IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 30 June 2023, accepted 6 August 2023, date of publication 9 August 2023, date of current version 16 August 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3303841

RESEARCH ARTICLE

MixNet: Physics Constrained Deep Neural Motion Prediction for Autonomous Racing

PHILLIP KARLE, (Graduate Student Member, IEEE), FERENC TÖRÖK, MAXIMILIAN GEISSLINGER, AND MARKUS LIENKAMP, (Member, IEEE)

Department of Mobility Systems Engineering, TUM School of Engineering and Design, Institute of Automotive Technology, Technical University of Munich (TUM), 85748 Munich, Germany

Corresponding author: Phillip Karle (phillip.karle@tum.de)

This work was supported in part by the Technical University of Munich, in part by the Bavarian Research Foundation (BFS), in part by the Institute for Ethics in Artificial Intelligence (IEAI), and in part by the Leibniz Supercomputing Centre.

ABSTRACT Reliably predicting the motion of contestant vehicles surrounding an autonomous racecar is crucial for effective and performant ego-motion planning. Although highly expressive, deep neural networks are black-box models, making their usage challenging in this safety-critical applications of autonomous racing. On the other hand, physics-based models provide high safety guarantees for the predicted trajectory but lack accuracy. The method presented in this paper targets this trade-off. We introduce a method to predict the trajectories of opposing racecars with deep neural networks considering physical constraints to restrict the output and to improve its feasibility. We report the method's performance against an LSTM-based encoder-decoder architecture on data acquired from multi-agent racing simulations. The proposed method outperforms the baseline model in prediction accuracy and robustness. Still, it fulfills quality guarantees of smoothness and consistency of the predicted trajectory and prevents out-of-track predictions. Thus, a robust real-world application of the model with high prediction accuracy is proven. The presented model was deployed on the racecar of the Technical University of Munich for the Indy Autonomous Challenge 2021. The code used in this research is available as open-source software at www.github.com/TUMFTM/MixNet.

INDEX TERMS Autonomous racing, hybrid deep neural networks, motion prediction, scenario understanding.

I. INTRODUCTION

Developing reliable autonomous vehicles has various purposes, including safer, and more efficient traveling. To fuel innovation, autonomous vehicle competitions such as the DARPA Grand Challenge [1] have taken place. There is, however, an aspect of autonomous driving challenges that has not been covered before: full-scale multi-vehicle racing against other competitors. The Indy Autonomous Challenge (IAC) [2] and its successor, the Autonomous Challenge at the CES 2022[1] (AC@CES) were meant to take this enormous next step. In the competition, the teams were provided with the same hardware and developed their own autonomous software stacks. Wheel-to-wheel racing poses serious challenges. Among these is predicting the future motion of other dynamic objects to operate safely and efficiently in the dynamic environment. To tackle this problem, there are multiple approaches from literature, which use the current estimated state of the vehicle and extrapolate it using a physics-based model [3], [4], [5], [6]. Although these methods appeal due to their transparency and small computational demands, the predicted trajectories become less reliable in the long term due to the simplified assumptions. Other approaches use machine learning techniques to match the motions of the objects to learned patterns and predict accordingly. These methods often provide long-term trajectory predictions of up to 8 s. However, the used approximators are often black-box models, so quality guarantees are hard to give.

The associate editor coordinating the review of this manuscript and approving it for publication was Wei Wei.

[1]https://www.indyautonomouschallenge.com/

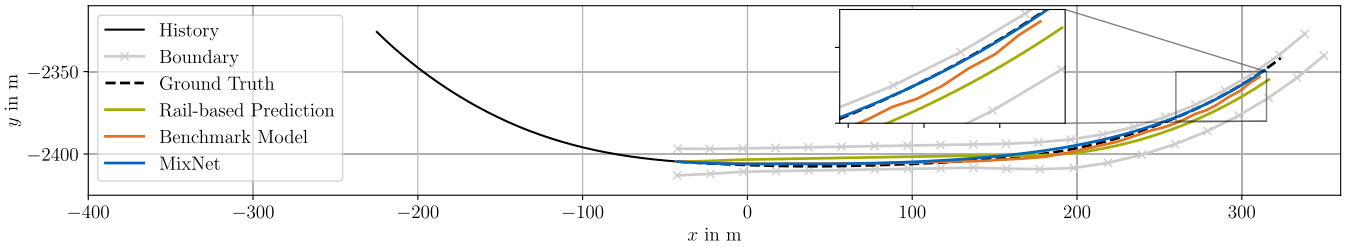

FIGURE 1. Exemplary scenario: The inputs to the MixNet, which are object history and sampled track boundaries, and the different prediction modes of a rail-based approach, the benchmark model, and the MixNet are shown. MixNet combines the best of two worlds: the superposition of continuous base curves and the comprehensive, learned scenario understanding.

Our method combines these two approaches. We propose a constrained deep neural motion prediction model, the MixNet. Our approach uses neural networks (NNs) to encode the scenario but outputs weight parameters for predefined base curves along the race track based on the latent information. So in contrast to common approaches, which determine the future trajectory in the decoder step entirely by deep neural network, MixNet calculates weighting parameters for the superposition of dynamically feasible base curves derived from the track. The output is a smooth and continuous trajectory with the guarantee to stay inside the race track and consistency between consecutive predictions. By this, the holistic scenario understanding of NNs is combined with external constraints given by the race track.

An illustrative example is given in Figure 1. MixNet predicts a smooth trajectory based on the encoded scenario with high accuracy. In comparison, the benchmark model, which has the same encoder architecture, and thus, incorporates the same scenario understanding, determines the prediction by means of a LSTM-based decoder and can not guarantee a stable output. Especially on a long prediction horizon, the trajectory gets noisy. To conclude, the main contributions of this work are:

- A novel trajectory prediction method for autonomous racing, which combines an NN-based encoder for comprehensive scenario understanding with the superposition of physics-based feasible trajectories for a constrained, but accurate output. The approach guarantees smoothness and consistency and trajectory predictions, which lie inside the track.
- A holistic prediction module comprising plausibility checks, external velocity fusion, and an interaction model besides the prediction algorithm for the robust application in a full autonomous racing software stack.
- A synthetic dataset with wheel-to-wheel autonomous racing scenarios to develop and benchmark prediction and planning algorithms

II. RELATED WORK

To give an overview of the state of the art in the field of motion prediction, we structure this section as proposed in [7]. Accordingly, the methods are categorized into physics-based, pattern-based, and planning-based approaches. After introducing the methods for motion prediction in general, we discuss the applicability of autonomous racing in the last subsection. For further details about the overall state of the art of software for autonomous vehicle racing the reader is referred to [8].

A. PHYSICS-BASED APPROACHES

In non-interactive scenarios with independent vehicle behavior and for short-term prediction up to 2 s, the application of kinematics- or dynamics-based vehicle models is a suitable choice [9]. Most commonly, deterministic or probabilistic kinematic models are propagated forward using a constant input assumption. One can choose various longitudinal and turning-related signals input. Hence models with constant velocity, acceleration, yaw rate, and steering angle or a combination of these can be obtained. Schubert et al. [6] provide a review of these models and conclude that Constant Turning Radius and Acceleration (CTRA) models provide the best compromise between prediction accuracy and computational demands. To incorporate uncertainty information, Bayesian Filters, especially Kàlmàn Filters [10] are commonly used. If the linearity assumption of the Kàlmàn Filter does not hold, the Extended (EKF) or Unscented (UKF) versions are applied. By means of an Interaction Multiple Model (IMM) various kinematic models can be combined based on heuristics to improve the expressiveness in heterogeneous scenarios [11]. Another physics-based approach is reachable set predictions [12], which utilize the set of physically possible behaviors. Thus all possible trajectories within the dynamic limits are determined, which are covered by a convex hull. Considering only the trajectories allowed by the traffic rules can limit the solution space accordingly and allows the use in a real vehicle for online verification [13]. For use in autonomous racing, these types of predictions are too conservative for long-term prediction due to the wide range of driving dynamics of the racing vehicle and the lack of explicit rules as in road traffic. The application of the online verification concept in a supervisor module with the use case of autonomous racing is shown in [14].

In general, physics-based methods are computationally cheap and their operation is transparent and well-studied, which makes them appealing in safety-critical application domains such as autonomous driving. The main limitation

arises from the simplified input assumption. This results in fairly accurate predictions on the short horizon. However, the prediction accuracy get insufficient in the long term (>2 s) as the assumption of constant movement does not hold anymore. Hence, these methods are often combined with other approaches which tend to produce more reliable long-term predictions [4], [5].

B. PATTERN-BASED APPROACHES

Pattern-based approaches build on the idea of taking observations of an object, matching it to a pattern, and carrying out the prediction based on it. The pattern assigned to a vehicle can be handcrafted or learned. Furthermore, patterns can be learned by clustering data points or in an abstract space such as the hidden representations of encoder-decoder models. Most of these methods are data-based approaches, hence the need for sufficiently rich datasets is inherent. When using handcrafted patterns, the motion of an object is assigned to one of the predefined maneuver classes. Then, the output prediction is constructed considering the assigned maneuver type and usually involves the usage of prototypes. The classification can be carried out based on heuristics [15] or ML models, such as Support Vector Machines (SVMs) [16], Hidden Markov Models (HMMs) [17], [18], [19] or RNNs [20]. HMMs and RNNs are commonly used due to their inherent capability of interpreting the temporal evolution of a motion history. Instead of using handcrafted patterns, it is also possible to learn clusters from data. Afterward, a single prototype trajectory [21] or a probabilistic representation [22] is obtained for each cluster. During inference time, the output is constructed by classifying the given scenario into a specific cluster and applying the respective representative. Encoder-decoder neural network architectures have recently shown huge potential for motion prediction tasks by dominating the leaderboards of various prediction challenges on public real-world datasets [23], [24]. For the use case of motion prediction, the encoder creates a latent summary of the motion history of an object and the environment information, which serves as an abstract pattern of the scenario. Conditioned on this abstract representation, the decoder determines the future movement of the object. The encoder and the decoder models can both be based on Convolutional Neural Networks [25], [26], RNNs [27], [28] or even Convolutional-RNNs [26], [29]. Attention mechanisms are also commonly used, to allow the network to focus on the relevant parts of the input [27], [28] or to take interactions between vehicles into account [30]. Alternatively, Graph Neural Networks (GNNs) can be used to model interactions in a non-euclidean space. A learned graph representation of the scene is applied to model individual interaction between the single agents. The application on public real-world datasets shows a significant improvement in prediction performance [31], [32]. Another advantage of graph-based prediction models is the more efficient representation in contrast to grid-based approaches, which leads to a reduced calculation time [33]. The bottleneck to provide a sufficiently rich dataset to train a performant algorithm is mitigated in terms of Encoder-decoder architectures due to the fact that no labeled data is needed to create a dataset. The reason for this is that the observation of an object's movement serves as ground truth for output of the prediction algorithm. Thus, the trajectories can simply be split into history and ground truth prediction parts and used in the self-supervised learning setup [34].

C. PLANNING-BASED APPROACHES

Planning-based approaches consider objects to be rational agents acting in an environment according to their hidden policies to reach their goals. The basis of planning-based approaches is the Markov Decision Process (MDP). The approaches usually differ in approximating different parts of the MDP. One approach is to derive handcrafted cost functions to model the agent's behavior [35]. However, the creation of a comprehensive cost function and rule set requires dedicated knowledge and complex traffic scenarios are challenging to represent. Alternatively, the driving behavior can be learned from data, which refers to the field of learning from demonstration. On the one hand, it is possible to apply Inverse Reinforcement Learning, which aims to derive a cost function that fits the observed expert behavior [36], [37], [38]. On the other hand, in the case of Imitation Learning the policy is directly learned from the observed data, i.e., the observation is directly related to a specific behavior [39]. To enhance the robustness of the learned policy generative methods are used, one common approach is Generative Adversarial Imitational Learning [40].

Both pattern-based and planning-based approaches are usually capable of producing more reliable long-term predictions compared to physics-based methods because more comprehensive features can be inputted into the model to consider the interaction between different road users and map constraints. Their shortcomings come from the fact that these methods are primarily data-based and rely on learned policies or patterns. Hence, their performance highly depends on the underlying dataset, which has to represent the Operational Design Domain (ODD) sufficiently. Besides the amount of data, the balance of scenario types influences the prediction performance in edge cases such as safety-critical situations. So, the under-representation of these scenarios directly impacts the applicability. The application in combination with safeguarding methods is also challenging as data-based methods lack of explainability. Hence their functionality can not be adequately supervised.

D. APPLICABILITY FOR AUTONOMOUS RACING

In the following the state of the art is evaluated regarding the applicability for autonomous racing. In our case, we focus on the Indy Autonomous Challenge (IAC), the target for the presented approach. In the inaugural edition, the IAC was held on the Indianapolis Motor Speedway (IMS) and

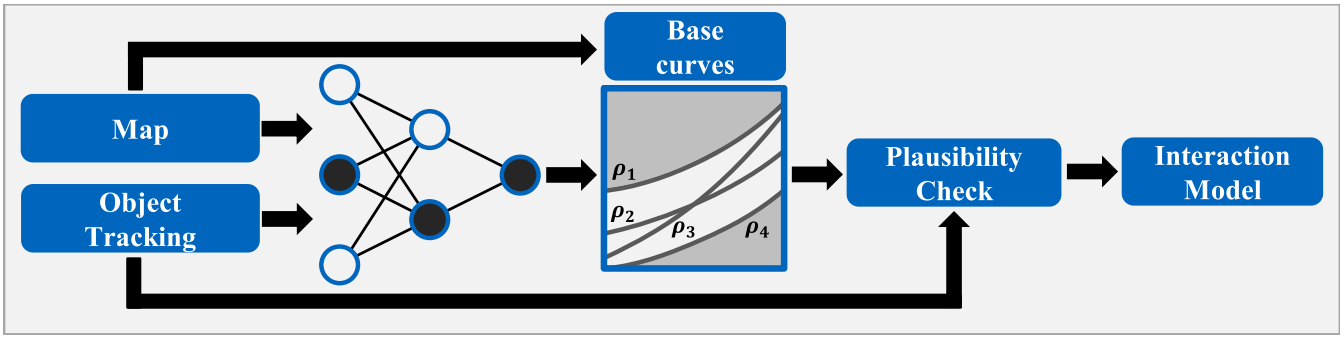

FIGURE 2. Overview of the MixNet prediction module. It combines a comprehensive, learned scenario understanding through an RNN-encoder and semantic knowledge to constrain the output by base curves extracted from the track map.

the Las Vegas Motor Speedway (LVMS), both oval circuits. The circuits do not offer any lanes to define lane-keeping or lane-changing maneuvers. Also, overtaking is assumed to be non-cooperative and the track layout offers many different overtaking options. The prediction algorithm developed for the given race has to fulfill some prerequisites:

1) Due to the chosen planning [41], [42] and overall software [43] concept and high velocities up to 270 km h^{-1} the output should be a single trajectory for each object with a 5 s prediction horizon.
2) The target inference time is 20 ms on a single CPU core due to the available hardware in the race car and the correlation of software latency and performance [44].
3) Motivated by the real-world application of the wheel-to-wheel race the algorithm has to be robust against noise from the perception stack and output a smooth and feasible, i.e., inside the track, trajectory for reliable motion planning.
4) No public race dataset is available for the development of the code.

In consideration of missing traffic lanes, the prediction length does not allow the usage of purely physics-based methods as constant movement assumptions do not hold and the set of dynamically reachable states gets too large. The factors of missing lane information and data discourage the application of classification or clustering-based approaches. The application of planning-based algorithms is questionable because these algorithms require even more comprehensive data to derive the expert behavior correctly and, especially in terms of IL, the robustness towards outliers is not given.

The design factors that motivate our approach combining data-based encoding with dynamically feasible superposition of base curves are the real-time capability and the need for robustness and performance guarantees. Using the encoder network, we can extract patterns from observation. These patterns cover comprehensive object behavior that complex scenarios can be modeled. The superposition of base curves in the decoder constrains the possible prediction to lie inside the race track and guarantees robustness in case of outliers in the input, smoothness of the predicted trajectory, and consistency between consecutive predictions. Thus, the applicability can be enhanced significantly. However, the superposition of base curves is still flexible enough to output a high prediction accuracy as the evaluation (section IV) shows. The interaction model using fuzzy logic guarantees collision-free predictions and further improves the real-world applicability.

III. METHOD

In this section, we introduce MixNet, our encoder-decoder neural network architecture with physical constraints for motion prediction. Figure 2 outlines the method. The object's past movement and map information are passed through the network, which outputs the weighting parameters. The base curves for the superposition are derived from the map. The object movement is also used for plausibility checks, described in section III-C. The last step is the interaction model, which considers the movement of multiple agents in the scene using a fuzzy logic. External velocity information fusion III-B is also part of the approach, which improves the real-world applicability besides the plausibility checks. Besides that, an implemented fuzzy logic that models overtaking maneuvers improves the interaction-awareness of our approach to resolve non-feasible trajectory predictions III-D. Finally, we describe the data mining and training procedure at the end of this section.

A. NETWORK ARCHITECTURE

The proposed network architecture of MixNet is shown in Figure 3. The LSTM layers in the encoder create a latent summary by encoding the object motion histories. The inputs to the network are the $H = 30$ historic 2D-positions up to 3 s in the past, sampled with $f = 10$ Hz. Besides that, the relevant map information, which are the left and right track boundaries starting from the current object's position, equidistantly sampled in vector representation, is inputted to the network. Considering the expected racing speed and prediction horizon, we sample the boundaries up to a horizon of 400 m with 20 m sample distance. During inference, N objects are fed into the network batch-wise. Figure 1 shows an exemplary input of history and track boundaries. The hidden states of the encoding LSTMs are then concatenated and passed to a linear layer, which outputs the latent representation of the scenario.

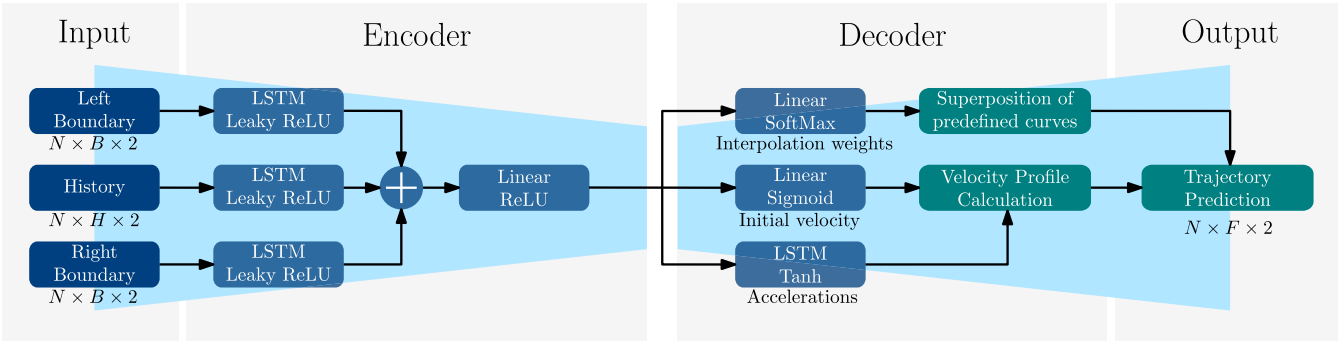

FIGURE 3. The architecture of MixNet. The prediction comprises a path prediction by superposition of base curves and the prediction of an acceleration profile to apply a piece-wise linear velocity profile. Inputs to the network are $H = 30$ past 2D-positions of N objects and the related left and right track boundaries in driving direction in vector representation with B samples. Output is the trajectory prediction of N objects in 2D with a horizon of $F = 50$ steps.

The decoder, the generative part of the model, creates a prediction conditioned on the latent summary. In contrast to other works [27], [28], the future states are not the direct output of an LSTM network, but the forecast trajectory is generated systematically from known schemes. First, the trajectory is obtained by predicting a path and applying a velocity profile to it. Both of these components are generated in a constrained way. The path is created through superpositioning various predefined base curves according to weights predicted by the network through the Linear SoftMax output. The velocity profile is piece-wise 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 by the Linear Sigmoid output activation. The five acceleration values are determined by LSTM layers with Tanh output activation. The final output trajectory is obtained by resampling the path according to the velocity profile. In this way, the set of possible output trajectories is constrained by construction.

We use four base curves for superpositioning: the two track boundaries, a pre-computed minimal curvature raceline [45], and the centerline of the track. The curves are represented by discrete 2D-points of equal number. The points of the boundaries and the raceline are defined by their distances from the centerline points. Hence, the points of each curve at any index i correspond to the same cross section on the track. Due to this fact, the superpositioned curve can be obtained as follows:

$$\rho_{\text{sup}}(s) = \sum_{c \in \mathcal{C}} \lambda_c \rho_c(s) \text{ with} \tag{1}$$

$$\lambda_c \in [0, 1] \quad \forall c \in \mathcal{C} \text{ and } \sum_{c \in \mathcal{C}} \lambda_c = 1 \tag{2}$$

where $\rho_{\text{sup}}(s)$ is the superposed curve along the arc length s of the track with base curves $\rho_c(s)$ and their corresponding weights λ_c in the set of base curves $\mathcal{C}$. Since all base curves are sampled along the same cross sections, s is equal for all base curves. Equation 2 provides the constraints for the superpositioning weights, which result in curves that lie between the left and right boundaries. These constraints can easily be enforced by applying the *softmax* activation to the output of the linear layer.

We emphasize that this method does not guarantee that a predicted trajectory starts exactly from the actual position of a vehicle, but a lateral offset can occur. This is because the superposition weights by Equation 1 are constant along the path and the model is trained to output a prediction with the smallest overall error along the prediction horizon. Hence, a lateral offset at the beginning of the predicted path is possible. To remedy this issue, during the first second of the motion, we apply a plausibility check to the output of the network using a comparison between the current object position and the predicted path. Above a specified threshold, a correction function is applied, which fades from the current position to the predicted path to keep consistency. Especially in transient scenarios, this characteristic is beneficial: If the current motion of the vehicle is strongly transversal, the network learns to predict a path that fits the later parts of the ground truth better by sacrificing accuracy at the beginning of the horizon. With the application of the correction shift, the consistency of the predicted motion is secured so that an additional quality guarantee can be given. During inference, if an adjustment of the first part of the trajectory is triggered to connect the actual position to the prediction path, trajectories similar to lane-change maneuvers can be obtained.

B. VELOCITY INFORMATION FUSION

A further advantage of modular trajectory prediction is that it can also incorporate velocity information from another source to fuse it with the outputted path of the network. The proposed implementation of MixNet offers two possibilities for incorporating external velocity information. First, one can take the complete velocity profile from an external source and use it instead of the one predicted by MixNet. The other possibility is to predict the piece-wise constant accelerations with MixNet but use the external velocity information only to determine the current velocity as initialization.

A reliable source of such information is object tracking which contains the filtered states of the surrounding vehicles. While the current velocity can be extracted from the tracked object's state independently from the ODD a complete velocity profile underlies specific assumptions. For our use case of autonomous racing on a closed track, we propose to determine a complete velocity profile along the whole prediction horizon by forward propagating the tracked object's state utilizing the underlying state-space model. However, this assumption is limited to scenarios with objects at race speed.

C. SAFETY OVERRIDE POSSIBILITY

Predicting accurately at the beginning of the horizon is a safety-relevant issue. On the other hand, significant inaccuracies towards the end of the prediction horizon lead to inefficient planning and bad performance during the race. This trade-off mainly occurs during overtaking maneuvers and when entering into turns. This is because, as stated in 1 and 2, the superposition weights to output the predicted trajectory are constant along the length s of the track. So, a lateral offset can occur. To solve this conflict, our prediction method recognizes and overrides dynamically infeasible predictions with a large lateral offset at the beginning of the horizon. Accordingly, our goal is to probabilistically identify the cases where this happens. Our measure of probability for having generated an initially highly inaccurate forecast is based on the raw path prediction output of the MixNet and the actual state of the vehicle. As stated before, without adjustment, the predicted path does not necessarily start at the actual position of the object. Thus, if the predicted path lies too far from the vehicle considering its actual position and orientation, the prediction is identified as invalid. In this case, we override it with a prediction approach, called the *rail-based* prediction. This approach is derived from the tracked object state and offers high robustness and guarantees kinematic consistency with the current object's state, which is of high importance as a fallback option. The rail-based prediction is composed of a separate path and velocity profile. The path is sampled starting from the current object position in parallel to the track boundaries. The velocity profile is determined by forward propagation of the state-space model as described in III-B. An exemplary prediction by means of the rail-based approach is shown in Figure 1. It can be seen that the rail-based prediction is consistent and accurate at the beginning of the prediction horizon, but has a high lateral error on the long-term horizon.

D. INTERACTION MODELING

As it can be seen in Figure 3 no information about surrounding race cars is fed into the neural network. Hence, there is no explicit interaction-awareness given by the model. Although this assumption holds for many race scenarios with cars following their raceline, interactive scenarios such as overtaking or blocking, which are highly interactive, require additional modeling. In the literature, these interactions are modeled using game theory [46] or learned by a prediction network [47]. These approaches have the disadvantage that they either require a lot of computing time due to iterations or require a vast amount of relevant data with interactions.

Therefore, we propose a two-step approach for trajectory prediction on the racetrack. This first predicts each vehicle by itself and then adjusts the predicted trajectories based on interactions in a second step. In doing so, we take advantage of the set of rules in motorsports. Similar to other racing series such as Formula 1, for safety reasons the movement options of an overtaken vehicle are restricted by the rule set. In the Indy Autonomous Challenge, the vehicle in front is even forced to "hold its raceline" [48]. In other words, it may not block the overtaking vehicle or initiate other unpredictable maneuvers.

Our approach is to predict each vehicle individually, including the ego-vehicle, as a first step. Subsequently, all existing predicted trajectories of the different vehicles are examined for collisions. If no collision is detected, we assume that the influence of the interaction is minor and no adjustment of the trajectories is necessary. However, if at least one collision is detected, we do adapt the trajectories to account for interactions. For this, we examine the race order for each collision and adjust the predicted trajectory of the rear vehicle, since the front vehicle is not allowed to adjust its behavior according to the rules.

To adjust the predicted trajectory, a high-level decision is first made: the faster rear vehicle will either overtake on the left, overtake on the right or not overtake at all and stay behind the front vehicle. This high-level decision is made by a fuzzy logic that takes into account the absolute and relative positions as well as the velocities of the participants. According to the decision, the predicted trajectory is adjusted laterally (in case of an overtake) or longitudinally (in case of no overtake). The adjusted trajectories do not necessarily have to be collision-free, since new collisions with other predicted trajectories may occur. Therefore, the procedure can be repeated as often as necessary until all predictions are resolved collision-free or a termination criterion occurs. In our application, it has proven to be useful in a second iteration to ensure only that the ego vehicle is collision-free with vehicles in the rear. Otherwise, it could happen that the trajectory planning tries to avoid this collision and even makes room for an overtaking vehicle, which would be contrary to winning the race.

E. DATASET

An essential part of Machine Learning applications is a dataset, which covers a diverse set of scenarios expected during inference. However, since this race is the first of its kind, building a dataset from previous races is not possible. Also, there is no public racing dataset available. In this respect, one of the key challenges to be solved is building a dataset for training our neural network approach.

Real-life data was not available until the last weeks before the race itself since real-world tests on the IMS track only

took place then. Hence, training data had to be acquired by simulating our software pipeline [43] in a multi-agent simulation on a Hardware-in-the-Loop (HIL) simulator. The HIL-simulator is able to simulate the full autonomous software stack of up to ten agents in real-time with the same interfaces and callback functions as on the real vehicle. The only constraint is a simplified perception input to reduce the computation load, i.e., the perception pipelines are bypassed with a synthetic object list generator that is input to the tracking module. However, this generator comprises various features to imitate real perception behavior such as limited sensor range and field-of-view, the addition of Gaussian and normal noise, variation of measured objects states, and the simulation of false positives and false negatives. Multi-vehicle races on the HIL-simulator and data from the official simulation race of the IAC, both with highly interactive and complex scenarios, are the basis of our dataset. The recorded logs contain the output of the tracking module [49], which is used to recreate the trajectories of each vehicle during a race. Using the tracking module output to build our dataset has the advantage that the training input distribution of tracked objects and tracking quality will be as close as possible to the expected input distribution during the race. Besides that, the synthetic object list generator is used to vary the perception quality with the aforementioned features to augment the dataset. We added Gaussian noise during the data generation process with mean $\mu = 0$ and different variances in longitudinal and lateral directions based on the evaluated perception and tracking performance.

From these recovered trajectories and the map of the racetrack, it is possible to generate a dataset for training the model. As it was stated before, the input of our model consists of the (x, y) positions of the historical trajectory and the track boundaries around and ahead of the vehicle. Before inputting these points to the model, they are transformed into a local coordinate system, which has its origin at the left boundary next to the current object position and is oriented with its x-axis tangential to the left boundary. With the process above we created 1,569 trajectories from 226 races with an average of 3.4 vehicles per race. From these trajectories, we created 358,025 input-output data pairs.

F. TRAINING

The loss function we defined for training has two terms, one for the fit of the interpolated curve, which relates to the error in the lateral direction, and one for the velocity profile, which reflects the accuracy in the longitudinal direction. The loss L_{path} related to the lateral deviation of the interpolated curve $\hat{\boldsymbol{x}}$ from the ground truth $\boldsymbol{x}$ is costed with a Weighted Mean Square Error (WMSE) at the beginning of the curve as follows:

$$L_{\text{path}} = \frac{1}{F}\left(L_{\text{wmse}} + L_{\text{mse}}\right) \tag{3}$$

$$L_{\text{wmse}} = \sum_{i=1}^{k} (\hat{\boldsymbol{x}}_{\boldsymbol{i}} - \boldsymbol{x}_{\boldsymbol{i}})^2 \left(1 + w\left(1 - \frac{i-1}{k}\right)\right) \tag{4}$$

$$L_{\text{mse}} = \sum_{i=k+1}^{F} (\hat{\boldsymbol{x}}_{\boldsymbol{i}} - \boldsymbol{x}_{\boldsymbol{i}})^2 \tag{5}$$

$$\text{with} \quad w = 0.5, \quad k = 10, \; F = 50 \tag{6}$$

The WMSE decreases linearly from the first prediction step with an additional weight w along the weighting horizon k. The remaining prediction horizon is weighted with 1.0, which corresponds to the conventional Mean Square Error (MSE). The weighting turned out to be beneficial to limit the lateral offset at the beginning of the prediction, even if an additional correction shift is applied at inference. The loss of the velocity profile is also the MSE coming from comparing it to that of the ground truth velocity profile. The overall loss L is then obtained from the path loss L_{path} and the velocity loss L_{vel} as follows:

$$L = L_{\text{path}} + \Delta t^2 \cdot L_{\text{vel}} \tag{7}$$

where $\Delta t = 1/f = 0.1\,\text{s}$ is the timestep size of the prediction in seconds. The multiplication of the velocity error with Δt^2 comes from the following intuition: The velocity error e_v results through integration in a $\Delta t \cdot e_v$ displacement error and a $\Delta t^2 \cdot e_v^2$ squared displacement error if the MSE is applied. By considering the relation $e_v^2 \sim L_{\text{vel}}$, it follows that multiplying the velocity loss term with Δt^2 is necessary to be able to add two loss terms with the same unit, which is m^2.

The hyperparameters are optimized by Bayesian optimization [50]. To train the network we use a learning rate of $5 \cdot 10^{-5}$ with a rate decay of 0.997 per epoch. The L_2 weight regularization has a strength of 10^{-7} and we use a batch size of 128. We train the model for 50 epochs and take the model with the best validation for the evaluation presented in Section IV. The final net and training parameters are published in the open-source code.

In conclusion, the proposed method offers guarantees that the predicted trajectories are always smooth and lie inside the racetrack by means of the structured composition of the prediction. Besides that, it is possible to fuse velocity information from other sources such as the state estimation to enhance the kinematic consistency of the predicted trajectory. Finally, it is possible to probabilistically detect predictions that are highly incorrect at the beginning of the horizon. In these cases, it is possible to override the predictions. For autonomous racing on a closed track, we propose to use a rail-based prediction, which outputs a constant velocity trajectory in parallel to the track boundaries starting from the current object's position.

IV. EXPERIMENTS

In this section, we describe the test procedure, which comprises details about the test dataset, and present a comprehensive evaluation of MixNet's prediction performance. The conducted experiments reveal the overall prediction performance and analyze the model's robustness. Besides that, we investigate the composition of the base curves.

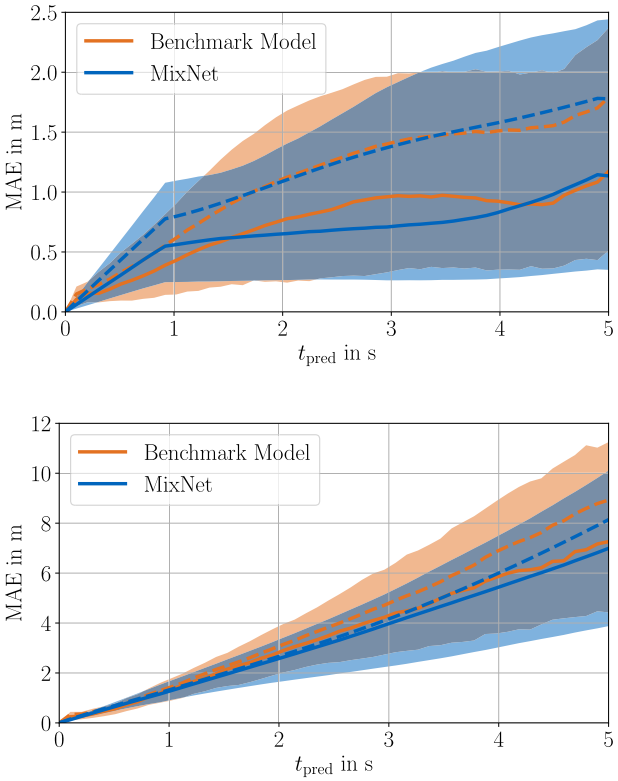

FIGURE 4. Lateral (top) and longitudinal (bottom) error distribution on the prediction horizon. The solid and dashed lines denote the median and mean errors respectively. The colored areas illustrate the range between the Q1 and Q3 quartiles.

A. TEST PROCEDURE

MixNet is compared to a purely LSTM-based encoder-decoder architecture at the evaluation. The encoder architecture of the benchmark model is identical to MixNet's. The difference lies in the decoder architecture, which is in the case of the benchmark model also constructed with LSTM-layers. Thus, the model iteratively outputs a 2D trajectory prediction with a shape of F x 2 for N objects. MixNet and the benchmark model have 198, 214 and 193, 797 trainable parameters respectively. We train both models on the same dataset which is described in subsection III-E. For MixNet, we incorporate initial velocity information from object tracking into the velocity profile but use the piece-wise constant acceleration outputted by the MixNet.

For reproducibility, we have recorded 10 scenarios on our HIL simulator. These include interactive race scenarios with different numbers of vehicles with various speed limits. The recordings can be replayed identically to the pipelines using the two prediction models. For the recording of the interactive scenarios, we used MixNet as the prediction model to run the full software stack. We would like to emphasize that this does not induce any bias in the evaluation process as the respective planning behavior of each object differs from the MixNet model behavior. We report the performances of the models by analyzing the absolute error distributions in the lateral and longitudinal directions with respect to the horizon length using the Mean Average Error (MAE), which is defined as follows:

$$\text{MAE} = \frac{1}{F}\sum_{i=1}^{F} |\hat{\boldsymbol{x}}_{\boldsymbol{i}} - \boldsymbol{x}_{\boldsymbol{i}}|_2 \tag{8}$$

B. OVERALL PREDICTION PERFORMANCE

Experimental evaluation shows that combining the tracked initial velocity and the piece-wise constant acceleration predicted by the MixNet provides the most accurate velocity profile predictions. Hence, similar to path prediction, the combination of an external constraint through the initial velocity and the scenario-aware data-based acceleration prediction performs best.

The overall MAE on the recorded interactive scenarios of MixNet and the benchmark model is 4.91 m and 5.36 m respectively. The reason why MixNet, although being constrained, outperforms the benchmark model becomes apparent when we look at the lateral and longitudinal error distributions on the prediction horizon illustrated in Figure 4. As it can be seen, the magnitude of errors in the lateral direction is very similar in the two cases. This result justifies the hypothesis that obtaining the prediction path by superpositioning our chosen base curves covers the set of possible trajectories properly. The superiority of MixNet in the overall error comes from the fact that it produces smaller errors in the longitudinal direction. Thus, it can be concluded that the combination of the initial velocity from the tracking module and the assumption of piece-wise constant acceleration models the longitudinal movement of the objects more accurately than the iteratively determined output of the LSTM-decoder.

The error distributions with respect to the average velocity of the history are shown in Figure 5. The predictions are separated into the three bins $v < 30\,\text{m}\,\text{s}^{-1}$, $30\,\text{m}\,\text{s}^{-1} < v < 60\,\text{m}\,\text{s}^{-1}$ and $60\,\text{m}\,\text{s}^{-1} < v$ based on the average velocity of their histories, which are associated with a slow-speed and start scenario range, a mid-speed range, and a top-speed range. As Figure 5 shows, the models have similar characteristics regarding their velocity-dependent accuracy. The mean error is the highest at low speeds and it gradually decreases as the velocity grows. There are two main reasons for this. Firstly, most of the transient scenarios like start scenarios, which are challenging to predict, happen at lower speeds. Once the cars have reached their normal racing speed, the scenarios tend to be more steady in the longitudinal direction. Secondly, since most of the racing happens at high speeds, the majority of the training data could be acquired in this velocity range.

The number of vehicles does not have a large effect on the accuracy of the predictions (Figure 6). This is due to the fact that the fuzzy logic can resolve prediction conflicts accurately by deciding between right and left overtake. Moreover, maneuvers with more than two vehicles racing wheel-to-wheel at the same time, which would result in strong lateral interaction, are rare. Instead, scenarios with more than two

TABLE 1. Robustness of the benchmark model and MixNet against zero-mean Gaussian noise.

Standard Deviation		Benchmark Model		MixNet	
σ_{lon} in m	σ_{lat} in m	abs. MAE in m	rel. MAE in %	abs. MAE in m	rel. MAE in %
0.0	0.0	5.36	-	4.91	-
0.5	0.5	5.61	+4.6	5.23	+6.5
1.0	1.0	6.50	+21.2	5.57	+13.4
0.0	1.0	5.76	+7.5	5.39	+9.7
1.0	0.0	6.09	+13.6	5.29	+7.7

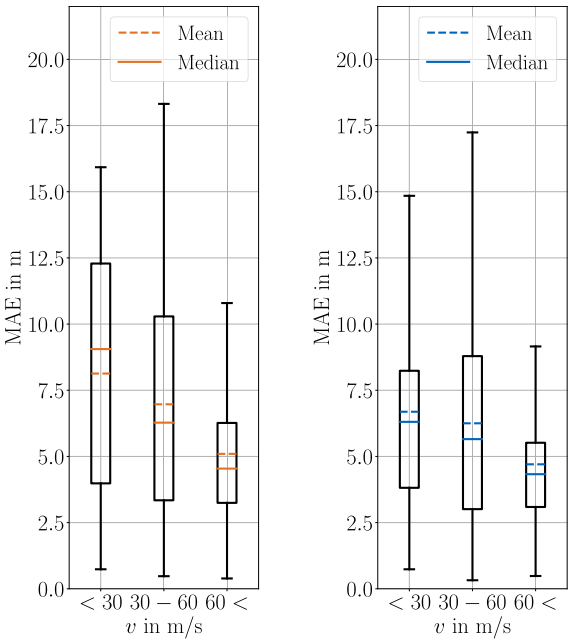

FIGURE 5. The error distributions over the object velocity of the benchmark model (orange) and the MixNet (blue).

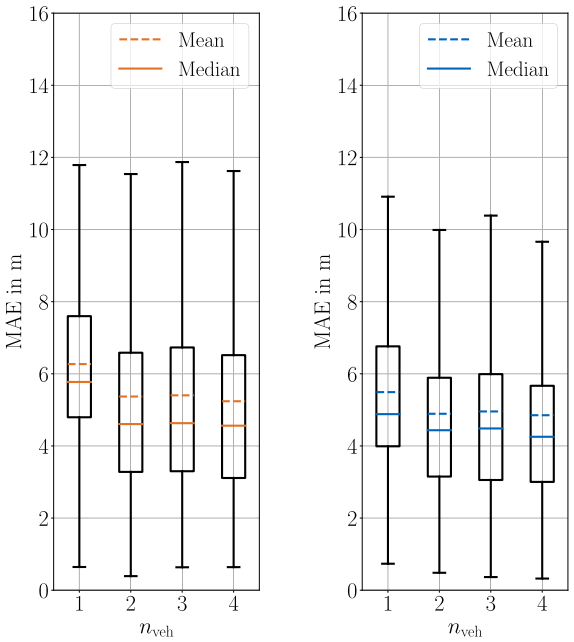

FIGURE 6. The error distributions over the number of objects of the benchmark model (orange) and the MixNet (blue).

vehicles mainly result in sequential overtaking maneuvers between two vehicles respectively.

C. ROBUSTNESS AGAINST NOISE

To investigate the robustness properties of both data-based approaches, we have replayed the test scenarios with extra Gaussian noise added. It should be noted that the original test data is already noisy, but its magnitude is much smaller. We carried out several experiments, all with zero-mean disturbances with different variances in the lateral and longitudinal directions. Table 1 reviews the MAEs of the approaches in the different test cases. As the analysis reveals, MixNet and the benchmark model are both robust against the added noise in the chosen range, although performance degradation occurs. However, it can be noticed that the MAE of the MixNet is lower in all cases. In the case of adding noise in lateral and longitudinal direction, the MAE of the benchmark model increases less than that of the MixNet in case of a standard deviation of 0.5 m in each direction. However, a bigger standard deviation of 1.0 m in both directions results in a significant increase in the MAE of the benchmark model by 21.2%. In contrast, the MixNet's relative increase in the MAE is only 13.4%. This observation indicates that the constraints applied to the MixNet result in the desired model behavior that the model is more robust against variations of the input. The application of Gaussian noise in only one direction reveals that the benchmark model is less sensitive in the lateral direction. However, the relative increase in the MAE for both models is similar. The application of Gaussian noise in the longitudinal direction clearly shows a big advantage of the MixNet. The incorporation of external velocity information from the tracking module results in significantly more robust prediction accuracy as the relative increase in the MAE is only half as big as in the case of the benchmark model.

D. SUPERPOSITIONING WEIGHTS

To investigate how consistently MixNet predicts the weights for curve superpositioning, we input synthetic history trajectories generated with random weights at the entrance of one of the turns on the racetrack. We then observe how well the outputs of MixNet match the inputs. Figure 7 illustrates the input and output weights of the four base curves.

Ideally, the model would output the exact same weights with which the input history is built up resulting

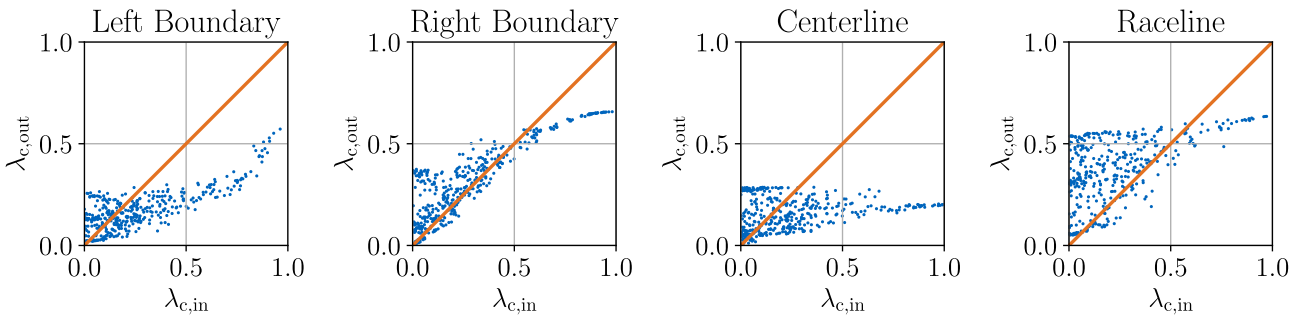

FIGURE 7. The relation between the inputted superpositioning weights of the synthetically generated trajectories and the outputted weights of MixNet.

in 45° straight lines in all of the plots. The closest to this is the figure of the raceline weights. The centerline weights hardly seem to correlate with the input at all. Generally, the network seems to overuse the raceline. The reason for that is that the trajectories in the data usually have a strong raceline component. Hence, the network often assumes raceline following, although the current object might differ from the raceline. Such an example of raceline overuse can be seen in Figure 8. In the turn, the prediction fits the ground truth very well close to the inner bound. However, in the turn exit, the model assumes more optimal driving and a lateral error occurs compared to the ground truth, which follows a more narrow line. The example also reveals that even though the same lateral position results at a specific point, a different combination of superposition weights influences the overall prediction path. As it can be derived from Equation 1 and 2 any point on the racetrack can be obtained through infinitely many different weighting combinations of the 4 base curves. Thus, in this case, a higher weight of the left boundary would result in the same lateral position in the turn, but a different position at the turn exit.

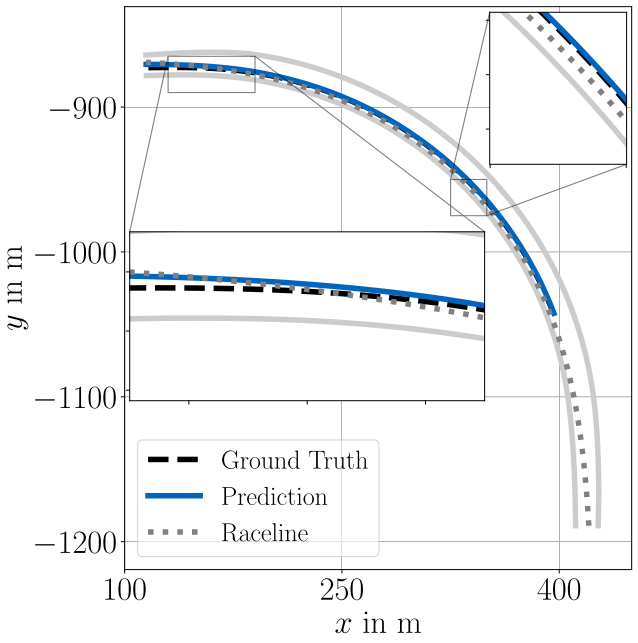

FIGURE 8. Exemplary scenario with overuse of the raceline in a turn. The driving direction is counter-clockwise.

Even though the superposition weights differ from the inputted ones, the overall errors of the prediction are very low in these synthetic cases. The fact that the centerline is underused indicates that this base curve is indeed redundant as it lies close to the middle between the left and right boundary for the major part of the track. Hence, we conduct the following analysis to prove this hypothesis and approximate the centerline as follows:

$$\rho_{\text{center}}(s) \approx 0.5 \cdot \rho_{\text{left}}(s) + 0.5 \cdot \rho_{\text{right}}(s) \tag{9}$$

In this case, the weights corresponding to the centerline can be redistributed and added to the left and right boundary weights without changing the overall superpositioned output as follows:

$$\lambda_{\text{left}} = \lambda_{\text{left}} + 0.5 \cdot \lambda_{\text{center}} \tag{10}$$

$$\lambda_{\text{right}} = \lambda_{\text{right}} + 0.5 \cdot \lambda_{\text{center}} \tag{11}$$

If we plot the left and right boundary weights again, we get the input-output weight relationships illustrated in Figure 9. Here, the weights for the right boundary already match very well the desired linear figure. The left boundary is still mostly substituted by overweighting the raceline. From this, we conclude that the superpositioning weights produced by

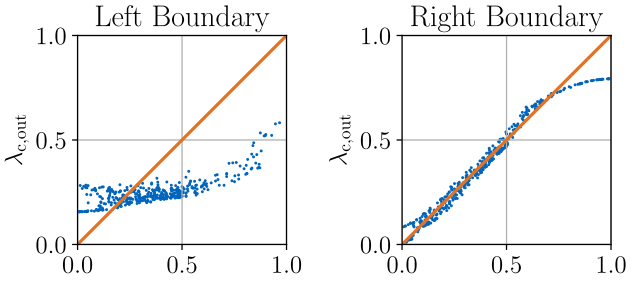

FIGURE 9. The relation between the input superpositioning weights of the synthetically generated trajectories and the outputted weights of MixNet if one redistributes the weights of the centerline to the right and left boundaries.

the network, although they do not match exactly the input weights, which is also not expected, due to the redundancy of the base curves, are consequent and reasonable.

E. COMPUTATION TIME

The average computation times for predicting four vehicles on a single core of an Intel i7-4720HQ 2.6 GHz CPU for MixNet is 9 ms and 15 ms for the benchmark model. The models have encoders of identical sizes, but the LSTM-decoder of the benchmark model takes longer to execute due to iterative calculation of the vehicles' future trajectory, which is the reason for the higher calculation time. Even on a CPU-single core, the computation time is low enough that the model can be applied in a full stack without GPU-usage.

V. CONCLUSION

The given work presents a motion prediction algorithm for autonomous racing, which was part of the software of TUM Autonomous Motorsport developed for Indy Autonomous Challenge, the first high-speed wheel-to-wheel autonomous racing competition.

The proposed method is an encoder-decoder neural network architecture with physical output constraints called MixNet. It contains the scenario understanding of NNs but avoids infeasibilities, which occur in the case of full black-box models. The predicted trajectories are guaranteed to be smooth and consistent and lie inside the racetrack. These guarantees can be given through the restricted base curves, which are superposed by weighting parameters predicted by the network. Furthermore, a rule-based overtaking logic resolves collisions between predicted trajectories and improves interaction awareness. The results show that the method is still flexible enough to produce accurate predictions. It outperforms the benchmark model, an unrestricted LSTM-based encoder-decoder architecture, in the overall accuracy and robustness against noisy inputs. In addition, the algorithm is real-time capable on a single CPU core. The entire code and recorded data are publicly available. The source code additionally comprises a ROS2 launch configuration and a Docker build file.

An interesting future research direction could be to determine a richer set of base curves for superpositioning. In addition, the flexibility of the approach could be enhanced by outputting multiple weights along the path instead of one constant set of weight parameters. Thus, the trade-off between accurate prediction towards the end of the prediction horizon and a low lateral offset between the current object position and the start of the predicted trajectory could be solved. However, both future directions require more data and add complexity to the training process. Besides that, the robustness of real-world applications has to be reviewed in the case of the more flexible network with varying superpositioning weights.

Moreover, interaction-awareness could be improved by directly incorporating surrounding objects into the input. Our presented fuzzy-logic is robust, explainable, and performs well with fewer cars (< 4). However, the scalability of the approach is limited because the sequential application of the decision results in an increasing number of prediction collisions when applied to a higher number of vehicles.

CONTRIBUTIONS

Phillip Karle as the first author initiated the idea of this paper and contributed essentially to its conception, implementation and content. Ferenc Török developed the concept of MixNet, contributed the data generation, training and evaluation procedure and contributed to the writing of the paper. Maximilian Geisslinger contributed to the conception and implementation of this research and the revision of the research article. Markus Lienkamp made an essential contribution to the conception of the research project. He revised the paper critically for important intellectual content. He gave final approval of the version to be published and agrees to all aspects of the work. As a guarantor, he accepts the responsibility for the overall integrity of the paper.

REFERENCES

[1] M. Buehler, K. Iagnemma, and S. Singh, *The 2005 DARPA Grand Challenge: The Great Robot Race*, vol. 36, 1st ed. Berlin, Germany: Springer, 2007, doi: 10.1007/978-3-540-73429-1.
[2] A. Wischnewski, M. Geisslinger, J. Betz, T. Betz, F. Fent, A. Heilmeier, L. Hermansdorfer, T. Herrmann, S. Huch, P. Karle, F. Nobis, L. Ögretmen, M. Rowold, F. Sauerbeck, T. Stahl, R. Trauth, M. Lienkamp, and B. Lohmann, "Indy autonomous challenge—Autonomous race cars at the handling limits," in *Proc. 12th Int. Munich Chassis Symp.*, P. Pfeffer, Ed. Berlin, Germany: Springer, 2022, pp. 163–182.
[3] S. Ammoun and F. Nashashibi, "Real time trajectory prediction for collision risk estimation between vehicles," in *Proc. IEEE 5th Int. Conf. Intell. Comput. Commun. Process.*, Aug. 2009, pp. 417–422.
[4] G. Xie, H. Gao, L. Qian, B. Huang, K. Li, and J. Wang, "Vehicle trajectory prediction by integrating physics- and maneuver-based approaches using interactive multiple models," *IEEE Trans. Ind. Electron.*, vol. 65, no. 7, pp. 5999–6008, Jul. 2018.
[5] N. Deo, A. Rangesh, and M. M. Trivedi, "How would surround vehicles move? A unified framework for maneuver classification and motion prediction," *IEEE Trans. Intell. Vehicles*, vol. 3, no. 2, pp. 129–140, Jun. 2018.
[6] R. Schubert, E. Richter, and G. Wanielik, "Comparison and evaluation of advanced motion models for vehicle tracking," in *Proc. 11th Int. Conf. Inf. Fusion*, 2008, pp. 1–6.
[7] P. Karle, M. Geisslinger, J. Betz, and M. Lienkamp, "Scenario understanding and motion prediction for autonomous vehicles—Review and comparison," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 10, pp. 16962–16982, Oct. 2022, doi: 10.1109/TITS.2022.3156011.
[8] J. Betz, H. Zheng, A. Liniger, U. Rosolia, P. Karle, M. Behl, V. Krovi, and R. Mangharam, "Autonomous vehicles on the edge: A survey on autonomous vehicle racing," *IEEE Open J. Intell. Transp. Syst.*, vol. 3, pp. 458–488, 2022, doi: 10.1109/OJITS.2022.3181510.
[9] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Cambridge, U.K.: Cambridge Univ. Press, 2006.
[10] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Trans. ASME-J. Basic Eng.*, vol. 82, pp. 35–45, 1960.
[11] C. Barrios and Y. Motai, "Improving estimation of vehicle's trajectory using the latest global positioning system with Kalman filtering," *IEEE Trans. Instrum. Meas.*, vol. 60, no. 12, pp. 3747–3755, Dec. 2011, doi: 10.1109/TIM.2011.2147670.
[12] M. Koschi and M. Althoff, "SPOT: A tool for set-based prediction of traffic participants," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2017, pp. 1686–1693, doi: 10.1109/IVS.2017.7995951.
[13] C. Pek, S. Manzinger, M. Koschi, and M. Althoff, "Using online verification to prevent autonomous vehicles from causing accidents," *Nature Mach. Intell.*, vol. 2, no. 9, pp. 518–528, Sep. 2020, doi: 10.1038/s42256-020-0225-y.

[14] T. Stahl, M. Eicher, J. Betz, and F. Diermeyer, "Online verification concept for autonomous vehicles—Illustrative study for a trajectory planning module," in *Proc. IEEE 23rd Int. Conf. Intell. Transp. Syst. (ITSC)*, Sep. 2020, pp. 1–7, doi: 10.1109/ITSC45102.2020.9294703.

[15] A. Houenou, P. Bonnifait, V. Cherfaoui, and W. Yao, "Vehicle trajectory prediction based on motion model and maneuver recognition," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Nov. 2013, pp. 4363–4369.

[16] H. M. Mandalia and M. D. D. Salvucci, "Using support vector machines for lane-change detection," in *Proc. Hum. Factors Ergonom. Soc. Annu. Meeting*, vol. 49, no. 22. Los Angeles, CA, USA: Sage, 2005, pp. 1965–1969.

[17] W. Yuan, Z. Li, and C. Wang, "Lane-change prediction method for adaptive cruise control system with hidden Markov model," *Adv. Mech. Eng.*, vol. 10, no. 9, 2018, Art. no. 1687814018802932.

[18] J. Li, W. Zhan, Y. Hu, and M. Tomizuka, "Generic tracking and probabilistic prediction framework and its application in autonomous driving," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 9, pp. 3634–3649, Sep. 2020.

[19] M. Bicego, V. Murino, and M. Figueiredo, "Similarity-based classification of sequences using hidden Markov models," *Pattern Recognit.*, vol. 37, no. 12, pp. 2281–2291, Dec. 2004.

[20] A. Khosroshahi, E. Ohn-Bar, and M. M. Trivedi, "Surround vehicles trajectory analysis with recurrent neural networks," in *Proc. IEEE 19th Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2016, pp. 2267–2272.

[21] B. T. Morris and M. M. Trivedi, "Trajectory learning for activity understanding: Unsupervised, multilevel, and long-term adaptive approach," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 11, pp. 2287–2301, Nov. 2011.

[22] M. Krüger, A. S. Novo, T. Nattermann, and T. Bertram, "Probabilistic lane change prediction using Gaussian process neural networks," in *Proc. IEEE Intell. Transp. Syst. Conf. (ITSC)*, Oct. 2019, pp. 3651–3656.

[23] M.-F. Chang, J. Lambert, P. Sangkloy, J. Singh, S. Bak, A. Hartnett, D. Wang, P. Carr, S. Lucey, D. Ramanan, and J. Hays, "Argoverse: 3D tracking and forecasting with rich maps," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 8748–8757.

[24] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuScenes: A multimodal dataset for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11621–11631.

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

[26] D. Ridel, N. Deo, D. Wolf, and M. Trivedi, "Scene compliant trajectory forecast with agent-centric spatio-temporal grids," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 2816–2823, Apr. 2020.

[27] A. Sadeghian, F. Legros, M. Voisin, R. Vesel, A. Alahi, and S. Savarese, "Car-Net: Clairvoyant attentive recurrent network," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 151–167.

[28] J. Li, H. Ma, and M. Tomizuka, "Conditional generative neural system for probabilistic trajectory prediction," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 6150–6156.

[29] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-C. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 802–810.

[30] J. Mercat, T. Gilles, N. Zoghby, G. Sandou, D. Beauvois, and G. P. Gil, "Multi-head attention for multi-modal joint vehicle motion forecasting," in *Proc. IEEE Int. Conf. Robot. Autom.*, May/Aug. 2020, pp. 9638–9644.

[31] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, "Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data," in *Computer Vision—ECCV 2020*, A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds. Cham, Switzerland: Springer, 2020, pp. 683–700.

[32] Z. Zhao, H. Fang, Z. Jin, and Q. Qiu, "GISNet: Graph-based information sharing network for vehicle trajectory prediction," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2020, pp. 1–7, doi: 10.1109/IJCNN48605.2020.9206770.

[33] J. Gao, C. Sun, H. Zhao, Y. Shen, D. Anguelov, C. Li, and C. Schmid, "VectorNet: Encoding HD maps and agent dynamics from vectorized representation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11525–11533.

[34] M. Geisslinger, P. Karle, J. Betz, and M. Lienkamp, "Watch-and-learn-net: Self-supervised online learning for probabilistic vehicle trajectory prediction," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2021, pp. 869–875, doi: 10.1109/SMC52423.2021.9659079.

[35] A. Rudenko, L. Palmieri, and K. O. Arras, "Joint long-term prediction of human motion using a planning-based social force approach," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 4571–4577.

[36] L. Sun, W. Zhan, and M. Tomizuka, "Probabilistic prediction of interactive driving behavior via hierarchical inverse reinforcement learning," in *Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2018, pp. 2111–2117.

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

[38] T. Fernando, S. Denman, S. Sridharan, and C. Fookes, "Deep inverse reinforcement learning for behavior prediction in autonomous driving: Accurate forecasts of vehicle motion," *IEEE Signal Process. Mag.*, vol. 38, no. 1, pp. 87–96, Jan. 2021.

[39] M. Bansal, A. Krizhevsky, and A. Ogale, "ChauffeurNet: Learning to drive by imitating the best and synthesizing the worst," in *Proc. Robot., Sci. Syst. XV*, A. Bicchi, H. Kress-Gazit, and S. Hutchinson, Eds., Jun. 2019, pp. 1–10, doi: 10.15607/RSS.2019.XV.031.

[40] A. Kuefler, J. Morton, T. Wheeler, and M. Kochenderfer, "Imitating driver behavior with generative adversarial networks," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2017, pp. 204–211.

[41] L. Ögretmen, M. Rowold, M. Ochsenius, and B. Lohmann, "Smooth trajectory planning at the handling limits for oval racing," *Actuators*, vol. 11, no. 11, p. 318, Nov. 2022, doi: 10.3390/act11110318.

[42] 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, p. 319, Nov. 2022, doi: 10.3390/act11110319.

[43] J. Betz, T. Betz, F. Fent, M. Geisslinger, A. Heilmeier, L. Hermansdorfer, T. Herrmann, S. Huch, P. Karle, M. Lienkamp, B. Lohmann, F. Nobis, L. Ögretmen, M. Rowold, F. Sauerbeck, T. Stahl, R. Trauth, F. Werner, and A. Wischnewski, "TUM autonomous motorsport: An autonomous racing software for the indy autonomous challenge," *J. Field Robot.*, vol. 40, no. 4, pp. 783–809, Jun. 2023, doi: 10.1002/rob.22153.

[44] 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 Int. J. Connected Automated Vehicles*, vol. 6, no. 3, 2023. [Online]. Available: https://www.sae.org/publications/technical-papers/content/12-06-03-0018/, doi: 10.4271/12-06-03-0018.

[45] A. Heilmeier, A. Wischnewski, L. Hermansdorfer, J. Betz, M. Lienkamp, and B. Lohmann, "Minimum curvature trajectory planning and control for an autonomous race car," *Vehicle Syst. Dyn.*, vol. 58, no. 10, pp. 1497–1527, Oct. 2020.

[46] 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, p. 1523, Feb. 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/4/1523., doi: 10.3390/s21041523.

[47] N. Deo and M. M. Trivedi, "Convolutional social pooling for vehicle trajectory prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2018, pp. 1549–1557, doi: 10.1109/CVPRW.2018.00196.

[48] Energy Systems Network. (2019). *Indy Autonomous Challenge Rules v5nov2019*. [Online]. Available: https://static1.squarespace.com/static/5da73021d0636f4ec706fa0a/t/5dc0680c41954d4ef41ec2b2/1572890638793/Indy+Autonomous+Challenge+Ruleset+-+v5NOV2019+%282%29.pdf

[49] P. Karle, F. Fent, S. Huch, F. Sauerbeck, and M. Lienkamp, "Multi-modal sensor fusion and object tracking for autonomous racing," *IEEE Trans. Intell. Vehicles*, early access, May 1, 2023, doi: 10.1109/TIV.2023.3271624.

[50] F. Nogueira. (2014). *Bayesian Optimization: Open Source Constrained Global Optimization Tool for Python*. [Online]. Available: https://github.com/fmfn/BayesianOptimization

PHILLIP KARLE (Graduate Student Member, IEEE) received the B.Sc. and M.Sc. degrees from the Technical University of Munich (TUM), Munich, Germany, in 2017 and 2019, respectively, where he is currently pursuing the Ph.D. degree in mechanical engineering with the Institute of Automotive Technology. His research interests include multi-object tracking, scenario understanding, motion prediction, and related applications for autonomous driving with a focus on real-world applications.

FERENC TÖRÖK received the B.Sc. degree from the Budapest University of Technology and Economics, Hungary, in 2019, and the M.Sc. degree from the Technical University of Munich, Germany, in 2021. Afterward, he joined Argo AI as a Prediction Software Engineer in Munich. His research interests include motion prediction and multi-agent decision making for autonomous driving.

MAXIMILIAN GEISSLINGER received the B.Sc. and M.Sc. degrees from the Technical University of Munich (TUM), Munich, Germany, in 2016 and 2019, respectively, where he is currently pursuing the Ph.D. degree in mechanical engineering with the Institute of Automotive Technology. His main research interests include ethics in autonomous driving, fair distribution of risk and quantification of uncertainties, and related applications for autonomous driving.

MARKUS LIENKAMP (Member, IEEE) received the Ph.D. degree from TU Darmstadt, in 1995. He studied mechanical engineering with TU Darmstadt and Cornell University. He researches in the area of autonomous vehicle with the objective to create an open-source software platform. Afterward, he joined an International Trainee Program with Volkswagen and worked in a joint venture between Ford and Volkswagen in Portugal, led the Brake Testing Department of the VW commercial vehicles. He was later appointed the Head of the Electronics and Vehicle Research Department. He has been a Professor with the Institute of Automotive Technology, Technical University of Munich (TUM), since November 2009. He has been heading the Chair of Automotive Technology with TUM, since November 2009.