seidenari and A. D. Bimbo, „Multiple Trajectory Prediction of Moving Agents With Memory Augmented Networks,“ 2023, pp. 6688–6702, DOI: 10.1109/TPAMI.2020.3008558.

[169] B. Kim, S. H. Park, S. Lee, E. Khoshimjonov, D. Kum, et al., „LaPred: Lane-Aware Prediction of Multi-Modal Future Trajectories of Dynamic Agents,“ in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 14631–14640, DOI: 10.1109/CVPR46437.2021.01440.

[170] Z. Zhou, J. Wang, Y.-H. Li and Y.-K. Huang, „Query-Centric Trajectory Prediction,“ in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 17863–17873, DOI: 10.1109/CVPR52729.2023.01713.

[171] T. Gilles, S. Sabatini, D. Tsishkou, B. Stanciulescu and F. Moutarde, „GOHOME: Graph-Oriented Heatmap Output for future Motion Estimation,“ in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 9107–9114, DOI: 10.1109/ICRA46639.2022.9812253.

[172] T. Gilles, S. Sabatini, D. Tsishkou, B. Stanciulescu and F. Moutarde, „THOMAS: Trajectory Heatmap Output with learned Multi-Agent Sampling,“ in *International Conference on Learning Representations*, 2022.

[173] F. J. Wirth, „Conditional Behavior Prediction of Interacting Agents on Map Graphs with Neural Networks,“ Karlsruher Institut für Technologie (KIT), 2023, DOI: 10.5445/IR/1000161378.

[174] J. Li, F. Yang, M. Tomizuka and C. Choi, „Evolvegraph: Multi-agent trajectory prediction with dynamic relational reasoning,“ *Advances in neural information processing systems*, vol. 33, pp. 19783–19794, 2020.

[175] Y. Ma, X. Zhu, S. Zhang, R. Yang, W. Wang, et al., „Trafficpredict: Trajectory prediction for heterogeneous traffic-agents," in *Proceedings of the AAAI conference on artificial intelligence*, 2019, pp. 6120–6127.

[176] W. Zeng, M. Liang, R. Liao and R. Urtasun, „LaneRCNN: Distributed Representations for Graph-Centric Motion Forecasting," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 532–539, DOI: 10.1109/IROS51168.2021.9636035.

[177] Y. Wu, T. Gilles, B. Stanciulescu and F. Moutarde, „TSGN: Temporal Scene Graph Neural Networks with Projected Vectorized Representation for Multi-Agent Motion Prediction," in *2023 IEEE Intelligent Vehicles Symposium (IV)*, 2023, pp. 1–8, DOI: 10.1109/IV55152.2023.10186764.

[178] C. Tang and R. R. Salakhutdinov, „Multiple Futures Prediction," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. dE. Fox and R. Garnett, ed. Curran Associates, Inc, 2019, pp. 15424–15434.

[179] N. Deo, E. Wolff and O. Beijbom, „Multimodal Trajectory Prediction Conditioned on Lane-Graph Traversals," in *Proceedings of the 5th Conference on Robot Learning*, 2022, pp. 203–212.

[180] S. Shi, L. Jiang, D. Dai and B. Schiele, „Motion Transformer with Global Intention Localization and Local Movement Refinement," in *Advances in Neural Information Processing Systems*, 2022, pp. 6531–6543.

[181] B. Varadarajan, A. Hefny, A. Srivastava, K. S. Refaat, N. Nayakanti, et al., „MultiPath++: Efficient Information Fusion and Trajectory Aggregation for Behavior Prediction," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 7814–7821, DOI: 10.1109/ICRA46639.2022.9812107.

[182] H. Zhao, J. Gao, T. Lan, C. Sun, B. Sapp, et al., „Tnt: Target-driven trajectory prediction," in *Conference on Robot Learning*, 2021, pp. 895–904.

[183] J. Gu, C. Sun and H. Zhao, „DenseTNT: End-to-end Trajectory Prediction from Dense Goal Sets," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 15283–15292, DOI: 10.1109/ICCV48922.2021.01502.

[184] D. Park, H. Ryu, Y. Yang, J. Cho, J. Kim, et al., „Leveraging Future Relationship Reasoning for Vehicle Trajectory Prediction," in *The Eleventh International Conference on Learning Representations*, 2023.

[185] M. Wang, X. Zhu, C. Yu, W. Li, Y. Ma, et al., „Ganet: Goal area network for motion forecasting," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 1609–1615.

[186] T. Salzmann, B. Ivanovic, P. Chakravarty and M. Pavone, „Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data," in *Computer Vision–ECCV 2020: 16th European Conference, Proceedings, Part XVIII 16*, 2020, pp. 683–700, DOI: 10.1007/978-3-030-58523-5_40.

[187] Z. Zhao, H. Fang, Z. Jin and Q. Qiu, „GISNet:Graph-Based Information Sharing Network For Vehicle Trajectory Prediction," in *2020 International Joint Conference on Neural Networks (IJCNN)*, 2020, pp. 1–7, DOI: 10.1109/IJCNN48605.2020.9206770.

[188] X. Li, X. Ying and M. C. Chuah, „GRIP: Graph-based Interaction-aware Trajectory Prediction," in *IEEE Intelligent Transportation Systems Conference (ITSC)*, 2019, pp. 3960–3966, DOI: 10.1109/ITSC.2019.8917228.

[189] Z. Sheng, Y. Xu, S. Xue and D. Li, „Graph-Based Spatial-Temporal Convolutional Network for Vehicle Trajectory Prediction in Autonomous Driving," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 10, pp. 17654–17665, 2022, DOI: 10.1109/TITS.2022.3155749.

[190] X. Gao, X. Jia, Y. Li and H. Xiong, „Dynamic Scenario Representation Learning for Motion Forecasting With Heterogeneous Graph Convolutional Recurrent Networks," *IEEE Robotics and Automation Letters*, vol. 8, no. 5, pp. 2946–2953, 2023, DOI: 10.1109/LRA.2023.3262150.

[191] J. Cheng, X. Mei and M. Liu, „Forecast-MAE: Self-supervised Pre-training for Motion Forecasting with Masked Autoencoders,“ in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 8679–8689, DOI: 10.1109/ICCV51070.2023.00797.

[192] S. Shi, L. Jiang, D. Dai and B. Schiele, „MTR-A: 1st Place Solution for 2022 Waymo Open Dataset Challenge–Motion Prediction,“ *arXiv preprint arXiv:2209.10033*, 2022.

[193] K. Gao, X. Li, B. Chen, L. Hu, J. Liu, et al., „Dual Transformer Based Prediction for Lane Change Intentions and Trajectories in Mixed Traffic Environment,“ *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 6, pp. 6203–6216, 2023, DOI: 10.1109/TITS.2023.3248842.

[194] J. Gao, C. Sun, H. Zhao, Y. Shen, D. Anguelov, et al., „VectorNet: Encoding HD Maps and Agent Dynamics From Vectorized Representation,“ in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 11522–11530, DOI: 10.1109/CVPR42600.2020.01154.

[195] X. Jia, P. Wu, L. Chen, Y. Liu, H. Li, et al., „HDGT: Heterogeneous Driving Graph Transformer for Multi-Agent Trajectory Prediction Via Scene Encoding,“ *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–15, 2023, DOI: 10.1109/TPAMI.2023.3298301.

[196] Y. Liu, J. Zhang, L. Fang, Q. Jiang and B. Zhou, „Multimodal Motion Prediction with Stacked Transformers,“ in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 7573–7582, DOI: 10.1109/CVPR46437.2021.00749.

[197] S. Mozaffari, M. A. Sormoli, K. Koufos and M. Dianati, „Multimodal Manoeuvre and Trajectory Prediction for Automated Driving on Highways Using Transformer Networks,“ *IEEE Robotics and Automation Letters*, vol. 8, no. 10, pp. 6123–6130, 2023, DOI: 10.1109/LRA.2023.3301720.

[198] M. Zhao, H. Tang, P. Xie, S. Dai, N. Sebe, et al., „Bidirectional Transformer GAN for Long-Term Human Motion Prediction,“ *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 19, no. 5, 2023, DOI: 10.1145/3579359.

[199] P. Nikdel, M. Mahdavian and M. Chen, „DMMGAN: Diverse Multi Motion Prediction of 3D Human Joints using Attention-Based Generative Adversarial Network,“ in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 9938–9944, DOI: 10.1109/ICRA48891.2023.10160401.

[200] L. Tang, F. Yan, B. Zou, W. Li, C. Lv, et al., „Trajectory prediction for autonomous driving based on multiscale spatial-temporal graph,“ *IET Intelligent Transport Systems*, vol. 17, no. 2, pp. 386–399, 2023, DOI: 10.1049/itr2.12265.

[201] S. Kullback and R. A. Leibler, „On information and sufficiency,“ *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951, DOI: 10.1214/aoms/1177729694.

[202] V. Prokhorov, E. Shareghi, Y. Li, M. T. Pilehvar and N. Collier, „On the Importance of the Kullback-Leibler Divergence Term in Variational Autoencoders for Text Generation,“ in *Proceedings of the 3rd Workshop on Neural Generation and Translation*, 2019, pp. 118–127, DOI: 10.18653/v1/D19-5612.

[203] F. Janjos, M. Keller, M. Dolgov and J. M. Zöllner, „Bridging the Gap Between Multi-Step and One-Shot Trajectory Prediction via Self-Supervision,“ *2023 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1–8, 2023.

[204] M.-H. Guo, T.-X. Xu, J.-J. Liu, Z.-N. Liu, P.-T. Jiang, et al., „Attention mechanisms in computer vision: A survey,“ *Computational visual media*, vol. 8, no. 3, pp. 331–368, 2022, DOI: 10.1007/s41095-022-0271-y.

[205] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, et al., „Graph neural networks: A review of methods and applications,“ *AI Open*, vol. 1, pp. 57–81, 2020, DOI: 10.1016/j.aiopen.2021.01.001.

[206] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, et al., „A Comprehensive Survey on Graph Neural Networks,“ *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 1, pp. 4–24, 2021, DOI: 10.1109/TNNLS.2020.2978386.

[207] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, et al., „Attention is all you need,“ *Advances in neural information processing systems*, vol. 30, 2017.

[208] Z. Kim, G. Gomes, R. Hranac and A. Skabardonis, „A machine vision system for generating vehicle trajectories over extended freeway segments,“ in *12th World Congress on Intelligent Transportation Systems*, 2005.

[209] P. Xu, X. Zhu and D. A. Clifton, „Multimodal Learning With Transformers: A Survey,“ *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 10, pp. 12113–12132, 2023, DOI: 10.1109/TPAMI.2023.3275156.

[210] W. L. Hamilton, *Graph representation learning*, Morgan & Claypool Publishers, 2020.

[211] T. N. Kipf and M. Welling, „Semi-Supervised Classification with Graph Convolutional Networks,“ *arXiv preprint arXiv:1609.02907*, 2017.

[212] W. Hamilton, Z. Ying and J. Leskovec, „Inductive representation learning on large graphs,“ *Advances in neural information processing systems*, vol. 30, 2017, ISBN: 9781510860964.

[213] J.-R. Xue, J.-W. Fang and P. Zhang, „A survey of scene understanding by event reasoning in autonomous driving,“ *International Journal of Automation and Computing*, vol. 15, no. 3, pp. 249–266, 2018, DOI: 10.1007/s11633-018-1126-y.

[214] P. Abbeel and A. Y. Ng, „Apprenticeship learning via inverse reinforcement learning,“ in *Proceedings of the twenty-first international conference on Machine learning*, 2004, p. 1, DOI: 10.1145/1015330.1015430.

[215] B. D. Ziebart, A. Maas, J. A. Bagnell and A. K. Dey, „Maximum entropy inverse reinforcement learning,“ in *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 3*, 2008, pp. 1433–1438, ISBN: 9781577353683. DOI: 10.5555/1620270.1620297.

[216] S. Rosbach, V. James, S. Großjohann, S. Homoceanu and S. Roth, „Driving with Style: Inverse Reinforcement Learning in General-Purpose Planning for Automated Driving,“ in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 2658–2665, DOI: 10.1109/IROS40897.2019.8968205.

[217] S. Rosbach, V. James, S. Großjohann, S. Homoceanu, X. Li, et al., „Driving Style Encoder: Situational Reward Adaptation for General-Purpose Planning in Automated Driving,“ in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 6419–6425, DOI: 10.1109/ICRA40945.2020.9196778.

[218] N. Deo and M. M. Trivedi, „Trajectory forecasts in unknown environments conditioned on grid-based plans,“ *arXiv preprint arXiv:2001.00735*, 2020.

[219] L. Sun, W. Zhan and M. Tomizuka, „Probabilistic Prediction of Interactive Driving Behavior via Hierarchical Inverse Reinforcement Learning,“ in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, 2018, pp. 2111–2117, DOI: 10.1109/ITSC.2018.8569453.

[220] S. K. S. Ghasemipour, S. Gu and R. Zemel, „Understanding the relation between maximum-entropy inverse reinforcement learning and behaviour cloning,“ 2019.

[221] D. A. Pomerleau, „Alvinn: An autonomous land vehicle in a neural network,“ *Advances in neural information processing systems*, vol. 1, 1988.

[222] A. Kuefler, J. Morton, T. Wheeler and M. Kochenderfer, „Imitating driver behavior with generative adversarial networks,“ in *2017 IEEE Intelligent Vehicles Symposium (IV)*, 2017, pp. 204–211, DOI: 10.1109/IVS.2017.7995721.

[223] J. Ho and S. Ermon, „Generative adversarial imitation learning,“ *Advances in neural information processing systems*, vol. 29, 2016.

[224] M. Bansal, A. Krizhevsky and A. Ogale, „Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst," *arXiv preprint arXiv:1812.03079*, 2018.

[225] J. Fu, K. Luo and S. Levine, „Learning robust rewards with adversarial inverse reinforcement learning," *arXiv preprint arXiv:1710.11248*, 2017.

[226] A. Somani, N. Ye, D. Hsu and W. S. Lee, „DESPOT: Online POMDP planning with regularization," *Advances in neural information processing systems*, vol. 26, 2013.

[227] D. Silver and J. Veness, „Monte-Carlo planning in large POMDPs," *Advances in neural information processing systems*, vol. 23, 2010.

[228] S. Ulbrich and M. Maurer, „Probabilistic online POMDP decision making for lane changes in fully automated driving," in *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*, 2013, pp. 2063–2067, DOI: 10.1109/ITSC.2013.6728533.

[229] C. Hubmann, M. Becker, D. Althoff, D. Lenz and C. Stiller, „Decision making for autonomous driving considering interaction and uncertain prediction of surrounding vehicles," in *2017 IEEE Intelligent Vehicles Symposium (IV)*, 2017, pp. 1671–1678, DOI: 10.1109/IVS.2017.7995949.

[230] C. Hubmann, J. Schulz, M. Becker, D. Althoff and C. Stiller, „Automated Driving in Uncertain Environments: Planning With Interaction and Uncertain Maneuver Prediction," *IEEE Transactions on Intelligent Vehicles*, vol. 3, no. 1, pp. 5–17, 2018, DOI: 10.1109/TIV.2017.2788208.

[231] A. Sinha, M. O'Kelly, H. Zheng, R. Mangharam, J. Duchi, et al., „FormulaZero: Distributionally robust online adaptation via offline population synthesis," in *International Conference on Machine Learning*, 2020, pp. 8992–9004.

[232] L. S. Shapley, „Stochastic Games*," *Proceedings of the National Academy of Sciences*, vol. 39, no. 10, pp. 1095–1100, 1953, DOI: 10.1073/pnas.39.10.1095.

[233] M. Bahram, A. Lawitzky, J. Friedrichs, M. Aeberhard and D. Wollherr, „A Game-Theoretic Approach to Replanning-Aware Interactive Scene Prediction and Planning," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 6, pp. 3981–3992, 2016, DOI: 10.1109/TVT.2015.2508009.

[234] M. A. Costa-Gomes, V. P. Crawford and N. Iriberri, „Comparing models of strategic thinking in Van Huyck, Battalio, and Beil's coordination games," *Journal of the European Economic Association*, vol. 7, no. 2-3, pp. 365–376, 2009, DOI: 10.1162/JEEA.2009.7.2-3.365.

[235] N. Li, D. W. Oyler, M. Zhang, Y. Yildiz, I. Kolmanovsky, et al., „Game theoretic modeling of driver and vehicle interactions for verification and validation of autonomous vehicle control systems," *IEEE Transactions on Control Systems Technology*, vol. 26, no. 5, pp. 1782–1797, 2017, DOI: 10.1109/TCST.2017.2723574.

[236] N. Li, I. Kolmanovsky, A. Girard and Y. Yildiz, „Game theoretic modeling of vehicle interactions at unsignalized intersections and application to autonomous vehicle control," in *2018 Annual American Control Conference (ACC)*, 2018, pp. 3215–3220.

[237] J. F. Fisac, E. Bronstein, E. Stefansson, D. Sadigh, S. S. Sastry, et al., „Hierarchical game-theoretic planning for autonomous vehicles," in *2019 International conference on robotics and automation (ICRA)*, 2019, pp. 9590–9596, DOI: 10.1109/ICRA.2019.8794007.

[238] A. Liniger and J. Lygeros, „A noncooperative game approach to autonomous racing," *IEEE Transactions on Control Systems Technology*, vol. 28, no. 3, pp. 884–897, 2019, DOI: 10.1109/TCST.2019.2895282.

[239] M. Wang, Z. Wang, J. Talbot, J. C. Gerdes and M. Schwager, „Game-theoretic planning for self-driving cars in multivehicle competitive scenarios," *IEEE Transactions on Robotics*, vol. 37, no. 4, pp. 1313–1325, 2021, DOI: 10.1109/TRO.2020.3047521.

[240] M. Rowold, „A Preview of Open-Loop and Feedback Nash Trajectories in Racing Scenarios," *arXiv preprint arXiv:2310.00766*, 2023.

[241] A. Tikna, M. Roveri, D. Fontanelli and L. Palopoli, „When graphs meet game theory: a scalable approach for robotic car racing," in *2023 IEEE 47th Annual Computers, Software, and Applications Conference (COMPSAC)*, 2023, pp. 1–8, DOI: 10.1109/COMPSAC57700.2023.00011.

[242] W. Schwarting, J. Alonso-Mora and D. Rus, „Planning and decision-making for autonomous vehicles," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, pp. 187–210, 2018, DOI: 10.1146/annurev-control-060117-105157.

[243] M. Gulzar, Y. Muhammad and N. Muhammad, „A Survey on Motion Prediction of Pedestrians and Vehicles for Autonomous Driving," *IEEE Access*, vol. 9, pp. 137957–137969, 2021, DOI: 10.1109/ACCESS.2021.3118224.

[244] H. Ben-Younes, É. Zablocki, M. Chen, P. Pérez and M. Cord, „Raising Context Awareness in Motion Forecasting," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2022, pp. 4409–4418, DOI: 10.1109/CVPRW56347.2022.00487.

[245] C. Gómez-Huélamo, M. V. Conde, R. Barea and L. M. Bergasa, „Improving Multi-Agent Motion Prediction with Heuristic Goals and Motion Refinement," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2023, pp. 5323–5332, DOI: 10.1109/CVPRW59228.2023.00561.

[246] A. Stockem Novo, C. Hürten, R. Baumann and P. Sieberg, „Self-evaluation of automated vehicles based on physics, state-of-the-art motion prediction and user experience," *Scientific Reports*, vol. 13, no. 1, p. 12692, 2023, DOI: 10.1038/s41598-023-39811-1.

[247] D. Fridovich-Keil, A. Bajcsy, J. F. Fisac, S. L. Herbert, S. Wang, et al., „Confidence-aware motion prediction for real-time collision avoidance," *The International Journal of Robotics Research*, vol. 39, no. 2-3, pp. 250–265, 2020, DOI: 10.1177/0278364919859436.

[248] S. Carrasco Limeros, S. Majchrowska, J. Johnander, C. Petersson, M. Á. Sotelo, et al., „Towards trustworthy multi-modal motion prediction: Holistic evaluation and interpretability of outputs," *CAAI Transactions on Intelligence Technology*, 2023, DOI: 10.1049/cit2.12244.

[249] M. Sensoy, L. Kaplan and M. Kandemir, „Evidential deep learning to quantify classification uncertainty," *Advances in neural information processing systems*, vol. 31, 2018.

[250] W. Shao, Y. Xu, L. Peng, J. Li and H. Wang, „Failure Detection for Motion Prediction of Autonomous Driving: An Uncertainty Perspective," *arXiv preprint arXiv:2301.04421*, 2023.

[251] A. Farid, S. Veer, B. Ivanovic, K. Leung and M. Pavone, „Task-Relevant Failure Detection for Trajectory Predictors in Autonomous Vehicles," in *Proceedings of The 6th Conference on Robot Learning*, 2023, pp. 1959–1969.

[252] M. Althoff and J. M. Dolan, „Online Verification of Automated Road Vehicles Using Reachability Analysis," *IEEE Transactions on Robotics*, vol. 30, no. 4, pp. 903–918, 2014, DOI: 10.1109/TRO.2014.2312453.

[253] C. Schöller, V. Aravantinos, F. Lay and A. Knoll, „What the Constant Velocity Model Can Teach Us About Pedestrian Motion Prediction," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1696–1703, 2020, DOI: 10.1109/LRA.2020.2969925.

[254] H. Wu, T. Phong, C. Yu, P. Cai, S. Zheng, et al., „What Truly Matters in Trajectory Prediction for Autonomous Driving?," *arXiv preprint arXiv:2306.15136*, 2023.

[255] A. Houenou, P. Bonnifait, V. Cherfaoui and W. Yao, „Vehicle trajectory prediction based on motion model and maneuver recognition," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 4363–4369, DOI: 10.1109/IROS.2013.6696982.

[256] X. Huang, S. G. McGill, B. C. Williams, L. Fletcher and G. Rosman, „Uncertainty-Aware Driver Trajectory Prediction at Urban Intersections,“ in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 9718–9724, DOI: 10.1109/ICRA.2019.8794282.

[257] S. Yalamanchi, T.-K. Huang, G. C. Haynes and N. Djuric, „Long-term Prediction of Vehicle Behavior using Short-term Uncertainty-aware Trajectories and High-definition Maps,“ in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, 2020, pp. 1–6, DOI: 10.1109/ITSC45102.2020.9294553.

[258] R. Has, „Development of a hybrid method for the motion prediction of other traffic participants,“ Semester Thesis, Lehrstuhl für Fahrzeugtechnik, Technischen Universität München, 2021.

[259] A. Leberer, „Development of a Hybrid Approach for Motion Prediction in Real-Vehicle Application,“ Master’s Thesis, Lehrstuhl für Fahrzeugtechnik, Technischen Universität München, 2022.

[260] P. Wu, X. Jia, L. Chen, J. Yan, H. Li, et al., „Trajectory-guided Control Prediction for End-to-end Autonomous Driving: A Simple yet Strong Baseline,“ in *Advances in Neural Information Processing Systems*, 2022, pp. 6119–6132.

[261] M. Bahram, C. Hubmann, A. Lawitzky, M. Aeberhard and D. Wollherr, „A Combined Model- and Learning-Based Framework for Interaction-Aware Maneuver Prediction,“ *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 6, pp. 1538–1550, 2016, DOI: 10.1109/TITS.2015.2506642.

[262] S. Kato, S. Tokunaga, Y. Maruyama, S. Maeda, M. Hirabayashi, et al., „Autoware on Board: Enabling Autonomous Vehicles with Embedded Systems,“ in *2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS)*, 2018, pp. 287–296, DOI: 10.1109/ICCPS.2018.00035.

[263] P. Ghorai, A. Eskandarian, Y.-K. Kim and G. Mehr, „State Estimation and Motion Prediction of Vehicles and Vulnerable Road Users for Cooperative Autonomous Driving: A Survey,“ *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 10, pp. 16983–17002, 2022, DOI: 10.1109/TITS.2022.3160932.

[264] T. Betz, P. Karle, F. Werner and J. Betz, „An Analysis of Software Latency for a High-Speed Autonomous Race Car —- A Case Study in the Indy Autonomous Challenge,“ *SAE International Journal of Connected and Automated Vehicles*, vol. 6, no. 3, pp. 283–296, 2023, DOI: 10.4271/12-06-03-0018.

[265] D. Falanga, S. Kim and D. Scaramuzza, „How Fast Is Too Fast? The Role of Perception Latency in High-Speed Sense and Avoid,“ *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1884–1891, 2019, DOI: 10.1109/LRA.2019.2898117.

[266] J. Houston, G. Zuidhof, L. Bergamini, Y. Ye, L. Chen, et al., „One Thousand and One Hours: Self-driving Motion Prediction Dataset,“ in *Proceedings of the 2020 Conference on Robot Learning*, 2021, pp. 409–418.

[267] A. Breuer, J.-A. Termöhlen, S. Homoceanu and T. Fingscheidt, „openDD: A Large-Scale Roundabout Drone Dataset,“ in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, 2020, pp. 1–6, DOI: 10.1109/ITSC45102.2020.9294301.

[268] A. Zakirov, „Sensitivitätsanalyse der Generalisierbarkeit von musterbasierter Prädiktionsmethoden für Autonomes Fahren,“ Bachelor’s Thesis, Lehrstuhl für Fahrzeugtechnik, Technischen Universität München, 2021.

[269] M. Itkina and M. Kochenderfer, „Interpretable self-aware neural networks for robust trajectory prediction,“ in *Conference on Robot Learning*, 2023, pp. 606–617.

[270] J. Wiederer, J. Schmidt, U. Kressel, K. Dietmayer and V. Belagiannis, „Joint Out-of-Distribution Detection and Uncertainty Estimation for Trajectory Prediction,“ *arXiv preprint arXiv:2308.01707*, 2023.

[271] P. Bhattacharyya, C. Huang and K. Czarnecki, „SSL-Lanes: Self-Supervised Learning for Motion Forecasting in Autonomous Driving," in *Proceedings of The 6th Conference on Robot Learning*, 2023, pp. 1793–1805.

[272] C. Jung, A. Finazzi, H. Seong, D. Lee, S. Lee, et al., „An Autonomous System for Head-to-Head Race: Design, Implementation and Analysis; Team KAIST at the Indy Autonomous Challenge," *arXiv preprint arXiv:2303.09463*, 2023.

[273] A. Raji, A. Liniger, A. Giove, A. Toschi, N. Musiu, et al., „Motion Planning and Control for Multi Vehicle Autonomous Racing at High Speeds," in *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, 2022, pp. 2775–2782, DOI: 10.1109/ITSC55140.2022.9922239.

[274] J. Zhong, Z. Liu and X. Chen, „Transformer-based models and hardware acceleration analysis in autonomous driving: A survey," *arXiv preprint arXiv:2304.10891*, 2023.

[275] A. Wischnewski, M. Geisslinger, J. Betz, T. Betz, F. Fent, et al., „Indy Autonomous Challenge - Autonomous Race Cars at the Handling Limits," in *12th International Munich Chassis Symposium 2021: chassis. tech plus*, 2022, pp. 163–182, DOI: 10.1007/978-3-662-64550-5_10.

[276] J. Betz, T. Betz, F. Fent, M. Geisslinger, A. Heilmeier, et al., „TUM autonomous motorsport: An autonomous racing software for the Indy Autonomous Challenge," *Journal of Field Robotics*, vol. 40, no. 4, pp. 783–809, 2023, DOI: 10.1002/rob.22153.

[277] J. Betz, H. Zheng, A. Liniger, U. Rosolia, P. Karle, et al., „Autonomous Vehicles on the Edge: A Survey on Autonomous Vehicle Racing," *IEEE Open Journal of Intelligent Transportation Systems*, vol. 3, pp. 458–488, 2022, DOI: 10.1109/OJITS.2022.3181510.

[278] L. Ögretmen, M. Rowold, M. Ochsenius and B. Lohmann, „Smooth Trajectory Planning at the Handling Limits for Oval Racing," *Actuators*, vol. 11, no. 11, 2022, DOI: 10.3390/act11110318.

[279] M. Rowold, L. Ögretmen, T. Kerbl and B. Lohmann, „Efficient Spatiotemporal Graph Search for Local Trajectory Planning on Oval Race Tracks," *Actuators*, vol. 11, no. 11, 2022, DOI: 10.3390/act11110319.

[280] P. Karle, F. Fent, S. Huch, F. Sauerbeck and M. Lienkamp, „Multi-Modal Sensor Fusion and Object Tracking for Autonomous Racing," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 7, pp. 3871–3883, 2023, DOI: 10.1109/TIV.2023.3271624.

[281] P. Karle and TUM – Institute of Automotive Technology, „TUMFTM/FusionTracking: Initial Release," 2022, DOI: 10.5281/zenodo.7220719. Available: 10.5281/zenodo.7220719.

[282] P. Karle. „*TUMFTM/FusionTracking: Dataset*," version 1.0.0. Zenodo, Oct. 2022. DOI: 10.5281/zenodo.7220695. Available: 10.5281/zenodo.7220695.

[283] P. Karle, F. Török, M. Geisslinger and M. Lienkamp, „MixNet: Physics Constrained Deep Neural Motion Prediction for Autonomous Racing," *IEEE Access*, vol. 11, pp. 85914–85926, 2023, DOI: 10.1109/ACCESS.2023.3303841.

[284] P. Karle and TUM – Institute of Automotive Technology, „TUMFTM/MixNet: Initial Release," 2022, DOI: 10.5281/zenodo.6953977. Available: 10.5281/zenodo.6953977.

[285] P. Karle. „*TUMFTM/MixNet: Dataset*," version 1.0.0. Zenodo, Aug. 2022. DOI: 10.5281/zenodo.6954020. Available: 10.5281/zenodo.6954020.

[286] Energy Systems Network, „Indy Autonomous Challenge Rules v5NOV2019," 2019. Available: https://static1.squarespace.com/static/5da73021d0636f4ec706fa0a/t/5dc0680c41954d4ef41ec2b2/1572890638793/Indy+Autonomous+Challenge+Ruleset+-+v5NOV2019+%282%29.pdf [visited on 11/04/2023].

[287] R. Trauth, P. Karle, T. Betz and J. Betz, „An End-to-End Optimization Framework for Autonomous Driving Software," in *2023 3rd International Conference on Computer, Control and Robotics (ICCCR)*, 2023, pp. 137–144, DOI: 10.1109/ICCCR56747.2023.10193889.

[288] S. Maneewongvatana and D. M. Mount. „*Analysis of approximate nearest neighbor searching with clustered point sets*," 1999. Available: http://arxiv.org/pdf/cs/9901013v1.

[289] S. Wolf, „Multi-Object Tracking for Autonomous Racecars," Semester Thesis, Lehrstuhl für Fahrzeugtechnik, Technischen Universität München, 2021.

[290] J. J. LaViola, „A comparison of unscented and extended Kalman filtering for estimating quaternion motion," in *Proceedings of the 2003 American Control Conference, 2003.* 2003, 2435–2440 vol.3, DOI: 10.1109/ACC.2003.1243440.

[291] Dan Simon, „Nonlinear Kalman filtering," *Optimal State Estimation: Kalman, H∞, and Nonlinear Approaches*, pp. 393–431, 2006, DOI: 10.1002/0470045345.ch13.

[292] S. Langenstein, „Multi-Object-Tracking für autonome Rennfahrzeuge," Semester Thesis, Lehrstuhl für Fahrzeugtechnik, Technischen Universität München, 2021.

[293] S. Langenstein, „Entwicklung eines robusten Trackingalgorithmus für Autonome Rennfahrzeuge," Master's Thesis, Lehrstuhl für Fahrzeugtechnik, Technischen Universität München, 2021.

[294] T. D. Powell, „Automated tuning of an extended Kalman filter using the downhill simplex algorithm," *Journal of Guidance, Control, and Dynamics*, vol. 25, no. 5, pp. 901–908, 2002, DOI: 10.2514/2.4983.

[295] F. Török, „Interaction-aware Motion Prediction of Opposing Vehicles for an Autonomous Racecar," Bachelor's Thesis, Lehrstuhl für Fahrzeugtechnik, Technischen Universität München, 2021.

[296] J. Li, H. Ma and M. Tomizuka, „Conditional generative neural system for probabilistic trajectory prediction," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 6150–6156, DOI: 10.1109/IROS40897.2019.8967822.

[297] A. Sadeghian, F. Legros, M. Voisin, R. Vesel, A. Alahi, et al., „Car-net: Clairvoyant attentive recurrent network," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 151–167, DOI: 10.1007/978-3-030-01252-6_10.

[298] A. Heilmeier, A. Wischnewski, L. Hermansdorfer, J. Betz, M. Lienkamp, et al., „Minimum curvature trajectory planning and control for an autonomous race car," *Vehicle System Dynamics*, vol. 58, no. 10, pp. 1497–1527, 2019, DOI: 10.1080/00423114.2019.1631455.

[299] M. Baumgartner, „Systematic Optimization of a data-based Prediction Algorithm for Autonomous Racing," Master's Thesis, Lehrstuhl für Fahrzeugtechnik, Technischen Universität München, 2023.

[300] N. Smirnov, Y. Liu, A. Validi, W. Morales-Alvarez and C. Olaverri-Monreal, „A Game Theory-Based Approach for Modeling Autonomous Vehicle Behavior in Congested, Urban Lane-Changing Scenarios," *Sensors*, vol. 21, no. 4, 2021, DOI: 10.3390/s21041523.

[301] D. R. Lavoie and R. Good, „The nature and use of prediction skills in a biological computer simulation," *Journal of Research in Science Teaching*, vol. 25, no. 5, pp. 335–360, 1988, DOI: 10.1002/tea.3660250503.

[302] A. Bubic, D. Y. von Cramon and R. I. Schubotz, „Prediction, cognition and the brain," *Frontiers in Human Neuroscience*, vol. 4, p. 1094, 2010, DOI: 10.3389/fnhum.2010.00025.

[303] M. Corbetta and G. L. Shulman, „Control of goal-directed and stimulus-driven attention in the brain," *Nature Reviews Neuroscience*, vol. 3, no. 3, pp. 201–215, 2002, DOI: 10.1038/nrn755.

[304] P. Karle, L. Furtner and M. Lienkamp, „Self-Evaluation of Trajectory Predictors for Autonomous Driving," *Electronics*, vol. 13, no. 5, 2024, DOI: 10.3390/electronics13050946.

[305] P. Karle, T. Betz, M. Bosk, F. Fent, N. Gehrke, et al., „EDGAR: An Autonomous Driving Research Platform - From Feature Development to Real-World Application,“ *arXiv preprint arXiv:2309.15492*, 2023.

[306] P. Karle and TUM – Institute of Automotive Technology, „TUMFTM/edgar_digital_twin: Initial Release,“ 2023. Available: https://github.com/TUMFTM/edgar_digital_twin.

[307] P. Karle and TUM – Institute of Automotive Technology, „TUMFTM/SETRIC: Initial Release,“ 2023, DOI: 10.5281/zenodo.8389903. Available: 10.5281/zenodo.8389903.

[308] M. Althoff, M. Koschi and S. Manzinger, „CommonRoad: Composable benchmarks for motion planning on roads,“ in *2017 IEEE Intelligent Vehicles Symposium (IV)*, 2017, pp. 719–726, DOI: 10.1109/IVS.2017.7995802.

[309] N. Uhlemann, F. Fent and M. Lienkamp, „Evaluating Pedestrian Trajectory Prediction Methods for the Application in Autonomous Driving,“ *arXiv preprint arXiv:2308.05194*, 2023.

[310] L. Furtner, „Application of Graph Neural Networks for Agent Motion Prediction and Social Interaction-Aware Model Selection,“ Master’s Thesis, Lehrstuhl für Fahrzeugtechnik, Technischen Universität München, 2022.

[311] P. Hart and A. Knoll, „Graph neural networks and reinforcement learning for behavior generation in semantic environments,“ in *2020 IEEE Intelligent Vehicles Symposium (IV)*, 2020, pp. 1589–1594, DOI: 10.1109/IV47402.2020.9304738.

[312] T. Herbst, „Outlier-Detection for Encoder-Decoder Prediction Algorithms,“ Master’s Thesis, Lehrstuhl für Fahrzeugtechnik, Technischen Universität München, 2022.

[313] L. Igler, „Maneuver Classification for Uncertainty Classification of Encoder-Decoder-Algorithms for Trajectory Prediction,“ Bachelor’s Thesis, Lehrstuhl für Fahrzeugtechnik, Technischen Universität München, 2021.

[314] F. Hechtl, „System Analysis for Decoding Human Driving Behavior,“ Master’s Thesis, Lehrstuhl für Fahrzeugtechnik, Technischen Universität München, 2021.

[315] J. Snoek, H. Larochelle and R. P. Adams, „Practical Bayesian Optimization of Machine Learning Algorithms,“ in *Advances in Neural Information Processing Systems*, 2012.

[316] M. Geisslinger, P. Karle and TUM – Institute of Automotive Technology, „Wale-Net Prediction Network for CommonRoad: Initial Release,“ 2023. Available: https://github.com/TUMFTM/Wale-Net.

[317] G. Li, M. Muller, A. Thabet and B. Ghanem, „DeepGCNs: Can GCNs Go As Deep As CNNs?,“ in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[318] V. G. Cerf, „A comprehensive self-driving car test,“ *Communications of the ACM*, vol. 61, no. 2, p. 7, 2018, DOI: 10.1145/3177753.

[319] N. Webb, D. Smith, C. Ludwick, T. Victor, Q. Hommes, et al., „Waymo’s safety methodologies and safety readiness determinations,“ *arXiv preprint arXiv:2011.00054*, 2020.

[320] T. N. Stahl, „Safeguarding complex and learning-based automated driving functions via online verification,“ Technische Universität München, 2022.

# Prior Publications

During the development of this dissertation, publications and student theses were written in which partial aspects of this work were presented.

## Journals; Scopus/Web of Science listed (peer-reviewed)

[39] F. Sauerbeck, S. Huch, F. Fent, P. Karle, D. Kulmer, et al., „Learn to See Fast: Lessons Learned From Autonomous Racing on How to Develop Perception Systems,“ *IEEE Access*, vol. 11, pp. 44034–44050, 2023, DOI: 10.1109/ACCESS.2023.3272750.

[52] F. Nobis, E. Shafiei, P. Karle, J. Betz and M. Lienkamp, „Radar Voxel Fusion for 3D Object Detection,“ *Applied Sciences*, vol. 11, no. 12, 2021, DOI: 10.3390/app11125598.

[86] P. Karle, M. Geisslinger, J. Betz and M. Lienkamp, „Scenario Understanding and Motion Prediction for Autonomous Vehicles —- Review and Comparison,“ *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 10, pp. 16962–16982, 2022, DOI: 10.1109/TITS.2022.3156011.

[264] T. Betz, P. Karle, F. Werner and J. Betz, „An Analysis of Software Latency for a High-Speed Autonomous Race Car —- A Case Study in the Indy Autonomous Challenge,“ *SAE International Journal of Connected and Automated Vehicles*, vol. 6, no. 3, pp. 283–296, 2023, DOI: 10.4271/12-06-03-0018.

[276] J. Betz, T. Betz, F. Fent, M. Geisslinger, A. Heilmeier, et al., „TUM autonomous motorsport: An autonomous racing software for the Indy Autonomous Challenge,“ *Journal of Field Robotics*, vol. 40, no. 4, pp. 783–809, 2023, DOI: 10.1002/rob.22153.

[277] J. Betz, H. Zheng, A. Liniger, U. Rosolia, P. Karle, et al., „Autonomous Vehicles on the Edge: A Survey on Autonomous Vehicle Racing,“ *IEEE Open Journal of Intelligent Transportation Systems*, vol. 3, pp. 458–488, 2022, DOI: 10.1109/OJITS.2022.3181510.

[280] P. Karle, F. Fent, S. Huch, F. Sauerbeck and M. Lienkamp, „Multi-Modal Sensor Fusion and Object Tracking for Autonomous Racing,“ *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 7, pp. 3871–3883, 2023, DOI: 10.1109/TIV.2023.3271624.

[283] P. Karle, F. Török, M. Geisslinger and M. Lienkamp, „MixNet: Physics Constrained Deep Neural Motion Prediction for Autonomous Racing,“ *IEEE Access*, vol. 11, pp. 85914–85926, 2023, DOI: 10.1109/ACCESS.2023.3303841.

[304] P. Karle, L. Furtner and M. Lienkamp, „Self-Evaluation of Trajectory Predictors for Autonomous Driving,“ *Electronics*, vol. 13, no. 5, 2024, DOI: 10.3390/electronics13050946.

## Conferences, Periodicals; Scopus/Web of Science listed (peer-reviewed)

[164] M. Geisslinger, P. Karle, J. Betz and M. Lienkamp, „Watch-and-Learn-Net: Self-supervised Online Learning for Probabilistic Vehicle Trajectory Prediction,“ in *2021 IEEE international conference on systems, man, and cybernetics (SMC)*, 2021, pp. 869–875, DOI: 10.1109/SMC52423.2021.9659079.

[275] A. Wischnewski, M. Geisslinger, J. Betz, T. Betz, F. Fent, et al., „Indy Autonomous Challenge - Autonomous Race Cars at the Handling Limits," in *12th International Munich Chassis Symposium 2021: chassis. tech plus*, 2022, pp. 163–182, DOI: 10.1007/978-3-662-64550-5_10.

[287] R. Trauth, P. Karle, T. Betz and J. Betz, „An End-to-End Optimization Framework for Autonomous Driving Software," in *2023 3rd International Conference on Computer, Control and Robotics (ICCCR)*, 2023, pp. 137–144, DOI: 10.1109/ICCCR56747.2023.10193889.

## Journals, Conferences, Periodicals, Reports, Conference Proceedings and Poster, etc.; not Scopus/Web of Science listed

[22] P. Karle and M. Lienkamp. „*Autonomous Driving Software Engineering*," Institute of Automotive Technology, Technical University of Munich. 2021. DOI: 10.13140/RG.2.2.10596.76165.

[305] P. Karle, T. Betz, M. Bosk, F. Fent, N. Gehrke, et al., „EDGAR: An Autonomous Driving Research Platform - From Feature Development to Real-World Application," *arXiv preprint arXiv:2309.15492*, 2023.

## Non-thesis-relevant publications; Scopus/Web of Science listed (peer-reviewed)

E. Enders, P. Karle, G. Bonelli, D. Killian and G. Burkhard, „Modal Vertical Vehicle Dynamics Control for Semi-Active and Active Suspension Systems," in *2020 Fifteenth International Conference on Ecological Vehicles and Renewable Energies (EVER)*, 2020, pp. 1–9, DOI: 10.1109/EVER48776.2020.9242985.

## Thesis-relevant open-source software

[281] P. Karle and TUM – Institute of Automotive Technology, „TUMFTM/FusionTracking: Initial Release," 2022, DOI: 10.5281/zenodo.7220719. Available: 10.5281/zenodo.7220719.

[284] P. Karle and TUM – Institute of Automotive Technology, „TUMFTM/MixNet: Initial Release," 2022, DOI: 10.5281/zenodo.6953977. Available: 10.5281/zenodo.6953977.

[306] P. Karle and TUM – Institute of Automotive Technology, „TUMFTM/edgar_digital_twin: Initial Release," 2023. Available: https://github.com/TUMFTM/edgar_digital_twin.

[307] P. Karle and TUM – Institute of Automotive Technology, „TUMFTM/SETRIC: Initial Release," 2023, DOI: 10.5281/zenodo.8389903. Available: 10.5281/zenodo.8389903.

[316] M. Geisslinger, P. Karle and TUM – Institute of Automotive Technology, „Wale-Net Prediction Network for CommonRoad: Initial Release," 2023. Available: https://github.com/TUMFTM/Wale-Net.

# Supervised Student Theses

The following student theses were written within the framework of the dissertation under the supervision of the author in terms of content, technical and scientific support as well as under relevant guidance of the author. In the following, the bachelor, semester and master theses relevant and related to this dissertation are listed. Many thanks to the authors of these theses for their extensive support within the framework of this research project.

[151] L. Furtner, „Development of a raster-based scene representation for motion prediction of road users via CNN,“ Semester Thesis, Lehrstuhl für Fahrzeugtechnik, Technischen Universität München, 2021.

[258] R. Has, „Development of a hybrid method for the motion prediction of other traffic participants,“ Semester Thesis, Lehrstuhl für Fahrzeugtechnik, Technischen Universität München, 2021.

[259] A. Leberer, „Development of a Hybrid Approach for Motion Prediction in Real-Vehicle Application,“ Master's Thesis, Lehrstuhl für Fahrzeugtechnik, Technischen Universität München, 2022.

[268] A. Zakirov, „Sensitivitätsanalyse der Generalisierbarkeit von musterbasierter Prädiktionsmethoden für Autonomes Fahren,“ Bachelor's Thesis, Lehrstuhl für Fahrzeugtechnik, Technischen Universität München, 2021.

[289] S. Wolf, „Multi-Object Tracking for Autonomous Racecars,“ Semester Thesis, Lehrstuhl für Fahrzeugtechnik, Technischen Universität München, 2021.

[292] S. Langenstein, „Multi-Object-Tracking für autonome Rennfahrzeuge,“ Semester Thesis, Lehrstuhl für Fahrzeugtechnik, Technischen Universität München, 2021.

[293] S. Langenstein, „Entwicklung eines robusten Trackingalgorithmus für Autonome Rennfahrzeuge,“ Master's Thesis, Lehrstuhl für Fahrzeugtechnik, Technischen Universität München, 2021.

[295] F. Török, „Interaction-aware Motion Prediction of Opposing Vehicles for an Autonomous Racecar,“ Bachelor's Thesis, Lehrstuhl für Fahrzeugtechnik, Technischen Universität München, 2021.

[299] M. Baumgartner, „Systematic Optimization of a data-based Prediction Algorithm for Autonomous Racing,“ Master's Thesis, Lehrstuhl für Fahrzeugtechnik, Technischen Universität München, 2023.

[310] L. Furtner, „Application of Graph Neural Networks for Agent Motion Prediction and Social Interaction-Aware Model Selection,“ Master's Thesis, Lehrstuhl für Fahrzeugtechnik, Technischen Universität München, 2022.

[312] T. Herbst, „Outlier-Detection for Encoder-Decoder Prediction Algorithms,“ Master's Thesis, Lehrstuhl für Fahrzeugtechnik, Technischen Universität München, 2022.

[313] L. Igler, „Maneuver Classification for Uncertainty Classification of Encoder-Decoder-Algorithms for Trajectory Prediction,“ Bachelor's Thesis, Lehrstuhl für Fahrzeugtechnik, Technischen Universität München, 2021.

[314] F. Hechtl, „System Analysis for Decoding Human Driving Behavior,“ Master's Thesis, Lehrstuhl für Fahrzeugtechnik, Technischen Universität München, 2021.

# Appendix

# A Model Parameterization

## A.1 Multi-Modal Late Fusion and Tracking

Table A.1: Parameterization of the multi-modal late fusion and tracking method [280].

| Parameter | Value | Unit | Description |
|---|---|---|---|
| $f_{\text{node}}$ | 50 | Hz | ROS2-node frequency |
| $\boldsymbol{z}_{\text{L}}$ | $(x, y, \psi)$ | - | Measured values LiDAR |
| $x_{\text{std,L}}$ | 0.3 | m | Longitudinal measurement uncertainty LiDAR |
| $y_{\text{std,L}}$ | 0.3 | m | Lateral measurement uncertainty LiDAR |
| $\psi_{\text{std,L}}$ | 0.35 | rad | Yaw angle measurement uncertainty LiDAR |
| $\boldsymbol{z}_{\text{R}}$ | $(x, y, \psi, v)$ | - | Measured values RADAR |
| $x_{\text{std,R}}$ | 3.0 | m | Longitudinal measurement uncertainty RADAR |
| $y_{\text{std,R}}$ | 3.0 | m | Lateral measurement uncertainty RADAR |
| $\psi_{\text{std,R}}$ | 0.35 | rad | Yaw measurement uncertainty RADAR |
| $v_{\text{std,R}}$ | 0.2 | $\text{m s}^{-1}$ | Velocity measurement uncertainty RADAR |
| $f_{\text{EKF}}$ | 100 | Hz | Filter frequency |
| $x_{\text{std,P}}$ | 0.01 | m | Longitudinal initial state standard deviation |
| $y_{\text{std,P}}$ | 0.01 | m | Lateral initial state standard deviation |
| $\psi_{\text{std,P}}$ | 0.3 | rad | Yaw angle initial state standard deviation |
| $v_{\text{std,P}}$ | 4.00 | $\text{m s}^{-1}$ | Velocity initial state standard deviation |
| $\dot{\psi}_{\text{std,P}}$ | 0.3 | $\text{rad s}^{-1}$ | Yaw rate initial state standard deviation |
| $k_v$ | 0.8 | - | Ego speed factor |
| $\dot{x}_{\text{std,Q}}$ | 0.2 | $\text{m s}^{-1}$ | Longitudinal process standard deviation per second |
| $\dot{y}_{\text{std,Q}}$ | 0.2 | $\text{m s}^{-1}$ | Lateral process standard deviation per second |
| $\dot{\psi}_{\text{std,Q}}$ | 0.3 | $\text{rad s}^{-1}$ | Yaw process standard deviation per second |
| $\dot{v}_{\text{std,Q}}$ | 1.0 | $\text{m s}^{-2}$ | Velocity process standard deviation per second |
| $\ddot{\psi}_{\text{std,Q}}$ | 0.5 | $\text{rad s}^{-2}$ | Yaw rate process standard deviation per second |
| $d_{\text{MRG}}$ | 5.1 | m | Merge distance |
| $d_{\text{OBF,out}}$ | 0.3 | m | Buffer distance out-of-bounds-filter, outside |
| $d_{\text{OBF,in}}$ | 0.3 | m | Buffer distance out-of-bounds-filter, inside |
| $d_{\text{MTC}}$ | 5.0 | m | Match distance |
| $t_{\text{MTC}}$ | 25 | - | Match threshold |

## A.2 Physics-Constrained Deep Neural Motion Prediction

Table A.2: Parameterization of the physics-constrained deep neural network prediction method [283].

| Parameter | Value | Unit / Type | Description |
|---|---|---|---|
| $\eta$ | $5 \cdot 10^{-5}$ | float | Initial learning rate |
| $\gamma$ | $0.997$ | float | Learning rate decay factor |
| $\omega$ | $1 \cdot 10^{-7}$ | float | Weight decay factor |
| $n_{\mathrm{hist}}$ | $30$ | int | Number of object history steps |
| $d_{\mathrm{b,d}}$ | $20$ | m | Discretization length along boundary |
| $d_{\mathrm{b,l}}$ | $400$ | m | Sampling length along boundary |
| $t_{\mathrm{pred}}$ | $5.0$ | m | Prediction horizon |
| $f_{\mathrm{pred}}$ | $10.0$ | Hz | Prediction frequency |
| $d_{\mathrm{int,lat}}$ | $1.5$ | m | Lateral interaction distance |
| $d_{\mathrm{int,long}}$ | $1.5$ | m | Longitudinal interaction distance |
| $n_{\mathrm{acc}}$ | $5$ | int | Number of acceleration sections |
| $l_{\mathrm{li}}$ | $1$ | int | Number of hidden layers of linear input embedding |
| $h_{\mathrm{li}}$ | $16$ | int | Size of hidden layers of linear input embedding |
| $l_{\mathrm{enc,h}}$ | $1$ | int | Number of LSTM hidden layers in history encoder |
| $h_{\mathrm{enc,h}}$ | $64$ | int | Size of LSTM hidden layers in history encoder |
| $l_{\mathrm{enc,b}}$ | $1$ | int | Number of LSTM hidden layers in boundary encoder |
| $h_{\mathrm{enc,b}}$ | $64$ | int | Size of LSTM hidden layers in boundary encoder |
| $h_{\mathrm{lat}}$ | $16$ | int | Size of latent embedding |
| $h_{\mathrm{dec,path}}$ | $(256, 256, 32)$ | tuple(int) | Size of hidden layers of path decoder |
| $h_{\mathrm{dec,path,out}}$ | $4$ | int | Output size of hidden layers of path decoder |
| $h_{\mathrm{dec,vel}}$ | $32$ | int | Size of hidden layers of velocity decoder |
| $h_{\mathrm{dec,vel,out}}$ | $1$ | int | Output of hidden layers of velocity decoder |
| $h_{\mathrm{dec,acc}}$ | $32$ | int | Size of LSTM hidden layers in decoder |

## A.3 Self-Evaluation of Trajectory Predictors

Table A.3: Parameterization of the self-evaluation method for trajectory predictors [304].

| Parameter | Value | Unit / Type | Description |
|---|---|---|---|
| $\Phi_{\mathrm{min,rmse}}$ | $0.0$ | float | Minimal selection rate, low-RMSE optimization |
| $\mathrm{RMSE}_{\mathrm{trgt,rmse}}$ | $0.2$ | m | Target RMSE, low-RMSE optimization |
| $\eta_{\mathrm{rmse}}$ | $5\cdot 10^{-4}$ | float | Initial learning rate, low-RMSE optimization |
| $\gamma_{\mathrm{rmse}}$ | $0.9$ | float | Learning rate decay factor, low-RMSE optimization |
| $\alpha_{\mathrm{rmse}}$ | $20$ | int | Learning rate decay step size, low-RMSE optimization |
| $\Phi_{\mathrm{min,sel}}$ | $0.7$ | float | Minimal selection rate, high selection optimization |
| $\mathrm{RMSE}_{\mathrm{trgt,sel}}$ | $0.3$ | m | Target RMSE, high selection optimization |
| $\eta_{\mathrm{sel}}$ | $5\cdot 10^{-3}$ | float | Initial learning rate, high selection optimization |
| $\gamma_{\mathrm{sel}}$ | $0.5$ | float | Learning rate decay factor, high selection optimization |
| $\alpha_{\mathrm{sel}}$ | $15$ | int | Learning rate decay step size, high selection optimization |
| $\Phi_{\mathrm{min,bl}}$ | $0.5$ | float | Minimal selection rate, balanced optimization |
| $\mathrm{RMSE}_{\mathrm{trgt,bl}}$ | $0.3$ | m | Target RMSE, balanced optimization |
| $\eta_{\mathrm{bl}}$ | $1\cdot 10^{-3}$ | float | Initial learning rate, balanced optimization |
| $\gamma_{\mathrm{bl}}$ | $0.1$ | float | Learning rate decay factor, balanced optimization |
| $\alpha_{\mathrm{bl}}$ | $20$ | int | Learning rate decay step size, balanced optimization |
| $d_{\mathrm{in}}$ | $4$ | int | Number of input features to the network per object |
| $d_{\mathrm{out}}$ | $2$ | int | Number of output features to the network per object |
| $f_{\mathrm{pred}}$ | $10.0$ | Hz | Prediction frequency |
| $l_{\mathrm{sc,in}}$ | $32$ | int | Number of input filters for scene image encoding |
| $h_{\mathrm{sc,out}}$ | $16$ | int | Size of output filters for scene image encoding |
| $l_{\mathrm{li}}$ | $1$ | int | Number of hidden layers of linear input embedding |
| $h_{\mathrm{li}}$ | $50$ | int | Size of hidden layers of linear input embedding |
| $h_{\mathrm{li,emb}}$ | $50$ | int | Size of linear input embedding |
| $l_{\mathrm{enc}}$ | $1$ | int | Number of LSTM hidden layers in encoder |
| $h_{\mathrm{enc}}$ | $50$ | int | Size of LSTM hidden layers in encoder |
| $\delta$ | $20$ | m | Threshold for proximity-dependent GNN |
| $m_{\mathrm{dg}}$ | $64$ | int | Message size of GNN embedding |
| $l_{\mathrm{dg}}$ | $1$ | int | Number of hidden layers of GNN embedding |
| $h_{\mathrm{dg}}$ | $48$ | int | Size of hidden layers of GNN embedding |
| $h_{\mathrm{dg,emb}}$ | $128$ | int | Size of GNN input embedding |
| $h_{\mathrm{lat}}$ | $48$ | int | Size of latent embedding |
| $h_{\mathrm{dec}}$ | $128$ | int | Size of LSTM hidden layers in decoder |

# B Components of the Research Vehicle EDGAR

Table B.1: Hardware components of the research vehicle *EDGAR* [305].

| **Component** | **Producer** | **Model** |
|---|---|---|
| Series vehicle | Volkswagen | T7 Multivan 1.4 eHybrid |
| Mono camera | Basler | acA1920-50gc |
| Stereo camera | Framos | D455e |
| Mid-range LiDAR | Ouster | Ouster OS1-128 |
| Long-range LiDAR | Innovusion | Falcon Gen-2 |
| RADAR | Continental | Continental ARS430 |
| GPS-IMU | Novatel | PwrPak7D-E2 |
| Microphone | Infineon | A2B Eval Kit |
| x86 HPC | InoNet | Mayflower B17 |
| ARM HPC | ADLINK | AVA AP1 |
| Network Switch | Netgear | M4250-40G8XF-PoE+ |
| PTP Grandmaster | Masterclock | GMR5000 |
| SDR Transceiver | Ettus | USRP B210 SDR Kit 2x2 |
| V2X System | Cohda | MK5 OBU |
| 5G Router | Milesight | UR75-500GL-G-P-W |
| MIMO Antenna | Panorama | LGMQM4-6-60-24-58 4x4 |
| LED | BTF lighting | WS2815 |
| LED controller | Strip Studio | SPI Matrix |
| Visualization PC | Spo-comm | RUGGED GTX 1050 Ti |

# C Data Analysis for the Self-Evaluation Method

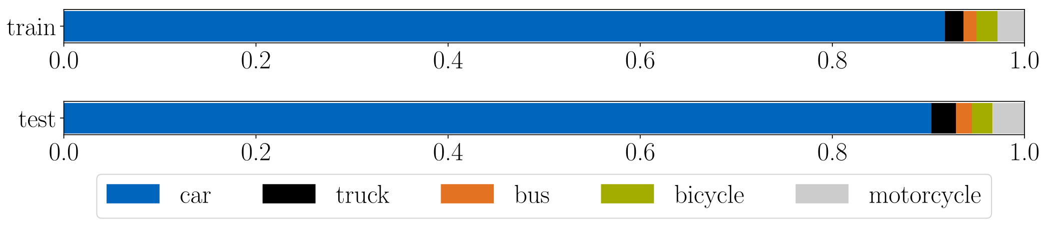


Figure C.1: Ratio of object types in training and test data.

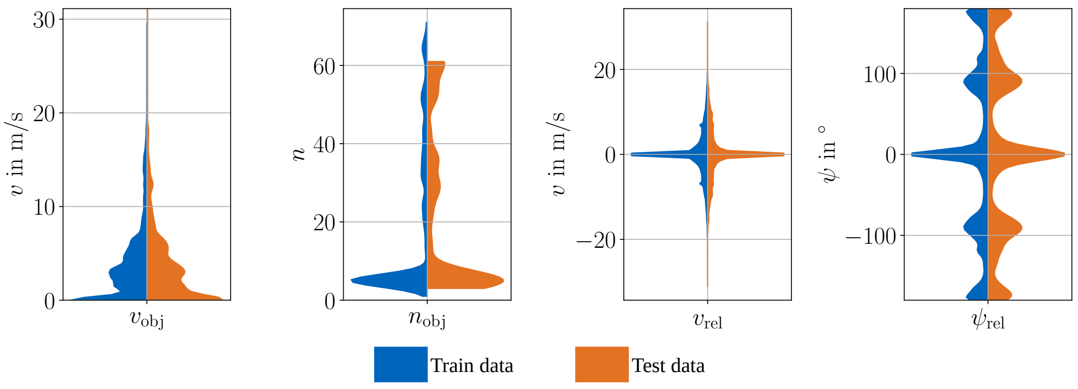


Figure C.2: Data distribution of training and test data. The distribution of the objects' average speeds $v_{\mathrm{obj}}$ and the number of objects per scene $n_{\mathrm{obj}}$ are visualized. In addition, the relative values of speed $v_{\mathrm{rel}}$ and heading $\psi_{\mathrm{obj}}$ between target objects and surrounding objects are given.

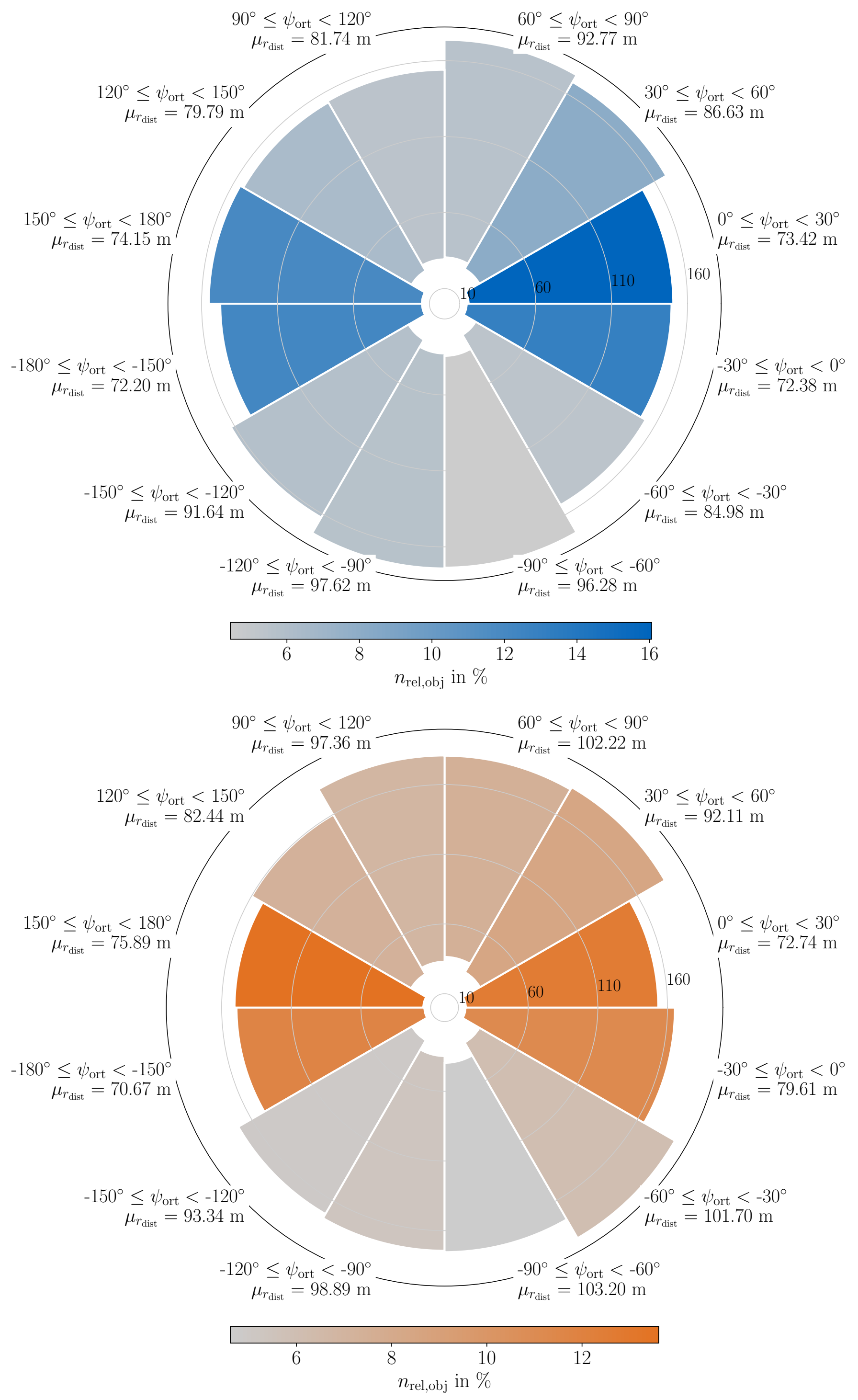


Figure C.3: Spectral distribution of the target objects' neighbors in the training data (top, blue) and the test data (bottom, orange). The radial distance and orientations are given relative to the target object, which is at $r_{\text{dist}} = 0,\ \psi_{\text{ort}} = 0$. The objects are grouped in angular bins of $\Delta\psi_{\text{ort}} = 30\,°$. The bins' lower and upper limits are the 10 %- and 90 %-quantile of the bin's distance distribution. The colors of the bins represent the relative share of objects per bin.