Learning Interaction-Aware Probabilistic Driver Behavior Models from Urban Scenarios

Jens Schulz[1], Constantin Hubmann[1], Nikolai Morin[2], Julian Löchner[1], and Darius Burschka[2]

***Abstract*— Human drivers have complex and individual behavior characteristics which describe how they act in a specific situation. Accurate behavior models are essential for many applications in the field of autonomous driving, ranging from microscopic traffic simulation, intention estimation and trajectory prediction, to interactive and cooperative motion planning. Designing such models by hand is cumbersome and inaccurate, especially in urban environments, with their high variety of situations and the corresponding diversity in human behavior. Learning how humans act from recorded scenarios is a promising way to overcome these problems. However, predicting complete trajectories at once is challenging, as one needs to account for multiple hypotheses and long-term interactions between multiple agents. In contrast, we propose to learn Markovian action models with deep neural networks that are conditioned on a driver's route intention (such as turning left or right) and the situational context. Step-wise forward simulation of these models for the different possible routes of all agents allows for multi-modal and interaction-aware scene predictions at arbitrary road layouts. Learning to predict only one time step ahead given a specific route reduces learning complexity, such that simpler and faster models are obtained. This enables the integration into particle-based algorithms such as Monte Carlo tree search or particle filtering. We evaluate the learned model both on its own and integrated into our previously presented dynamic Bayesian network for intention estimation and show that it outperforms our previous hand-tuned rule-based model.**

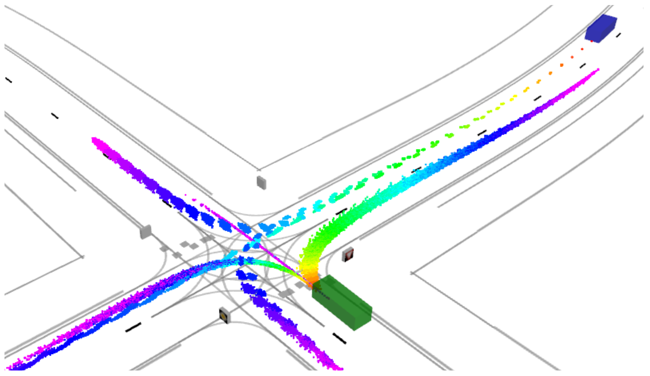


Fig. 1. Learned probabilistic and interaction-aware driver behavior models iteratively applied to generate possible scene predictions (colors indicate prediction horizon of up to 7 s). The green vehicle yields for the hypotheses of turning left/going straight, but is allowed to drive for turning right. The variance in the driver's actions highly depends on the situation, which can be seen in a higher lateral uncertainty for turning maneuvers.

I. Introduction

Driver behavior models are frequently used as transition models for iterative trajectory prediction, filtering-based intention estimation, forward simulation based motion planning as well as for microscopic simulation and traffic flow modeling. Common approaches consist of rule-based models which define a (in most cases) deterministic mapping from situation and intention of a driver to an action resulting in a state transition to the subsequent time step. Although simple approaches such as the Intelligent Driver Model (IDM) [1] are well suited for high-level traffic flow modeling or deterministic trajectory prediction in car following scenarios, they are neither capable of capturing the complex decision making process of human drivers in more diverse scenarios, nor of representing the variance of the potential future behavior. Especially in urban environments, the magnitude of different influences makes the modeling of driver behavior by hand cumbersome and error-prone.

Data-driven approaches are more promising to achieve high-accuracy models that are able to represent subtle nuances of driver behavior. In the area of trajectory prediction, typically a mapping from current state to complete trajectories of several seconds into the future is learned. However, these relationships are in general hard to learn: the prediction of a traffic scene with multiple agents needs to incorporate possible interactions and account for both low-level action uncertainty (such as lateral position in a lane) as well as high-level decision uncertainty, such as the route an agent will follow at an intersection. Incorporating all this within the learning process requires both complex models and lots of data. Existing work either learns solely high-level intentions [2], does not consider urban scenarios [3] or tackles the whole prediction problem with complex, nested neural network based models [4].

In contrast, we propose to learn probabilistic and interaction-aware Markovian behavior models that are conditioned on the driver's route intention using deep neural networks. Such models allow the integration into various kinds of state-of-the-art algorithms that commonly still rely on hand-tuned models: forward simulation based motion planning algorithms (e.g. Monte Carlo tree search (MCTS) [5], partially observable Markov decision processes (POMDPs) [6]), or intention estimation and trajectory prediction algorithms (e.g. dynamic Bayesian networks (DBNs) [7]). As many of these algorithms are sampling-based, the models might be called thousands of times per time step. Thus, our focus is on simple and fast models that are still able to accurately capture the variety that is present in urban

[1]Jens Schulz, Constantin Hubmann, and Julian Löchner are with BMW Group, Munich, Germany {jens.schulz | constantin.hubmann | julian.loechner}@bmw.de

[2]Nikolai Morin and Darius Burschka are with the Department of Computer Science, Technical University of Munich, Germany nikolaimorin@gmail.com, burschka@tum.de

scenarios. By learning the mean and variance of a Gaussian action distribution, the presented models account for situation-dependent variance in human actions.

We compare different network architectures and show that even simple feed-forward models are able to outperform the hand-tuned rule-based model from [7], providing a more accurate trajectory prediction using forward simulation. The model is able to learn lane following, distance keeping, yielding to other traffic participants at intersections, stopping at red lights as well as to even capture subtleties such as cutting curves and curvature dependent lateral uncertainty (see Fig. 1). To showcase the applicability of such a learned model, we integrate it into our previously presented DBN [7] for driver intention estimation and trajectory prediction and compare it to the previous rule-based action model.

II. Related Work

The need for accuracy and detail in driver behavior modeling typically varies depending on the purpose of application. Models used for traffic flow analysis tend to be less detailed, as individual driver behavior patterns do not matter as much, whereas models used for estimating driver intentions, or for predicting how humans react to an autonomous vehicle, should be able to capture human behavior patterns in more detail.

A. Rule-based Behavior Models

The traffic simulator SUMO [8] comes with different behavior models built in. It is distinguished between car-following, lane-changing, and junction models. The most known car-following models are the so-called *Krauß*-model [9] and the intelligent driver model (IDM) [1]. Typically, these models have free parameters, such as desired time headway or acceleration/deceleration ability and define a deterministic mapping from relative distances and velocities to accelerations. Most of the models follow the concept of driving as fast as possible while ensuring complete safety, meaning that each agent will always be able to avoid a collision with its preceding agent. This is a strong assumption which typically does not hold in real traffic. Similarly, the lane changing and junction models can be parameterized, e.g., in terms of maximum lateral acceleration, minimum time gap or tendency to ignoring the right of way. Thanks to its simplicity, the IDM is widely applied for driver intention estimation (e.g., [7], [10]) and for modeling how others react to specific plans of an autonomous vehicle (e.g., during forward simulation using Monte Carlo tree search [5]).

Although these heuristics-based models are well suited for traffic simulation, they tend to be not detailed and realistic enough for accurate prediction of humans. Furthermore, they consist of multiple interdependent parameters that are cumbersome to tune by hand.

In our previous work [7], we define a set of influence-based action models, each determining a range of reasonable acceleration given a specific influence, such as a *preceding vehicle*, the *road curvature*, or an upcoming *traffic light*. Besides applying the IDM for the influence of the *preceding vehicle*, we propose new heuristics-based models for stopping at stop signs and red lights, slowing down before curvatures and approaching gaps at intersections. These additional components successfully extend the usage of rule-based models to complex urban scenarios. However, the design and tuning of all of the aforementioned models by hand is cumbersome and they are not able to distinguish minor subtleties in human behavior. Furthermore, they do not account for the fact that the variance of human behavior is situation-dependent as well. This model serves as a baseline for the evaluation of the learned models and is referred to as *rule-based* model.

B. Data-driven Behavior Models

Learning complete trajectories, i.e., multiple time steps at once, is a common approach, as it allows to consider long-term deviations within the loss function. To account for interactions between multiple agents, Alahi et al. [11] combine what they call social pooling with LSTMs, allowing to create relationships between an agent and its close neighborhood. They show promising results in the area of pedestrian prediction in crowded spaces.

A recent deep-learning-based multi-agent trajectory prediction framework called DESIRE [4] aims to achieve interaction-awareness, account for the multi-modality of the future (e.g. induced by different possible routes) and achieve long-term accuracy–all within one training procedure. The model consists of multiple modules handling the trajectory sample generation (based on a conditional variational auto-encoder), the trajectory ranking and refinement (based on inverse optimal control) and the scene context fusion (using a convolutional neural network encoded scene context). Although it shows very promising results, such nested models typically need lots of training data and may be too complex to be integrated into sampling-based filtering and forward simulation frameworks.

In [12], a Bayesian network estimates route intentions of drivers at intersections based on learned conditional probability tables that define the kinematic state transitions. Discretizing the state space results in low complexity, but also does not allow for accurate, fine-grain models.

Wheeler et al. [13] present a survey on learned probabilistic driver behavior models for highways scenarios. Comparing different models including random forests, linear/static Gaussian and linear/discrete Bayesian on the context classes of *free-flow*, *car following* and *lane change*, they found that mixture regression and linear Bayesian achieve the best results, depending on the evaluation metric.

In [14], learned context-dependent action models of traffic participants based on random forests are embedded into a DBN to estimate the state of the current situation and predict the future motion of drivers. As an outlook, they highlight the possible benefits of conditioning the learning process on driver intentions.

Lenz et al. [3] compare different deep neural network architectures for Markovian motion prediction in highway scenarios. As they do not condition on driver intentions such as lane changing or lane keeping, they model their actions as Gaussian mixture distribution to account for future multimodality. They find that a fully connected feed-forward network outperforms recurrent architectures on the domain at hand. As their presented architecture achieves promising results, we include it in the evaluation of this paper.

In contrast to the presented literature, we propose to learn neural network based probabilistic and context-dependent action models that are conditioned on a driver's route intention, which are able to cope with the high uncertainty and situational complexity present in urban environments.

III. Problem Statement

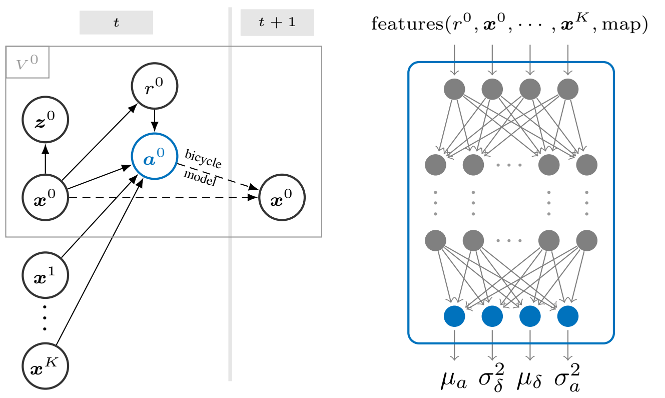
A traffic scene consists of a set of agents $\mathcal{V} = \{V^0, \cdots, V^K\}$, with $K \in \mathbb{N}_0$, in a static environment (map) with discrete time, continuous state, and continuous action space. The map consists of a road network with topological, geometric and infrastructure (yield lines, traffic signs, etc.) information as well as the prevailing traffic rules. At time step t, each of the agents is represented by its route intentions r^i_t and its kinematic state $\boldsymbol{x}^i_t = [x^i_t, y^i_t, \psi^i_t, v^i_t]^\top$ comprising the Cartesian position, heading, and absolute velocity. The agents' lengths and widths are considered to be given, but for the sake of brevity, are not included within $\boldsymbol{x}^i$. The route intention r^i_t defines a path through the road network the agent desires to follow (see Sec. IV-A for detailed definition). At each time step, each agent executes an action $\boldsymbol{a}^i_t = [a^i_t, \delta^i_t]^\top$ comprising the longitudinal acceleration and the steering angle. This action depends on the agent's route intention, the map and the kinematic states of all agents, transforming the current kinematic state $\boldsymbol{x}^i_t$ to the new state $\boldsymbol{x}^i_{t+1}$. Noisy measurements $\boldsymbol{z}^i_t = [z^i_{x,t}, z^i_{y,t}, z^i_{\theta,t}, z^i_{v,t}]^\top$ are used to update the belief of the agent's state. The data association (detection and tracking of objects) is considered to be given as it is handled by another module. Thus, high-level cubic objects represent the measurements of the single vehicles.

The objective of this work is to derive an accurate action model $p(\boldsymbol{a}^i|r^i, \boldsymbol{x}^0, \cdots, \boldsymbol{x}^K, \text{map})$ which allows to predict the next kinematic state of an agent as close as possible given its current kinematic state and its route intention. As this model is intended to be integrated as a probabilistic transition model into sampling based algorithms, the input to the model is deterministic (a sample of the belief), whereas the output is a probability distribution over actions, from which one can again draw samples, if desired. In this work, the action is modeled to be normally distributed given a specific route intention and the current situational context:

$$p(\boldsymbol{a}^i|r^i, \boldsymbol{x}^0, \cdots, \boldsymbol{x}^K, \text{map}) = \mathcal{N}\left(\begin{bmatrix}\mu_a \\ \mu_\delta\end{bmatrix}, \begin{bmatrix}\sigma_a^2 & 0 \\ 0 & \sigma_\delta^2\end{bmatrix}\right) \quad (1)$$

IV. Approach

We model the development of a traffic situation as a Markov process consisting of multiple interacting agents. The action of one agent at a specific time step t is modeled to be independent of the actions of the other agents at the same



t
$t+1$
V^0
r^0
z^0
a^0
bicycle model
x^0
x^0
x^1
x^K
features$(r^0, x^0, \cdots, x^K, \text{map})$
μ_a σ_δ^2 μ_δ σ_a^2

Fig. 2. Excerpt of the DBN (left) from [7] showing the dependencies of an agent's action on its route intention and the states of the surrounding agents. We learn this Markovian action model using a neural network (right), which defines the probabilistic mapping from features to a Gaussian action distribution comprising acceleration a and steering angle δ. The features are determined given the parents of the action node and the map object.

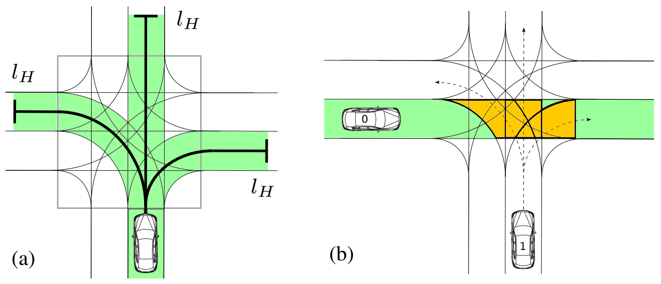


Fig. 3. (a): Possible routes at an intersection (green). (b): Conflict areas (yellow) from V^0's perspective for going straight, considering all possible route intentions of V^1. [7]

time step given their current kinematic states. This is shown in the left part of Fig. 2: the action $\boldsymbol{a}^i$ of agent V^i only depends on its route intention r^i and the kinematic states $[\boldsymbol{x}^0, \cdots, \boldsymbol{x}^K]$ of all agents, but not on other agents' actions. Thus, all agents can be predicted independently from t to $t+1$. For small time steps, this is a mild assumption, as one can assume that drivers don't know the future actions of other drivers when deciding for an action. As the context is updated in each time step (given the one-step prediction of each agent of the last time step), an interdependency between multiple agents' trajectories emerges *over time*.

In contrast to existing rule-based action models, we propose to learn this probabilistic mapping from context and route intention to action distribution with deep neural networks (right part of Fig. 2).

A. Route Intention and Conflict Areas

The agent's route intention $r^i \in \mathcal{R}^i$ serves as a path that guides its behavior. It is represented by a sequence of consecutive lanes in the road network. The set of possible routes $\mathcal{R}^i$ is determined given the current lane matching and a specified metric horizon l_H using breadth-first search in the topological map (see Fig. 3(a)). As each route has a different geometry, may imply different traffic rules and relations to other agents, the desired route strongly influences a driver's actions.

The conditioning on a driver's route intention mainly serves two purposes: Firstly, it reduces the neural network's prediction complexity by eliminating the need to predict for multiple route hypotheses. Thus, the learning algorithm does not have to cope with the multi-modality induced by different route options. Furthermore, the varying number of possible routes (depending on the road topology) is handled outside of the neural network, which would be hard to model with a fixed input and output sized neural network. Secondly, it allows to define relevant features along the planned path, such as upcoming road curvature or longitudinal distances to stop lines, and to build simplified relationships between agents on complex road layouts. The routes of two agents are related by dividing them into parts that either *merge*, *diverge*, *cross*, are *identical*, or have no relevant relation at all. Different road junction types such as roundabouts, intersections or highway entrances can thus be broken down into these types of relations, allowing for a better generalization.

To derive interaction features of agents at intersections, we introduce the notion of conflict areas: Given two agents on two routes, their conflict area is defined by the intersecting set of the areas of both routes, i.e., the area in which their lanes overlap. We assume an agent doesn't know which route other agents are going to follow, thus, all possible conflict areas are considered (see Fig. 3(b)). For each conflicting agent, the possible conflict areas are merged and the distances of both agents to entry and exit of this area are determined and represented as features. Furthermore, the other agent's velocity and the right of way are used as features (see Sec. IV-D).

B. Motion Model

Instead of directly learning a state transition model, we restrict the neural network to learn a two dimensional action distribution comprising acceleration a and steering angle δ. This reduces learning complexity and enforces the non-holonomic vehicle constraints. The kinematic state transition is defined deterministically given these actions using the kinematic bicycle model [15]

$$\dot{\boldsymbol{x}} = \begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\psi} \\ \dot{v} \end{pmatrix} = \begin{pmatrix} v\cos(\psi+\beta) \\ v\sin(\psi+\beta) \\ \frac{v}{l_r}\sin(\beta) \\ a_t \end{pmatrix}, \tag{2}$$

with $\beta = \arctan\left(\frac{l_r}{l_f+l_r}\tan(\delta)\right)$. The parameters l_f and l_r define the distances from center of gravity to the front and rear axis respectively. They are determined by minimizing the motion reconstruction error (see Sec. IV-C). In between two discrete time steps, it is assumed that the acceleration and the steering angle are kept constant.

C. Target Generation

The inverse of the bicycle model allows the calculation of the acceleration and steering angle given two consecutive kinematic states $\boldsymbol{x}_t$ and $\boldsymbol{x}_{t+1}$. The targets can thus be determined given a sequence of states. As the system of equations is over-determined, we ignore the components of x and y and rely on v and ψ to generate the targets:

$$a_t = \frac{v_{t+1} - v_t}{\Delta T} \tag{3}$$

$$\delta_t = \operatorname{sgn}\left(\frac{\Delta\psi}{\bar{v}}\right)\arctan\left(\frac{l_f + l_r}{\sqrt{(\frac{\bar{v}}{\Delta\psi})^2 - l_r^2}}\right), \tag{4}$$

with $\bar{v} = \frac{v_t + v_{t+1}}{2}$. The steering angle is considered to be zero if an agent stands still or the term under the square root is negative. Before calculating the targets, the data sequences are resampled to $\Delta T = 0.2\,\mathrm{s}$ by linearly interpolating the kinematic states and features. To reduce the negative impact of noise in the targets on the learning process, the velocity and yaw angle are smoothed with moving mean and median (window size $1\,\mathrm{s}$) before applying the inverse motion model (see first plot of Fig. 4 for a comparison of reconstructed acceleration with filtered and unfiltered velocity).

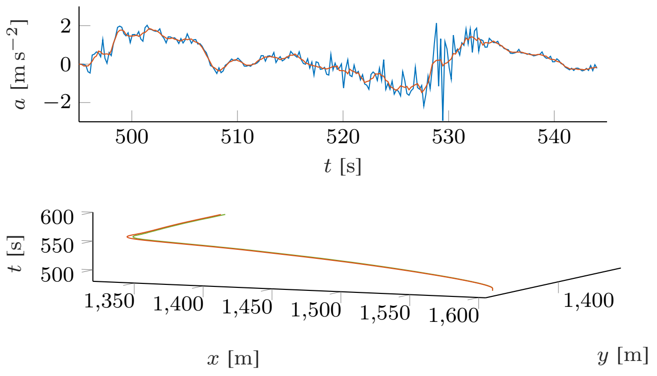


Fig. 4. First plot: determination of acceleration using inverse motion model with unfiltered (blue) and filtered (red) velocity and yaw-angle (real data). The filter applies a moving mean and a moving average with window size 1 s. Second plot: corresponding motion reconstruction (red) and original motion (green). As the reconstruction with unfiltered velocity and yaw-angle does not differ noticeably from the filtered one, it is not depicted.

To analyze the accuracy of the motion model and to derive the parameters l_f and l_r, the kinematic states can be reconstructed using the actions and the forward motion model and compared to the original trajectories. A typical example of the original data and the reconstructed states can be seen in the second plot of Fig. 4.

For the learning procedure, we consider the smoothed trajectories (and corresponding targets) and the actually driven route to be ground truth, thus we can learn with complete data and do not need to utilize expectation-maximization.

D. Feature Generation

The input features of the neural network summarize the current situational context including the geometry of the upcoming road, traffic rules and road infrastructure, and the most relevant surrounding agents. Most of these features can only be defined given a specific route intention, such as the upcoming road curvature, the conflict areas (see IV-A) and the corresponding right of way, or which vehicle is considered to be the directly preceding one. The route intention is thus described by these features, rather than by

TABLE I
FEATURES

Predicted Vehicle	
velocity of predicted vehicle V^i	v^i
lateral position in lane	$d_{\text{lat},0}$
Route	
curvature values ahead at distances	$[c_0, c_5, \cdots, c_H]$
reasonable acc given curvature ahead	a_{curv}
relative angle to point on centerline at distances	$[\phi_0, \phi_1, \cdots, \phi_{15}]$
relative angle to direction of centerline	γ_0
width of lane	w_0
Traffic Rules	
speed limit	v_{speedlim}
distance to next traffic light	$d_{\text{trafficlight}}$
next traffic light state	$s_{\text{trafficlight}}$
distance to next stop line	d_{stopline}
distance to next yield line	$d_{\text{yieldline}}$
distance to next intersection	$d_{\text{intersection}}$
whether always right of way at next intersection	$\text{row}_{\text{always}}$
Interaction	
velocity of preceding agent V^p	v^p
distance to preceding agent V^p	d^p
velocity of closest conflicting agent V^c	v^c
distance V^c to conflict area (entry)	d^c_{entry}
distance V^c to conflict area (exit)	d^c_{exit}
distance V^i to conflict area (entry)	d^i_{entry}
distance V^i to conflict area (exit)	d^i_{exit}
right of way for conflict	$\text{row}^{i,c}$

discrete options such as *right turn*. This allows an arbitrary road layout with different number of possible routes.

In this work, we found that the directly preceding agent and the closest conflicting agent are the two most influencing agents. To reduce complexity, only these two other agents are included within the feature set. Considering more and potentially even a varying number of agents should be part of future research. A complete list of used features is given in Tab. I. Different combinations of features are tried during evaluation to determine their importance (see Sec. V).

There are two features that are not self-explanatory: the so called *reasonable acceleration* given the upcoming curvature a_{curv} is a pre-computed feature based on domain knowledge. It is intended to subsume the set of upcoming curvature values and is calculated according to a desired maximum lateral acceleration. Given this lateral acceleration, one can determine the maximum acceleration for one time step that still allows the agent to brake in time. This heuristic is also used in the rule-based model for the velocity adaptation depending on the curvature (see [7] for more details). The boolean feature $\text{row}_{\text{always}}$ specifies whether an agent does always have right of way on its route at the next intersection (no matter if other agents are present nor what routes they are going to take). We added this feature, as agents on priority lanes tend to not slow down before intersections at all, whereas agents that might have to yield slow down even if there are no other agents around.

As the model is conditioned on the driver's intended route, this intention has to be known for training in order to set the features accordingly. We assume drivers do not change their minds about the routes they desire to follow. This allows to automatically label the route intention after observing which routes were actually taken. Tracks of agents that were lost before they have completely traversed an intersection are disregarded for training.

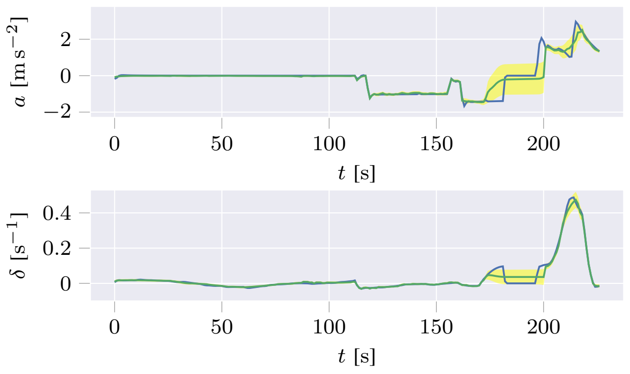


Fig. 5. Heteroscedastic variance in human driver behavior: depicted is a validation plot of a learned model with mean (green) and variance (yellow) and one trajectory of the validation dataset (blue). It can be seen that the variance of the learned actions strongly depends on the situation.

E. Loss function

To be able to integrate the action model into various sampling based algorithms, it is supposed to learn a mapping from a deterministic set of features to a probability distribution of actions. Our data shows that the variance in driver behavior strongly depends on the situation. Thus, we do not only learn the mean of the actions, but also learn the context-dependent variance of the action distribution. A validation plot showing this so-called heteroscedastic variance can be seen in Fig. 5. We model this distribution to be a unimodal Gaussian given a specific route hypothesis. Therefore, the overall belief including the uncertainty about the route intention will still be multimodal. For the sake of problem simplification, the covariance between acceleration and steering angle is assumed to be zero.

The loss function given the predicted Gaussian with mean $\boldsymbol{\mu} = [\mu_a, \mu_\delta]^\top$ and covariance matrix $\boldsymbol{\Sigma} = \text{diag}(\sigma_a^2, \sigma_\delta^2)$ and the targets $\boldsymbol{t} = [a^t, \delta^t]^\top$ is given by the negative log likelihood

$$l_{\text{NLL}}(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{t}) = \frac{1}{2}(\boldsymbol{\mu} - \boldsymbol{t})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu} - \boldsymbol{t}) + \log(\sqrt{|\boldsymbol{\Sigma}|}), \quad (5)$$

with $|\boldsymbol{\Sigma}|$ being the determinant of the covariance matrix. This loss enables to learn both the mean and variance in one training procedure.

V. EVALUATION

A. Models and Hyperparamters

We compare the following types of neural networks to show how they are suited for the task of one-step motion prediction: *Linear* fully connected, long short-term memory (*LSTM*), gated recurrent unit (*GRU*), and the architecture presented by *Lenz et al.* [3], which was originally introduced as a driver behavior model in highway scenarios. The last

TABLE II
HYPERPARAMETERS

Parameter	Linear	LSTM	GRU	Lenz
f_{act}	ReLU	–	–	ELU
N_{layers}	4	2	2	5
$N_{\text{n/layer}}$	274	274	274	400
$p_{\text{drop.}}$	0.06	0.2	0.3	0.5
$m_{\text{batchn.}}$	0.3	–	–	–

layer of each of the network types is a fully connected linear layer that outputs the action distribution parameters $\mu_a, \mu_\delta, \sigma_a^2, \sigma_\delta^2$. The outputs for the variances are transformed by an exponential function before calculation of the loss to ensure positive values.

The best found hyperparameters for the presented models as well as the parameters used by Lenz et al. are presented in Tab. II. Those parameters are used for the remainder of the evaluation. Furthermore, the Adam optimizer [16] is chosen, using a batch size of 1024, a learning rate of 0.001 and a sequence length for the recurrent architectures of 3 s.

The different architectures are furthermore compared to the *rule-based* action model from [7] (see Sec. II-A). This model is able to slow down before curvatures, keep appropriate distances to preceding vehicles, stop at traffic lights, stop lines and before intersections (if yielding is required) and implements a gap-approach for intersection crossing. The parameters of this model were tuned by hand using different urban scenarios.

B. Training and Validation Data

Real driving data is recorded with a measuring vehicle with GPS/INS based localization and dynamic occupancy grid based object tracking using lidar and radar sensors [17]. This dataset, denoted as $\mathcal{D}^{\text{real}}$, consists of 40 minutes of urban scenarios including both roundabouts and unsignalized intersections and is split into 30 minutes for training and 10 minutes for validation.

As the real driving data is very limited and not as diverse (e.g. does not contain traffic lights), we additionally recorded simulations using a proprietary traffic simulator. This dataset, denoted as $\mathcal{D}^{\text{sim}}$, consists of 0.465 hours of naturalistic driving data including 40 agents randomly traversing a different, and more diverse urban environment. It includes traversing signalized as well as unsignalized intersections (both T-junction and 4-way junctions) and some lane changes on multi-lane intersections. This results in 18.6 hours of data, split into 10 hours for training and 8.6 hours for validation.

The map which is used to determine the possible routes and features is based on *OpenDRIVE* [18]. The agents of the simulation are controlled using the driver model of [19].

C. Results

1) Comparison of Model Architectures: The different model architectures are compared using the simulation dataset $\mathcal{D}^{\text{sim}}$, as it contains more diverse scenes (e.g. including traffic lights or lane changes, not present in the real driving dataset), thus allowing for a better comparison.

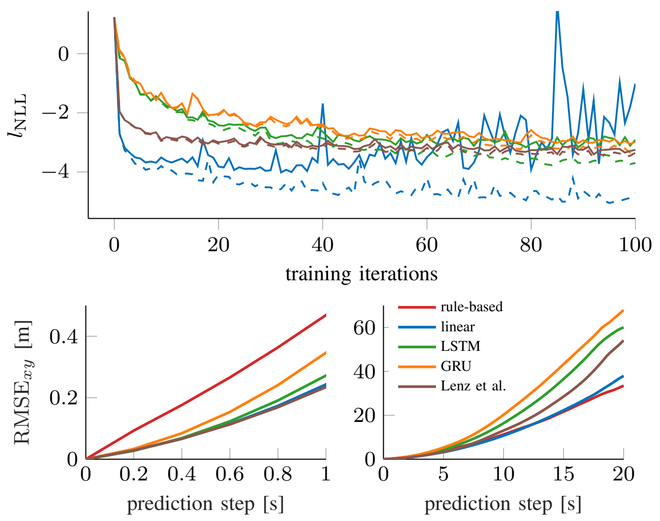
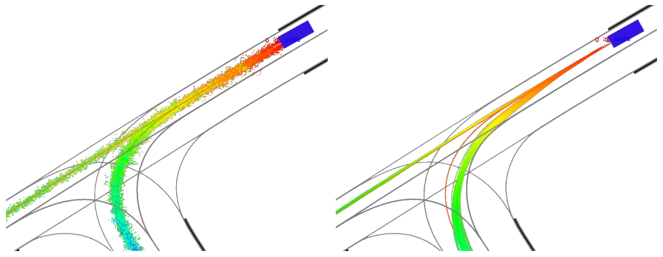


Fig. 6. Comparison of different network architectures using $\mathcal{D}^{\text{sim}}$: Losses over training iterations (validation solid, training dashed) depicted in the upper image and corresponding position prediction root mean square errors in the lower images with different zoom factors.

The validation and training losses of the different models are depicted in the upper part of Fig. 6. It can be seen that the linear model trains the fastest and achieves the lowest overall loss ($l_{\text{NLL}} = -4.02$, see also Tab. III). The lower part of Fig. 6 shows the corresponding root mean square position errors RMSE_{xy} depending on the prediction horizon. For short term predictions of up to around 2 s, all of the learned models outperform the rule-based model. Low one-step prediction errors are especially important for inference in a DBN. For long-term predictions, the linear model achieves the best results amongst the neural networks. However, the rule-based model becomes more accurate for predictions with very long horizons exceeding 15 s. This can be explained by the accumulating prediction error, for which the learned models cannot counteract. If this error is too high, the features determined during forward simulation are not represented within the training data anymore. This problem is commonly known in machine learning based trajectory prediction and could be reduced with techniques such as *data as demonstrator* [20]. The rule-based system, on the contrary, is designed to be attracted by the center of the lane, such that a vehicle will always stay close to the centerline. The runtime τ of any of the learned models (averaged over a batch of 1000 samples) is less than a third of the one of the rule-based model (see Tab. III). This is foremost the case due to efficient batch processing, allowing for fast execution when multiple samples can be processed independently (such as in particle filtering).

A qualitative result on the real data $\mathcal{D}^{\text{real}}$ can be seen in Fig. 7, comparing the forward simulation of the rule-based model to the learned linear model. It can be seen that the learned model is able to pick up subtleties such as cutting curves and curvature dependent lateral uncertainty. The linear neural network model also outperforms the rule-based model, as shown in Fig. 8. However, this dataset is quite small and does not contain as diverse situations as the simulated one.



(a) rule-based (b) linear network

Fig. 7. Qualitative comparison of rule based and learned linear model on real driving data. The learned model is able to reproduce subtleties such as cutting curves and different lateral uncertainties for curved and straight roads.

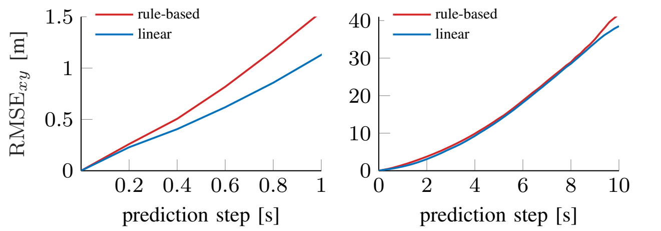


Fig. 8. Position prediction root mean square errors on real dataset with different zoom factors.

2) Feature Importance: To determine the importance of the single features, we train the linear network on $\mathcal{D}^{\text{sim}}$ starting with using only a single feature and iteratively adding features which result in the highest reduction of validation loss. Thus, redundant features can be pruned and the complexity of the model reduced. The five most important features and the corresponding validation losses by adding each of these features are:

	v^i	ϕ_{15}	$d_{\text{inters.}}$	ϕ_7	c_{70}	all
l_{NLL}	-0.714	-2.093	-2.779	-3.249	-3.346	-4.016

It can be seen, that even with as few as five features, it is possible for the network to learn basic behavior models that achieve lower losses than both recurrent networks.

3) Route Intention Estimation: By conditioning the action on a driver's route intention, one can compare the different route hypotheses with the actual observations and thus estimate their probabilities. This is achieved by integrating the learned model into the dynamic Bayesian network from [7]. A qualitative comparison of route estimation and the corresponding Kullback-Leibler divergence D_{KL} to the ground truth is shown in Fig. 10. The learned model achieves a lower D_{KL} is able to tell the correct route faster than the rule-based model.

TABLE III
RUNTIMES τ [s], LOSSES AND POSITION PREDICTION RMSE ϵ [m] FOR DIFFERENT NETWORK ARCHITECTURES USING DATASET $\mathcal{D}^{\text{sim}}$

Method	τ	l_{NLL}	$\epsilon_{0.2\,\text{s}}$	$\epsilon_{1\,\text{s}}$	$\epsilon_{5\,\text{s}}$	$\epsilon_{10\,\text{s}}$	$\epsilon_{20\,\text{s}}$
Rule-Based	2.2e-6	–	0.093	0.469	3.959	11.28	**33.48**
Linear	7.1e-7	**-4.02**	0.028	0.243	**3.237**	**10.75**	37.95
LSTM	6.8e-7	-3.26	**0.027**	0.272	4.414	16.40	60.02
GRU	6.4e-7	-3.05	0.033	0.346	5.326	20.25	67.96
Lenz et al.	**6.3e-7**	-3.40	0.029	**0.235**	3.393	13.07	54.11

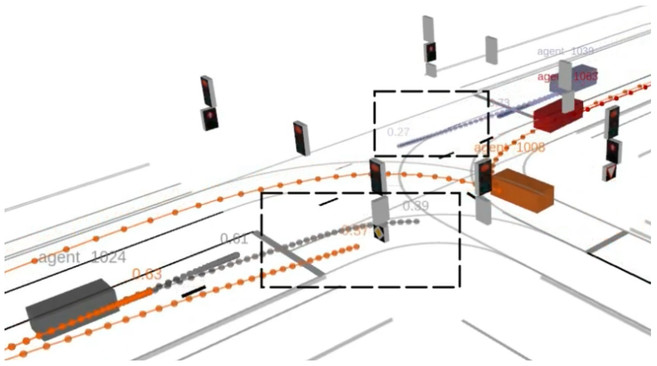


Fig. 9. A disadvantage of conditioning a machine learning model on a driver's route intention: routes that are so unlikely that they are not present in training data may result in unreasonable actions, such as red light violations.

An interesting disadvantage of conditioning the model on a driver's route intention can be seen in Fig. 9: When enumerating all possible routes and running a forward simulation for each of the conditioned models, there might exist route candidates that are so unlikely that they have never been followed in the training data. Thus their features may result in unreasonable actions during inference, as the network only learns what actions are reasonable given a route, but not which routes are reasonable given a situation. As an improvement, one could additionally learn the route priors and prune very unlikely hypotheses before forward simulation.

VI. CONCLUSIONS

This paper proposes to learn Markovian action models of human drivers from urban scenarios with deep neural networks. These models represent a probabilistic mapping from a feature-based representation of a traffic scene from the point of view of a driver to a distribution over his/her acceleration and steering angle. Conditioning this model on the driver's route intention, which is the main cause of multimodality in predictions, reduces learning complexity and allows for arbitrary road layouts with varying number of route hypotheses. The presented networks are able to predict yielding to other traffic, stopping at traffic lights and even account for subtleties such as cutting curves.

Learning the variance in a driver's actions allows to have a context-dependent magnitude of uncertainty in the transition model, which better captures the reality of human driving. Due to the simplicity of the models (2-4 layers), they come with low runtimes, enabling their application to sampling based frameworks such as Monte Carlo tree search or particle filtering. We showed that the learned models can easily be integrated into a DBN for route intention estimation. All presented architectures outperformed the rule-based model during forward simulation for short prediction horizons. However, the networks do not counteract for the accumulating error during forward simulation and thus have difficulties with longer horizons.

Future work should focus on improving long-term prediction performance: this could be achieved using techniques such as *data as demonstrator* [20], allowing the models

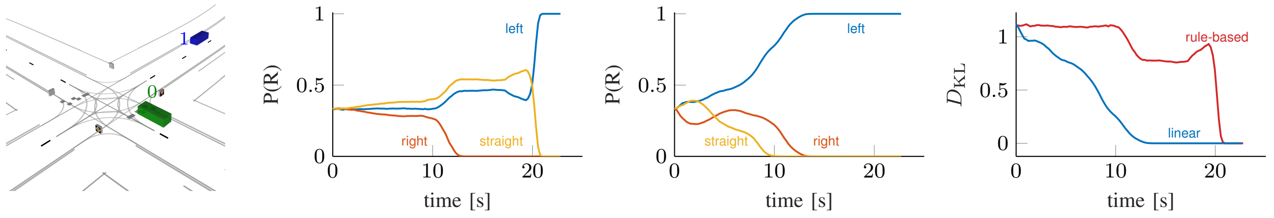


Fig. 10. Qualitative comparison of route estimation of green agent 0 at an urban scenario, wanting to turn left, using a DBN with rule-based (first plot) and learned linear neural network (second plot) action models and corresponding Kullback-Leibler divergence D_{KL} to ground truth (third plot).

to learn to approach the ground truth trajectories again after slight deviation. Furthermore, a more detailed and quantitative analysis of real driving data should be conducted to determine the applicability to even more diverse scenarios. Integration of the learned models in forward simulation based interaction-aware planning algorithms (such as [5], [6]) will yield insights on the benefits of more accurate behavior models for the driving behavior of autonomous vehicles.

References

[1] M. Treiber, A. Hennecke, and D. Helbing, "Congested Traffic States in Empirical observations and Microscopic Simulations," *Physical review E*, vol. 62, no. 2, 2000.

[2] D. J. Phillips, T. A. Wheeler, and M. J. Kochenderfer, "Generalizable Intention Prediction of Human Drivers at Intersections," in *Intelligent Vehicles Symposium (IV), 2017 IEEE*, pp. 1665–1670, IEEE, 2017.

[3] D. Lenz, F. Diehl, M. T. Le, and A. Knoll, "Deep neural networks for Markovian interactive scene prediction in highway scenarios," in *Intelligent Vehicles Symposium (IV)*, pp. 685–692, IEEE, 2017.

[4] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. S. Torr, and M. Chandraker, "DESIRE: Distant Future Prediction in Dynamic Scenes with Interacting Agents," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2165–2174, IEEE, July 2017.

[5] D. Lenz, T. Kessler, and A. Knoll, "Tactical cooperative planning for autonomous highway driving using Monte-Carlo Tree Search," in *Intelligent Vehicles Symposium (IV)*, pp. 447–453, IEEE, June 2016.

[6] 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, pp. 5–17, Mar. 2018.

[7] J. Schulz, C. Hubmann, J. Löchner, and D. Burschka, "Interaction-aware probabilistic behavior prediction in urban environments," in *Int. Conf. on Intell. Robots and Syst. (IROS)*, IEEE, 2018.

[8] D. Krajzewicz, "Traffic Simulation with SUMO – Simulation of Urban Mobility," in *Fundamentals of Traffic Simulation* (J. Barceló, ed.), pp. 269–293, New York, NY: Springer New York, 2010.

[9] S. Krauß, P. Wagner, and C. Gawron, "Metastable states in a microscopic model of traffic flow," *Physical Review E*, vol. 55, no. 5, p. 5597, 1997.

[10] M. Liebner, M. Baumann, F. Klanner, and C. Stiller, "Driver intent inference at urban intersections using the intelligent driver model," in *Intelligent Vehicles Symposium (IV)*, pp. 1162–1167, IEEE, 2012.

[11] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social LSTM: Human Trajectory Prediction in Crowded Spaces," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 961–971, IEEE, June 2016.

[12] F. Kuhnt, J. Schulz, T. Schamm, and J. M. Zöllner, "Understanding interactions between traffic participants based on learned behaviors," in *Intelligent Vehicles Symposium (IV)*, pp. 1271–1278, IEEE, 2016.

[13] T. A. Wheeler, P. Robbel, and M. J. Kochenderfer, "Analysis of microscopic behavior models for probabilistic modeling of driver behavior," in *International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1604–1609, IEEE, 2016.

[14] T. Gindele, S. Brechtel, and R. Dillmann, "Learning context sensitive behavior models from observations for predicting traffic situations," in *International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pp. 1764–1771, IEEE, 2013.

[15] J. Kong, M. Pfeiffer, G. Schildbach, and F. Borrelli, "Kinematic and dynamic vehicle models for autonomous driving control design," in *2015 IEEE Intelligent Vehicles Symposium (IV)*, (Seoul, South Korea), pp. 1094–1099, IEEE, June 2015.

[16] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[17] S. Steyer, G. Tanzmeister, and D. Wollherr, "Object tracking based on evidential dynamic occupancy grids in urban environments," in *2017 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1064–1070, June 2017.

[18] VIRES Simulationstechnologie GmbH, "OpenDRIVE." http://opendrive.org.

[19] A. Hochstädter, P. Zahn, and K. Breuer, "A comprehensive driver model with application to traffic simulation and driving simulators," in *Proc. Human-Centered Transportation Simulation Conf. HCTSC, Iowa City*, 2001.

[20] A. Venkatraman, B. Boots, M. Hebert, and J. A. Bagnell, "Data as demonstrator with applications to system identification," in *ALR Workshop, NIPS*, 2014.