Digital Behavioral Twins for Safe Connected Cars

Ximing Chen*
Department of Electrical and Systems Engineering
University of Pennsylvania
Philadelphia, Pennsylvania, USA
ximingch@seas.upenn.edu

Eunsuk Kang
School of Computer Science
Carnegie Mellon University
Pittsburgh, Pennsylvania, USA
eskang@cmu.edu

Shinichi Shiraishi
Toyota InfoTechonology Center Co., Ltd.
Tokyo, Japan
sshiraishi@jp.toyota-itc.com

Victor M. Preciado
Department of Electrical and Systems Engineering
University of Pennsylvania
Philadelphia, Pennsylvania, USA
preciado@seas.upenn.edu

Zhihao Jiang
School of Information Science and Technology
ShanghaiTech University
Shanghai, China
jiangzhh@shanghaitech.edu.cn

ABSTRACT

Driving is a social activity which involves endless interactions with other agents on the road. Failing to locate these agents and predict their possible future actions may result in serious safety hazards. Traditionally, the responsibility for avoiding these safety hazards is solely on the drivers. With improved sensor quantity and quality, modern ADAS systems are able to accurately perceive the location and speed of other nearby vehicles and warn the driver about potential safety hazards. However, accurately predicting the behavior of a driver remains a challenging problem. In this paper, we propose a framework in which behavioral models of drivers (Digital Behavioral Twins) are shared among connected cars to predict potential future actions of neighboring vehicles, therefore improving the safety of driving. We provide mathematical formulations of models of driver behavior and the environment, and discuss challenging problems during model construction and risk analysis. We also demonstrate that our digital twins framework can accurately predict driver behaviors and effectively prevent collisions using a case study in a virtual driving simulation environment.

CCS CONCEPTS

• **Computing methodologies** → **Modeling methodologies**; *Markov decision processes*; *Simulation evaluation*; Classification and regression trees;

KEYWORDS

Connected cars, Modeling methodologies, Markov decision processes, Classification

ACM Reference Format:
Ximing Chen, Eunsuk Kang, Shinichi Shiraishi, Victor M. Preciado, and Zhihao Jiang. 2018. Digital Behavioral Twins for Safe Connected Cars. In *ACM/IEEE 21th International Conference on Model Driven Engineering Languages and Systems (MODELS '18), October 14–19, 2018, Copenhagen, Denmark.* https://doi.org/10.1145/3239372.3239401

1 INTRODUCTION

Driving is a social activity that involves endless interactions with other agents on the road. Drivers are required to constantly gather information about their surroundings in order to make safe driving decisions. However, human drivers are also subjected to limited observations and distractions. Failing to know **where** these agents are and predict **what** these agents will do may result in serious safety hazards. Traditionally, the responsibility for avoiding these safety hazards is solely on the drivers. With improved sensor quantity and quality, modern ADAS systems are able to accurately perceive the location and speed of other vehicles nearby and warn the driver about potential safety hazards. However, accurately predicting the future behaviors of other agents in real time remains an unsolved problem. There are two key challenges associated with this problem:

1.1 Knowledge About Other Agents on the Road

In most of the driving scenarios, a driver does not have prior knowledge about other the agents on the road. Therefore, a driver has to make assumptions about these agents on the road to predict how they behave in certain driving scenario. There are two typical assumptions a driver can make with no prior knowledge:

- **Pessimistic Assumption:** In the worst case, a driver may assume that other agents can take any available actions with equal probability. The problem with this assumption is that there will be many false-positives. i.e. the ego vehicle may have no safe actions to choose from.

*This work was done when Ximing Chen, Eunsuk Kang, Shinichi Shiraishi and Zhihao Jiang were at Toyota InfoTechnology Center, Mountain View, California, USA. Emails: {xchen, ekang, sshiraishi, zjiang}@us.toyota-itc.com.

MODELS '18, October 14–19, 2018, Copenhagen, Denmark

ACM ISBN 978-1-4503-4949-9/18/10…$15.00
https://doi.org/10.1145/3239372.3239401

- **Optimistic Assumption:** A driver may assume that other agents will avoid actions that may lead to unsafe scenarios. (i.e. when a driver decides to overtake another car in another lane, he/she may assume that the other car will not suddenly change lane and smash into his/her car) The problem with this assumption is that there will be many false-negatives. i.e. an agent may not know the existence of the ego vehicle thus violating this assumption. Although the optimistic assumption is biased, it can still serve as a reasonable starting point for prior knowledge.

The truth lies somewhere in between and none of these two assumptions can produce accurate prediction of agent behaviors. However, if we have prior knowledge about an agent, we can have a more customized and accurate assumption about what the agent may do under different driving context.

1.2 Partial Observation of Driving Context

The behavior of a vehicle is determined by its driving context, which includes road conditions, agents nearby, infrastructures (e.g., traffic light), and even the mental state of the driver. With better sensors and connected technology, the capability of a vehicle to identify the driving context is improving. However, observation of driving context is always *partial*. Failure to infer other vehicles' driving context is one major cause for accidents. For instance, a car driving in front of you may brake to avoid an accident ahead, which may not be observable to you, leading to your collision with the car in front. Furthermore, the driving context perceived by the human driver may differ from the one perceived by the vehicle (e.g., through sensors), which makes it even more challenging to predict driver's behaviors.

1.3 Digital Behavioral Twin

In this paper, we focus on the challenge of providing accurate context information to drivers with partial, limited view of their surroundings. In particular, we propose a novel Digital Behavioral Twin framework, which leverages the idea of *model sharing* to improve the safety of connected cars. The overview of the framework is shown in Figure 1. With an increasing number of higher-quality sensors on board, modern vehicles have the capability to collect historical driving data. These data then are then used to construct a behavioral *profile* model of a driver for each vehicle, which can be used to predict his or her future behaviors under different driving contexts. Using the connected vehicle-to-vehicle (V2V) technology, these profiles are shared among a pair of neighboring vehicles and used to estimate the potential risks of a collision depending on the actions taken by the drivers. The risks for the available actions are then visualized to the drivers so that they can take safer actions to avoid a collision.

1.4 Related Work

Particularly, digital behavioral twin framework faces two challenges: (i) how to model the behavior of a driver and (ii) how to perform risk analysis. On one hand, the former question was discussed in [19], where the author identified three classes of task processes for driving: operational processes, tactical processes and strategic processes. Later on, seminal work such as [23, 27] proposed

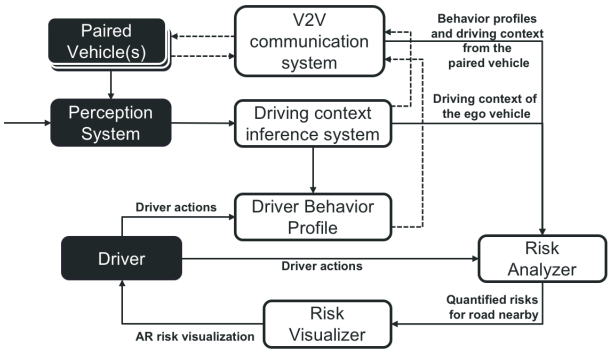


Figure 1: Overview of Digital Behavioral Twin

Markov Dynamical Models (MDM) to characterize the evolution of state of vehicles under different human status (e.g., relaxed or tight). Based on those proposed dynamical models, human behaviors over a few seconds time may be predicted. Similarly, authors in [18] modelled the driving process as selection of different control behavior governed by different internal cognitive states. Such an idea is further extended in [26] using ACT-R cognitive architecture – a general framework for specifying computational behavioral models of human cognitive performance. As an alternative, Hidden Markov Models (HMM) are also adopted for modelling driver's behavior [9, 15, 16, 29]. Subsequently, given observations on status of vehicles, e.g., speed and acceleration, one may infer the internal mental state, unobservable physical values of vehicles or the status of the driver [16]. To compensate the high computational complexity of HMM, these models are generally relaxed into switching linear dynamical models.

These pioneer work gave rise to context-aware systems, which utilize various-sources of information to infer the status (e.g., normal or intoxicated) of the driver and are able to provide a warning to other drivers on the road [3, 10]. For example, one can predict what the driver may do in the future by tracking driver's eye movement [13] or through foot gestures [33]. Instead of monitoring the physical condition of drivers, one can also infer driver status through speed, lateral position, steering wheel angle of vehicles [28].

While earlier work focuses more on modeling the intention or monitoring the status of drivers, researchers recently have also incorporated machine learning or data-driven techniques for predicting driver actions [12, 22, 34, 36]. For example, in [22], pedal operation patterns are examined by leveraging Gaussian Mixture Models whereas in [12], the authors used neural network to predict car driver's steering behavior from road curvature, velocity and acceleration of a car.

On the other hand, building a driver model enables risk analysis of certain driving behaviors. The risk analysis result can be leveraged to provide warning for drivers [7]. Nonetheless, how to formally define the notion of risks and calculate risks under different driving scenarios remain a challenging problem. Verifying whether a vehicle will crash into other vehicles can be abstracted into the problem of deciding whether the vehicle will enter unsafe states within a fixed horizon given initial conditions, which is also

referred to as reachability problems [30]. Nonetheless, due to modelling inaccuracy and uncertainties in behavior of other vehicles, the reachable sets are non-deterministic. To deal with these uncertainties, stochastic reachability has also been studied [4, 5]. In [6], the authors presented an efficient approach to evaluate the probability of a crash for specific driving trajectories of autonomous car by relaxing the system into grid-based Markov chains. In addition to reachability analysis, various solutions to finding collision probability have been proposed. For example, through modelling the movement of vehicles using Unscented Kalman Filter, the collision probability can be estimated using Monte Carlo simulation [35]. In [37] and [24], the authors proposed to generate possible vehicle paths and estimate collision probability using time-to-collision.

Our Contributions. It is shown in literature that various efficient solutions haven been proposed to cope with driver status modeling and inference as well as collision risk analysis. Nonetheless, few were proposed to leverage the advantage of both approaches and provide concrete framework for providing safety assessment for driver's actions. To address the above concerns, in this paper, we attempt to achieve the following objectives: (i) propose an application in which connected cars share their driver's behavioral models to better predict and prevent collisions; (ii) propose a mathematical formulation for driver's behavioral model and collision prediction; and (iii) prototype a case study in virtual platform and demonstrated the benefits of model sharing.

The rest of the paper is arranged as follows: Section 2 introduces a simple case study in which the Digital Behavioral Twin idea will be evaluated, as well as the virtual platform development for data generation and evaluation. Section 3 discusses the formulation and evaluation of driver's behavioral models. Section 4 introduces the formulation of driving context and collision prediction. Section 5 discusses the evaluation of the Digital Behavioral Twin idea in virtual environment. Section 6 summarizes the paper and discuss challenges and future work.

2 CASE STUDY: HIGHWAY DRIVING

The behaviors of a driver depend on many factors, which we refer to as *driving contexts*. We categorize driving contexts into two categories: (i) driver factors, and (ii) non-driver factors. The first category includes, but not limited to, the level of driving expertise, distractions (e.g., due to mobile phones) and the mental status of the driver (normal or intoxicated), whereas the second category includes the weather, the road quality and behaviors of other vehicles. In this study, we focus on the second category, which is easier to observe with the increasing quantity and quality of modern onboard sensors. The driving context is also limited to a simple "3 cars on two-lane straight highway" scenario, in order to keep the number of factors minimum. We will later see that our formulation can be easily extended to more complex driving scenarios.

The driving scenario is illustrated in Figure 2, where 3 cars are driving on a two-lane straight highway with arbitrary length, and each car can lead/follow or change lanes. By adjusting initial conditions, which include the gaps between vehicles, their speeds, and driving strategies (e.g., conditions in which overtaking or lane change occurs), we can cover a large number of driving scenarios one can encounter during highway driving.

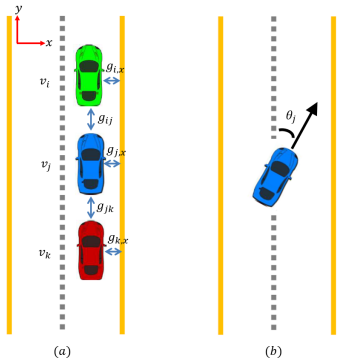


Figure 2: A driving scenario consisting of three cars driving on a two-lane highway. In (a), the green blue and red cars are indexed by i, j and k, respectively. The y-direction denotes the direction parallel to the highway and x-direction is perpendicular to the lanes. The speed of vehicles are represented by v_i, v_j and v_k. The distance to the right-most lanes are denoted by $g_{i,x}$ whereas g_{ij} denotes the distance between car i and j in y-direction. In (b), θ_j represents the angle of the car with respect to lanes.

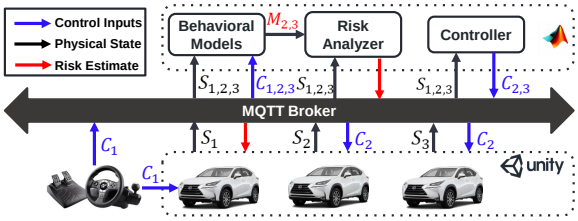


Figure 3: Virtual platform for data generation and framework evaluation.

In order to eliminate the effect of unrelated factors, we have developed a virtual platform to evaluate the performance of the Digital Behavioral Twin idea. The system overview for the virtual platform is shown in Figure 3. In this platform, the road structure and vehicle dynamics are implemented in the Unity game engine. The controllers that determine the vehicle behaviors and the risk analyzer are implemented in Matlab. Cars can also be driven manually and the parameters of the controllers can be adjusted to cover more behavior variations. A communication middleware called MQTT [1] is used to perform information exchange between Unity3D [2] and MATLAB. This virtual platform is used for data generation as well as evaluation of the Digital Behavioral Twin idea.

3 LEARNING DRIVER BEHAVIOR

In this section, we aim to design a framework for predicting driver's behavior given a certain driving scenario. To achieve this goal, we define the driver behavior of interest and propose a mathematical formulation that captures this behavior in terms of five different types of driver actions. Then, we propose using two interpretable classification algorithms to learn driver's behavior. Thus, based on the approaches, we are able to predict the probability of driver making a particular decision given driving scenarios.

3.1 Behavior Abstraction

To achieve the goal of predicting driver's behavior, as a first step, we characterize the set of possible behaviors of a driver. In one possible approach, one may consider the turning angle of the wheel, the degree of the gas paddle, and how hard the driver is pushing the brake as possible independent actions. Nonetheless, in practice, the driver may not tell the difference between two angles of the wheel (e.g., close-wise 90 degrees and close-wise 95 degrees). Instead, she is generally concerned more about which direction and how fast the vehicle goes. Thus, alternatively, we consider a high-level abstraction of the behavior of drivers. More specifically, we define a set of actions to model the *intention* of drivers as supposed to taking into account how to maneuver of the vehicle. For instance, given a fixed driving context, a driver may choose between following five actions:

slide left, slide right, accelerate, slow down, maintain speed. (1)

Therefore, the problem of learning behavior of the driver is recast as learning which action the driver may choose given in a particular driving context. It can be further formulated as a *classification problem* in machine learning, as we describe next.

3.2 Learning Abstracted Driver Behavior

Although there exist other formulations for learning driver's behavior, a classification-based approach achieves following advantages. First of all, it is *flexible*, in that it can be easily extended to include other contextual factors as features in the classification framework. For example, one may consider that the current action of a driver may depend on the driving context in the past few seconds, or the current weather condition. Moreover, it is possible to consider online learning in the context of learning driver behavior. For instance, if a driver is an amateur who does not have any driving records, we may learn his driving style while gathering data by utilizing online machine learning techniques.

Classification aims to identify which category a new observation belongs to by utilizing a set of training data containing observation and category membership pairs. In our case, the observations are driving contexts whereas the categories are the actions taken by drivers. Let k be the number of driving context components, $\mathbf{x} \in \mathbb{R}^k$ be a driving context and $A = \{1, \ldots, m\}$ be the set of labels for actions. We consider that we are given a set of training data $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ containing n context-action pairs, where $y_i \in A$ is the corresponding action in driving context $\mathbf{x}_i$. Given $\mathcal{D}$, we aim to obtain a function $f : \mathbb{R}^k \to A$ that predicts the action in A given a new driving context through classification algorithms.

For example, in the highway driving scenario from Section 2, we consider following features: (i) gaps, (ii) lane information, (iii) velocity difference between vehicles and (iv) angles of vehicles with respect to the lanes. Thus, to learn a particular vehicle e's behavior, each corresponding observation admits the following form:

$$\mathbf{x}_i = [g_{ej}, g_{ek}, g_{e,x}, g_{j,x}, g_{k,x}, v_{ej}, v_{ek}, \theta_e, \theta_j, \theta_k]^\top, \quad (2)$$

where subscript e refers to the ego vehicle, and j and k refer to the other two vehicles (e.g., g_{ej} is the gap between the ego vehicle e and another vehicle j). In addition, each label $y_i \in \{1, \ldots, 5\}$ corresponds to an action according to (1). Note that it is also possible to include the exact speed, denoted by v_e, v_j, v_k, into the feature vector in additional to the speed differences.

After defining the features as driving contexts and labels as actions, we propose two classification methods to learn drivers' behavior, which can provide accurate predictions as well as easy-to-interpret models.

3.2.1 Decision Tree. As discussed in [14], a decision tree is a well-known classification method that can be used to effectively model human decisions. Consider a driving scenario in which a driver intends to change lane. The driver may first check whether there are cars in front before checking the rear mirror. Subsequently, the driver decision process can be modelled as a tree in which each node represents one particular concern on the road, e.g., how far is the vehicle in front. Conversely, after obtaining a decision tree through training data $\mathcal{D}$, it is possible to interpret which factor a driver concerns more while driving.

3.2.2 K-Nearest Neighbor. In addition to decision trees, we leverage another well-known classification algorithm called k-nearest neighbor (KNN) [11], since a driver tends to behave similarly when encountering similar situations. For example, a conservative driver may not try to cut lanes whenever the traffic is crowded. Thus, the similarity measurement in different driving contexts can be handled by different metrics used in KNN-classification algorithms.

The details of how our approach uses decision trees and KNN, as well as the evaluation of their performance, are described in Section V. In the next section, we propose a framework which utilizes the predicted driver actions to evaluate the risk of collision in the context of connected vehicles.

4 DRIVING CONTEXT MODELS AND RISK ANALYSIS

In this section, we aim to perform risk analysis through sharing the predicted actions and contexts of a driver among connected vehicles. To achieve this goal, we first propose a centralized model using the union of all driving contexts, as a Markov Decision Process in Section 4.1.1. Nonetheless, such method exhibits scalability issues. To address this issue, we further propose an alternative local model using only information available to a pair of vehicles in Section 4.1.2. Finally, we perform risk analysis based on the proposed models.

We define risk of a certain action given a particular driving context as the probability of the vehicle entering an unsafe driving context if that action is performed at the current time instance. In order to perform probability estimation rigorously, we need to create models that can capture the dynamics of vehicles and the decision process of drivers. We picked discrete-time model over continuous-time model for the following reasons: 1) **Complexity:** Our goal is to perform real-time collision prediction. Estimating the collision probability between vehicles using a continuous-time model is a version of forward reachable set calculation in hybrid systems [20], which can be computationally challenging [21]. 2) **Model identification:** As the number of states is finite, identifying model parameters is easier for discrete-time models. 3) **Flexibility:** It is flexible to model different driving scenarios. For example, we may use a coordinate in the state to encode how many vehicles are around. 4) **Intuitiveness:** A large number of driving contexts are

equivalent from driver's point of view. i.e. The driver care more about whether another car is faster than him/her, than the exact speed difference. As a result, we may use a finite number of states to represent the set of possible driving contexts. Moreover, as a detailed model of certain aspects of vehicle dynamics cannot be easily specified, we regard the vehicle dynamics as a black box. Subsequently, only transition probabilities among states are available. Finally, a driver may only consider high-level actions, e.g., turn left and turn right, instead of considering the angle of the wheel and how hard the brake should be pushed, as discussed in the previous section. Thus, it is possible to use a fixed finite set of actions to represent drivers' choices. Based on the above heuristics, we propose to formulate the vehicle model as a Markov Decision Process in which the states represent driver contexts while actions represent driver behavior.

4.1 Driving Context Models using MDP

Formally, a *discrete-time Markov Decision Process* (MDP) M is a three-tuple (S, A, P), where S is a countable set of *states*, A is a finite set of *actions*, $P : S \times A \times S \to [0, 1]$ is a transition probability matrix such that $\sum_{s' \in S} P(s, a, s') \in \{0, 1\}$ for all states $s \in S$ and actions $a \in A$. Different from commonly used MDP, we ignore the reward function and discount factor in M. Hereafter, we provide two MDP formulations for driving context models.

4.1.1 Combined Context Model. We use the case study introduced in Section II as an example to demonstrate the formulation of the context model. We start by defining the states in S. In general, the decision made by a driver is dependent on the current driving context as mentioned in Section II. However, not all the factors play an equal role in affecting the driver's decision. For instance, distractions due to phones or text messages can be considered as rare events compared to the accumulated length of one's driving time. In addition, drivers may be concerned more about other vehicles on the road than the current weather. As a result, following this intuition we focus on the category of contextual factors that capture the behavior as well as status of other nearby vehicles.

To identify the factors within this category, we note that the following elements play an important role in affecting the driver's decisions: (i) the distances between vehicles, (ii) the distances to the rightmost lane, (iii) the relative speed of each vehicle and (iv) the angle of the vehicle with respect to the lanes.

The component of state s in S are defined as follows:

- *Relative gap*: We define relative gap between vehicle i and j as the position differences between vehicles in y-direction. In particular, it is measured between the center of vehicles. We quantize the value g_{ij} into k_y values by partitioning the real-line $(-\infty, \infty)$, where $g_{ij} > 0$ indicates that vehicle i is in front of vehicle j. We proceed the same quantization for g_{ik} and g_{jk}. The quantized relative gap are denoted by $\hat{g}_{ij}$.
- *Distance to the rightmost lane*: In general, the vehicles are driven within the lanes, except for the case when its driver tries to change the lane. Thus, it suffices to use three values (i.e., $\{0, 1, 2\}$) to indicate their relative positions with respect to the rightmost lane (i.e., {At right lane, In the middle, At left lane}). In other words, $g_{i,x}$ (resp. $g_{j,x}$ and $g_{k,x}$) are quantized into three values, denoted by $\hat{g}_{i,x}$ (resp. $\hat{g}_{j,x}$ and $\hat{g}_{k,x}$).
- *Speed of vehicles*: Similar to the approach for relative gap, we quantize the speed of vehicles into k_v values by partitioning it into a range of discrete numbers. The quantized speed is denoted by $\hat{v}_i$ for vehicle i.
- *Angle of vehicles*: We define the angle of vehicle as the angle between the direction at which the vehicle is going and the lane, as illustrated in Figure 2-(b). Furthermore, we assume that the vehicles are always moving forward; hence, its angle is quantized into k_θ discrete values by partition the range $[-\pi, \pi]$. Analogously, the quantized angle is denoted by $\hat{\theta}_i$ for vehicle i. In particular, negative angle indicates that the vehicle is heading towards north-west direction whereas positive angle indicates that the vehicle is heading towards north-east.

Using the definition above, each state $s \in S$ is represented by

$$s = [\hat{g}_{ij}, \hat{g}_{ik}, \hat{g}_{jk}, \hat{g}_{i,x}, \hat{g}_{j,x}, \hat{g}_{k,x}, \hat{v}_i, \hat{v}_j, \hat{v}_k, \hat{\theta}_i, \hat{\theta}_j, \hat{\theta}_k]^\top. \quad (3)$$

Therefore, according to this definition, the dimension of the state space is equal to $|S| = k_y^3 \times 3^3 \times k_v^3 \times k_\theta^3$, where the number 3 denotes the number of vehicles in the network. Notice that such a formulation can be generalized to cope with more vehicles by enlarging the dimension of the states.

To formulate the action space, we consider the actions listed in (1). Since each of the vehicles can choose 5 actions, the action space A consists of $|A|^3 = 125$ actions, where each one triplet of action represents the action taken by the three vehicles. With the above definitions, the size of the transition probability matrix equals to $|S| \times |S| \times 125$. Intuitively, the finer quantization we adopt leads to a better representation of the underlying system. In addition, a majority of the transitions are governed by vehicle dynamics. Nonetheless, finer abstractions introduce scalability issues. For instance, suppose that we adopt $k_y = 5$, $k_v = 3$ and $k_\theta = 3$ as parameters for quantization; then, $|S| = 2460375$. In this case, the number of entries in the transition probability matrix exceeds 3 quadrillion. Thus, the matrix will be learned inaccurately when the data is sparse while the state space is enormous. In general, how to select an appropriate quantization level is a challenging problem that requires insights into the problem domain. In Section 4.1.3, we will provide intuitions on quantization level selection schemes that are suitable for collision risk prediction.

4.1.2 Local Context Model. Previously, we have introduced a model containing the contexts of all vehicles on the road. Nonetheless, this approach exhibits scalability issues even when a very coarse quantization scheme is used let alone considering scenarios consisting of numerous vehicles. Fortunately, when calculating the collision probability given an action and a driver context, it is possible to consider **pair-wise** relationship between vehicles. Therefore, we propose a pair-wise model as an alternative approach. Let $M_{ej} = (S_{ej}, A, P_{ej})$ be the MDP model between vehicle e and vehicle j. Its state space and action space are defined as follows:

- *Relative gap*: Similar to the formulation in Section 4.1.1, we select the relative gap between vehicles as one of the components in a driving context. Nonetheless, in this model, we only consider g_{ej}, i.e., the signed distance between vehicle e and vehicle j in y-direction.

- *Distance to the rightmost lane and angle of ego vehicle*: We adopt the same quantization for the lane information of vehicles as well as the angle of the ego vehicle. However, different from previous formulation, we only consider the lane and angle information of vehicle e and j.
- *Speed difference of vehicles*: We notice that human drivers may not be able to distinguish between vehicle driving 17 miles per hour or 20 miles per hour. In fact, the driver is more concerned with the *relative speed*; i.e., whether a car nearby is faster or slower than her, rather than its exact speed. Based on this intuition, we adopt two kinds of quantization on the speed differences between vehicle e and j. First, we classify vehicle j as being *faster, slower* or at *roughly same speed*. More specifically, we compare the speed difference v_{ej} with a pre-defined threshold value t_v. If $|v_{ej}| > t_v$ (resp. $v_{ej} < -t_v$), then we say that the vehicle j is faster (resp. slower) than vehicle e. Similarly, we compare v_{ej} with two pre-defined threshold values to define whether j is *much slower* or *much faster*. We denote this quantized parameter by $\hat{v}_{ej}$.

In this formulation, state $s_{ej} \in S_{ej}$ is represented by the following 6-dimensional vector:

$$s_{ej} = [\hat{g}_{ej}, \hat{g}_{e,x}, \hat{g}_{j,x}, \hat{v}_{ej}, \theta_e, \theta_j]^\top. \tag{4}$$

The size of the state space is equal to $|S_{eg}| \leq k_y \times 3^2 \times 5 \times k_\theta^2$. The action space is defined by the actions of vehicle e and j. In other words, for every $a \in A_{ej}$, we have

$$a = [a_e, a_j]^\top, \tag{5}$$

where a_e and a_j are selected from the set of the actions listed in (1). As a result, there are 25 actions in the action space in total.

Remark 1. *The pair-wise model M_{ej} is a state abstraction of the global model M, see [17] for a more concrete definition of state abstraction. More specifically, one can define a function $\phi : S \to S_{ej}$ such that for all $s \in S$, there exists $s_{ej} \in S_{ej}$, where $\phi(s) = s_{ej}$ holds.*

4.1.3 State Quantization for MDPs. By utilizing this formulation, we are able to shrink the size of the state space from billions to thousands. This allows us to consider a finer quantization on the relative gap g_{ej} and the angle of the vehicles than was previously possible. The finer quantization, in turn, leads to a more accurate modelling of the dynamics and interactions between vehicles. However, depending on the number of states in M_{ej}, the number of samples required to learn the model accurately may still be enormous. Conversely, a coarse quantization may neither correctly characterize the driving scenarios nor ensure accurate collision probability calculation. When we adopt a coarse quantization on the gap between vehicles, it is likely that no matter what action the drivers take, the next state remains the same as the current one. In this case, the transition probabilities are concentrated on the diagonal of the transition probability matrix. Subsequently, the remaining transition probabilities are small, which may introduce inaccuracy in the calculation of collision probabilities. As a result, it is crucial to consider an appropriate abstraction such that state space is of an acceptable size and the prediction of collision can be estimated accurately.

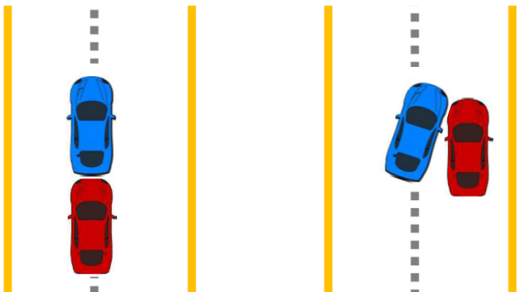
To address the aforementioned issue, we adopt a quantization scheme based on following intuition. In most of the driving scenarios, the driver may not care about vehicles that are far away. Thus, at the next time instance, no matter how well we quantize the gap, it does not affect the probability of collision. On the contrary, drivers are more cautious about nearby vehicles. As such, we only adopt a finer quantization of the gap when this value is small. In addition, we notice that the ultimate goal of the proposed work is to estimate the collision probability between vehicles. Thus, adopting a finer quantization on the gap and velocity is most crucial when vehicles are close in physical distances. For example, when the traffic is congested, vehicles are close to each other, and a sudden acceleration may lead to a higher collision probability than when the vehicles are far away.

Thus, instead of adopting a uniform quantization, we leverage a mixed approach in which a finer quantization is applied at the parameters with small values. For example, we may partition the positive real line as $(0, 6) \cup [6, 8) \cup [8, 12) \cup [12, 20) \cup [20, \infty)$., where quantization is finer when the gap between a pair of vehicles is smaller. By using this heuristic, we obtain a relatively accurate model while avoiding scalability issues. In the next section, we describe how we can utilize this model for risk analysis.

4.2 Risk Analysis

In this section, we consider using M_{ej} to calculate the probability of a collision between a pair of vehicles e and j. As a first step, we define the notion of *collision* under the pre-defined driving context. We notice that there are two possible circumstances at which a collision between vehicles may occur. In Figure 4-(a), we show the first case where a *rear-end collision* occurs, whereas in Figure 4-(b), we depict the scenario when a *non-rear-end collision* occurs (frequent in situations when a driver attempts to cut into lanes). In the first case, the gap between vehicles is less or equal to $\frac{l_e+l_j}{2}$, where l_e and l_j represent the length of vehicle e and j, respectively. Moreover, in this case, the two vehicles must be in the same lane or in the adjacent lanes. In the second case, the vehicles are in the adjacent lanes and their relative gap is less than or equal to $\frac{l_e+l_j}{2}$. In addition, the angle must be large enough. Although not explicitly shown in the figure, we remark that it is possible to generalize Figure 4-(a) to consider the case when head-on collision occurs, e.g., two vehicles heading towards each other in opposite directions. Similarly, we may generalize Figure 4-(b) to incorporate scenarios when one vehicle collides with another on the side, which happens frequently when drivers ignore traffic lights. As a consequence, it suffices to use a single collision state to represent all driving contexts corresponding to collision between vehicles. Let $c \in S_{ej}$ be the collision state. We are interested in calculating the probability of reaching this state given the current action and driving context.

During driving, the driver may not know the current action of the other driver. In some cases, the driver may not be able to observe certain information about neighboring vehicles (e.g., the angle or lane location). Under these circumstances, we denote the driver contexts in s_{ej} exclusive to ego vehicle e (resp. j) as s_e (resp. s_j). In other words, vehicle e only has access to a subset of components in the state s_{ej}. We assume that the union of information contained in s_e and s_j suffices to reconstruct the state s_{ej}.



(a) (b)

Figure 4: This figure demonstrates two possible scenarios when two vehicles collide.

In the context of connected vehicles, vehicle j is able to share s_j and a_j, where a_j can be obtained through learning algorithms in Section III, to vehicle e using a V2V communication network. As such, the collision probability can be calculated by:

$$\mathbb{P}(c \mid s_e, s_j, a_e, a_j) = P_{ej}(c \mid a, s_{ej}), \tag{6}$$

where $a = [a_e, a_j]^\top$.

On the other hand, when information is not shared, the only available information to vehicle e (resp. vehicle j) are a_e and s_e (resp. a_j and s_j). Thus, the vehicle has to estimate the collision probability given information exclusive to itself by:

$$\begin{aligned}\mathbb{P}(c \mid s_e, a_e) &= \sum_{s_j, a_j} \mathbb{P}(c \mid s_e, s_j, a_e, a_j)\mathbb{P}(s_j, a_j \mid a_e, s_e) \\ &= \sum_{s_j, a_j} P_{ej}(c \mid a, s_{ej})\mathbb{P}(s_j, a_j \mid a_e, s_e).\end{aligned} \tag{7}$$

Note that $\mathbb{P}(s_j, a_j \mid a_e, s_e)$ is the probability calculated using the probability distribution defined in M_{ej}. From Equation (7), we observe that in order to estimate the collision probability without information sharing, the ego vehicle has to infer the state and action of vehicle j. However, in practice, humans may perceive the state and actions of other vehicles incorrectly, which may result in a wrong estimation on the risk of his or her own driving behavior. With model sharing, uncertainties in estimation of the other vehicle's action can be eliminated, as described in Equation (6).

We further consider estimating the collision probability at h steps after. More specifically,

$$\begin{aligned}&\mathbb{P}(s^{t+h} = c \mid s^t, a^t) = \\ &\sum_{\substack{a^{t+1}, \ldots, a^{t+h-1} \\ s^{t+1}, \ldots, s^{t+h-1}}} P(s^{t+h} = c \mid s^{t+h-1}, a^{t+h-1}) \times \cdots \\ &\times P(s^{t+1} \mid s^t, a^t),\end{aligned} \tag{8}$$

where $a^t \in A_{ej}$ and $s^t \in S_{ej}$ denote the action and state at time step t, respectively. When information is not shared, a similar calculation can be used. As a result, it admits a similar form to Equation (8), except an additional term as described in Equation (7).

Remark 2. *In addition to estimating the risk of performing particular actions, if a vehicle observes a subset of components of states, it may utilize Bayesian rules to infer the next states, as discussed in [31] and [25]. In turn, we may invoke learning algorithms described in Section III to predict the corresponding actions.*

Consequently, using our pair-wise model and risk analysis framework, we may estimate the probability of vehicle e colliding with other vehicles.

4.3 Use of Heuristics and Prior Knowledge

Although we have proposed two different models in driving contexts, there are several remaining issues. Hereafter, we list a few challenges we faced during the implementations of the models and provide heuristics to tackle them.

4.3.1 Estimation of transition probability matrix. Although we can obtain the collision probability by using the pair-wise model M_{ej}, there is one caveat—the transition probability of the model is unknown. To address this issue, we propose the maximum likelihood estimation (MLE) to infer the entries of P_{ej} from a data set.

As shown in [8], the maximum likelihood estimator for an entry in P_{ej} equals to:

$$[P_{ej}]_{s,s',a} = \frac{n_{s,s',a}}{\sum_{\hat{s} \in S_{ej}} n_{s,\hat{s},a}}, \tag{9}$$

where $n_{s,\hat{s},a}$ is equal to the number of samples such that the transition from s to $\hat{s}$ occurs under action a. However, depending on k_y and k_θ, such an approach requires an enormous number of samples to train the model accurately [32]. One approach to tackle this problem is to introduce a prior on the distribution of transition probability matrices. For instance, we can consider the Dirichlet prior on each row of the transition matrix P_{ej}. Moreover, we can utilize first principles in physics to rule out unreachable states in our abstraction. In particular, we adopt the following rules to create a sparsity pattern for the transition probability matrix.

- *Rules to regulate lane transitions*: When the driver chooses to slide left (resp. slide right), the lane number of the next state will not be smaller (resp. larger) than that of current state, since we use 0 to denote the leftmost lane.
- *Rules to regulate the relative gap and speed transitions*: For example, when the driver in vehicle e is slowing down while the driver in vehicle j is maintaining speed or speeding up and if the vehicle e is in front of vehicle j, then the signed distance between vehicle g_{ej} in the next state will not be larger than that in the current state. In this case, the quantized speed differences $\hat{v}_{ej}$ will not increase. Analogously, we can characterize other rules for the cases when a driver intends to change the vehicle speed.

Using the above physics rule and domain knowledge, for each state and action, we can manually identify the set of unreachable next states.

4.3.2 Sampling period abstraction. As the notion of time in our proposed model is discrete, the time is sampled periodically. In other words, we set the difference between step $t+1$ and t in an MDP M equals to some predefined value t_s (in seconds). Intuitively, the time cannot be sampled either too sparsely or densely. On one hand, if we adopt 5 seconds as the sampling period, then the risk

analysis may be ineffective since a collision often occurs within the window of one or two seconds. On the other hand, if we set $t_s = 0.1s$, we may encounter computational issues. More specifically, as shown in multi-step collision probability calculation (8), an estimate of the probability at h steps ahead involves several matrix multiplication operations, whose complexity is $O(n^3)$, where n is the dimension of the matrix. Thus, to predict the collision probability at one second after, we need at least 10 matrix multiplication operations. Such operations incur a significant amount of latency and may be impractical, as the computational power on a typical vehicle CPU is limited. Taking into account the above concerns, we adopt $t_s = 1s$ in this paper. Note that when there is a collision within the sampling interval, the current state will transition to the particular collision state during the process of learning the entries of the transition probability matrix.

In the next section, we provide experimental results to evaluate the performance of our approach to the driver behavior prediction and risk analysis.

5 FRAMEWORK EVALUATION

In this section, we evaluated the performance of the framework discussed above using the virtual platform introduced in Section 2.

5.1 Evaluation of Behavior Prediction

We first evaluated whether we can construct a behavioral model for a driver; thence accurately predicted the driver's behavior in different driving contexts.

5.1.1 Data Generation. For evaluation purposes, we started with constructing a simple controller for vehicles. The controller acts as the "driver" of the cars. The more deterministic controller can provide repetitive results and automated evaluations. The same approach can be easily extended to human drivers in the future. The controller mimics how human drivers make driving decisions during highway driving. Based on the location and speed of another two cars, the controller can choose to follow/lead, overtake and merge. We may adjust parameters (e.g., gap between the vehicle and the front vehicle, overtaking speed) to represent different driving profiles. In what follows, we would like to learn the behavioral model of one of the car, which we refer to as Car 1. The simulation trails are created as follows: We ran 100 20-sec (1000 samples) simulations, with the 3 cars starting from different initial conditions, which can cover how Car 1 interact with other drivers in different driving contexts. In total, we generated $n = 1,000,000$ samples containing driving contexts of three cars and derived one million observations in the format of Equation (2).

5.1.2 Driver Model Learning. A priori to learning the behavior of Car 1, we labelled the samples (i.e., defining driver's actions for each observations) by incorporating the controller information. Following this step, we tried to train our behavior model using decision trees. However, it is unclear which particular criteria we adopt for splitting the features. In what follows, we compared the performance of two commonly used criteria for splitting: (i) cross entropy and (ii) Gini index. In Figure 5-(a) and (b), we showed the training and testing error of the decision tree when its number of splits are constrained. To counter the defect of over-fitting, we adopted 10-fold cross validation, a technique which randomly partitions the data set into 10 equally-sized sets and use 9 of them for training, one for testing. As the size of the tree grows, the training error reduces since the complexity of the model increases. However, the testing error also reduces significantly from more than 10% to 5% in both cases. It is also worth noticing that although different criteria are implemented, there is no significant differences in training and testing error.

To explore what features matter the most in this learning process, we constraint the number of splits of the decision tree to 5. Subsequently, we train our decision tree with one million samples using Gini index as criterion for splitting and with features defined in Equation (2). We depict the structure of the tree on Figure 6. When the angle of the vehicle is negative, indicating that it is heading towards left, it is likely that the driver is trying to slide left. Conversely, when θ_1 is larger than a threshold, the driver is most likely to slide right. Finally, when the speed of vehicle is smaller than 32.57 miles per hour, the driver may choose to speed up. However, in the figure, none of the leaves of the tree correspond to the slow down action. This is because as we constraint the number of splits, the resulting training samples within each of the particular splits contains more labels on other actions than the action slow down. We adopt the same approach to decision tree trained using cross entropy, whose structure is depicted in Figure 6-(a). Although different criteria results in different structure of the tree, they tend to select similar features.

We adopted similar set-up for learning using K-nearest neighbor. Intuitively, the drivers may behave similarly when they encounter similar driving context. Indeed, as demonstrated in Figure 5-(c), the nearest neighbor classifier achieves less than 1.5% cross-validated training error and 2.5% testing error. Furthermore, the training and testing error does not vary too much as the number of neighbors increases. Compared with decision trees, KNN classifier achieves lower testing error in our particular setup. However, when it comes to actual implementation, in additional to prediction accuracy, it is also crucial how much time it requires to obtain the predicted actions. We remark that although both of above classification algorithms achieves less than 5% testing error, it may be also due to the fact that the driving scenario is limited to highway driving. In this particular case, the control strategy is almost deterministic.

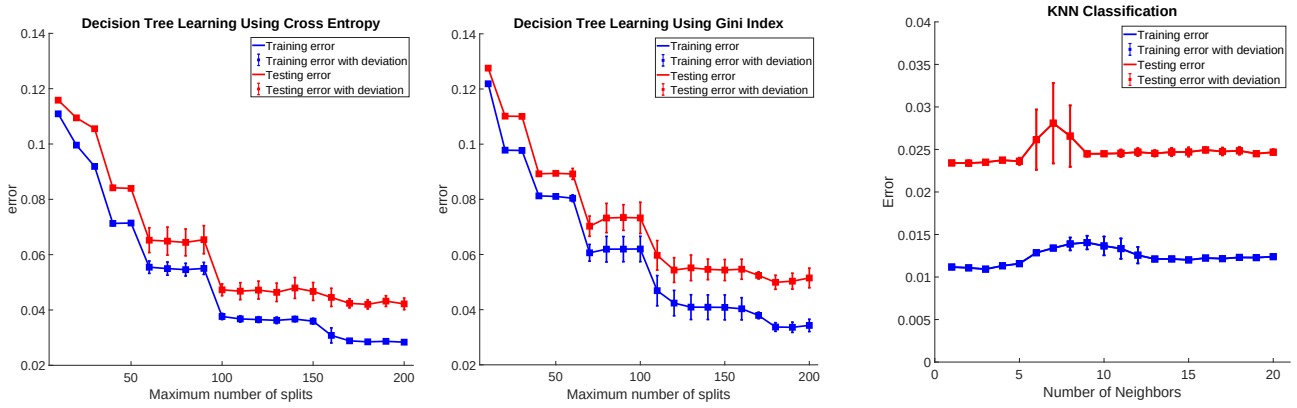
5.2 Evaluation of Risk Analysis

We evaluated the performance of the pairwise model when it is used to predict potential collisions.

5.2.1 Data Generation and Model Construction. In order to cover the interactions between two cars, we set the initial conditions of the two cars, while each car taking 1 out of 5 actions for 3 seconds. The initial conditions were sampled as follows:

- Speed difference between two cars: from -10 to 10 with step size equals to 4 in miles per hour (mph),
- Gap between two cars: from -10 to 10 with step size equals to 4 in meters,
- Lane for Car i: lane 1 or lane 2,
- Action for Car i: 1 to 5,

where i ranges from $\{1, 2\}$. There were, in total, 1008 simulation runs after removing invalid initial conditions (e.g., two cars colliding



(a) (b) (c)

Figure 5: In (a) and (b), we show the training and testing error versus the maximum number of splits in decision tree trained according to minimizing cross entropy and Gini index at each split, respectively. In (c), we show the training and testing error versus the number of neighbors considered in KNN classification algorithm.

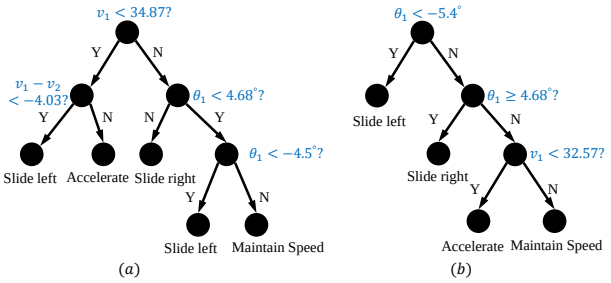


Figure 6: In (a) and (b), we show the structure of decision tree trained according to cross entropy and Gini index as splitting criteria, respectively. Notice that we measure the speed of vehicles at each sampled time-instances in the unit of miles per hour.

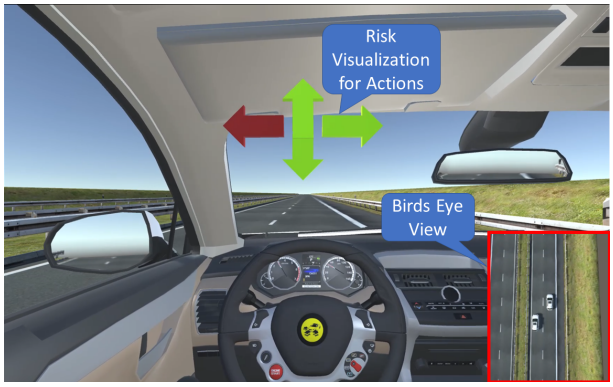


Figure 7: Risk visualization in virtual environment. The action with the largest probability to collision is marked red.

with each other initially). With these generated data, we constructed the MDP model whose size of state space equals to 1944. Due to limited amount of data, only 0.18% of the entries in the transition matrix are covered. Although the coverage seems low, most of the uncovered states are actually unreachable. In other words, by introducing physics rules described in the previous section, we mitigates the effect brought by data shortage.

5.2.2 Risk Analysis & Prevention. In order to evaluate the accuracy of our risk prediction, we ran another set of simulations with slightly different initial conditions, as testing data. The initial conditions were sampled as follows:

- Speed difference between two cars: from -15 to 15 with step size equals to 5 in mph,
- Gap between two cars: from 0 to 10 with step size 4 in meters,
- Lane for Car i: lane 1 or lane 2,
- Action for Car i: 1 to 5.

We observed that there were 332 collisions cases among the 784 simulations. Next, we evaluated the collision probability for each action, using the methods discussed in Section 4.2. The action with the highest collision probability is visualized to driver as shown in Fig. 7.

We then proposed a naive risk prevention algorithm to mimic how human driver may react to visual warnings:
If the action specified in the initial condition has the maximum collision probability among 5 available actions in a particular state, change the action to "Maintain Speed".
We then ran simulations with the same testing initial conditions and count the number of collisions. The intuition is that if the risk predictions are accurate and on time, the algorithm should be able to prevent most of the collisions. We observed that, out of 332 collisions without the prevention algorithm, only 83 collisions remain with the algorithm running. Furthermore, most of these remaining collisions are due to extreme initial conditions which cannot be prevented, e.g. cars are too close with large speed difference. We depict a case in Fig. 7, a car is in the blind spot of the mirror and our framework accurately predicted that changing lane to the left is dangerous, thus effectively prevented collision.

6 DISCUSSION AND FUTURE WORK

As the advance of technology allows vehicles to perceive the status of other vehicles nearby, it is of interest to develop a mechanism to provide warning and safe instructions for drivers to avoid potential hazards while driving. To achieve this goal, we have proposed a Digital Behavioral Twin framework, in which a vehicle is capable of learning the driver's behavior and predicting the collision risk by sharing information with other connected vehicles. More specifically, by properly defining driving contexts and driver behaviors, we have reformulated the problem of learning driver's behavior into classification problems and leveraged two methods (decision trees and KNN) for learning. Moreover, we have constructed a centralized model using a Markov Decision Process to model the evolution of driving contexts. To counter the defect of model complexity incurred from the centralized model, we adopted heuristics to relax it into a pairwise model—a model that depicts interactions between a pair of vehicles. Such a pairwise model allows us to tackle scalability challenges and compute the collision probability with and without information sharing. Finally, we have built a virtual platform using MATLAB, MQTT and Unity, in which we demonstrated that our framework can accurately predict driver behaviors and avoid potential collisions.

In the future, we may design multiple test cases, in addition to 3 cars on a highway, to evaluate our framework. Although we have formulated the connected vehicle model as a Markov Decision Process, the transitions probabilities rely on training data and may not correctly characterize the collision probability. Therefore, in the future, we plan to explore other methods such as continuous-time formulation using hybrid dynamical systems. Furthermore, in practice, the vehicle may have uncertainties in perception due to measurement errors in sensors; subsequently, it is also of interest to consider a model formulation that takes into account partial observations.

ACKNOWLEDGMENT

The authors gratefully acknowledge the contributions of Wenchao Li, Jiameng Fan, Kacper Wardega, and Weichao Zhou for helpful discussions on the digital behavioral twin project.

REFERENCES

[1] 1999. MQTT Protocol. http://mqtt.org.
[2] 2018. Unity. https://unity3d.com.
[3] S. Al-Sultan, A. H. Al-Bayatti, and H. Zedan. 2013. Context-aware driver behavior detection system in intelligent transportation systems. *IEEE transactions on vehicular technology* 62, 9 (2013), 4264–4275.
[4] M. Althoff, O. Stursberg, and M. Buss. 2007. Safety assessment of autonomous cars using verification techniques. In *Proceedings of American Control Conference*. IEEE, 4154–4159.
[5] M. Althoff, O. Stursberg, and M. Buss. 2008. Stochastic reachable sets of interacting traffic participants. In *Intelligent Vehicles Symposium*. IEEE, 1086–1092.
[6] M. Althoff, O. Stursberg, and M. Buss. 2009. Model-based probabilistic collision detection in autonomous driving. *IEEE Transactions on Intelligent Transportation Systems* 10, 2 (2009), 299–310.
[7] P. Angkititrakul, R. Terashima, and T. Wakita. 2011. On the use of stochastic driver behavior model in lane departure warning. *IEEE Transactions on intelligent transportation systems* 12, 1 (2011), 174–183.
[8] L. E. Baum and T. Petrie. 1966. Statistical inference for probabilistic functions of finite state Markov chains. *The annals of mathematical statistics* 37, 6 (1966), 1554–1563.
[9] H. Berndt, J. Emmert, and K. Dietmayer. 2008. Continuous driver intention recognition with hidden markov models. In *Proceedings of IEEE International Conference on Intelligent Transportation Systems*. IEEE, 1189–1194.
[10] R. Chhabra, S. Verma, and C. R. Krishna. 2017. A survey on driver behavior detection techniques for intelligent transportation systems. In *Proceedings of International Conference on Cloud Computing, Data Science & Engineering-Confluence*. IEEE, 36–41.
[11] T. Cover and P. Hart. 1967. Nearest neighbor pattern classification. *IEEE transactions on information theory* 13, 1 (1967), 21–27.
[12] A. Demcenko, M. Tamosiunaite, A. Vidugiriene, and A. Saudargiene. 2008. Vehicle's steering signal predictions using neural networks. In *Intelligent Vehicles Symposium*. IEEE, 1181–1186.
[13] M. S. Devi and P. R. Bajaj. 2008. Driver fatigue detection based on eye tracking. In *Proceedings of IEEE International Conference on Emerging Trends in Engineering and Technology*. IEEE, 649–652.
[14] G. James, D. Witten, T. Hastie, and R. Tibshirani. 2013. *An introduction to statistical learning*. Vol. 112. Springer.
[15] T. Kumagai and M. Akamatsu. 2006. Prediction of human driving behavior using dynamic Bayesian networks. *IEICE Transactions on Information and Systems* 89, 2 (2006), 857–860.
[16] T. Kumagai, Y. Sakaguchi, M. Okuwa, and M. Akamatsu. 2003. Prediction of driving behavior through probabilistic inference. In *Proceedings of International Conference on Engineering Applications of Neural Networks*. 117–123.
[17] L. Li, T. J. Walsh, and M. L. Littman. 2006. Towards a unified theory of state abstraction for MDPs. In *ISAIM*.
[18] A. Liu and D. Salvucci. 2001. Modeling and prediction of human driver behavior. In *International Conference on HCI*.
[19] J. A. Michon. 1985. A critical view of driver behavior models: what do we know, what should we do? In *Human behavior and traffic safety*. Springer, 485–524.
[20] I. M. Mitchell. 2007. Comparing forward and backward reachability as tools for safety analysis. In *International Workshop on Hybrid Systems: Computation and Control*. Springer, 428–443.
[21] I. M. Mitchell, A. M. Bayen, and C. J. Tomlin. 2005. A time-dependent Hamilton-Jacobi formulation of reachable sets for continuous dynamic games. *IEEE Transactions on automatic control* 50, 7 (2005), 947–957.
[22] C. Miyajima, Y. Nishiwaki, K. Ozawa, T. Wakita, K. Itou, K. Takeda, and F. Itakura. 2007. Driver modeling based on driving behavior and its evaluation in driver identification. *Proc. IEEE* 95, 2 (2007), 427–437.
[23] A. Pentland and A. Liu. 1999. Modeling and prediction of human behavior. *Neural computation* 11, 1 (1999), 229–242.
[24] M. Roth, F. Flohr, and D. M. Gavrila. 2016. Driver and pedestrian awareness-based collision risk analysis. In *Proceedings of IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 454–459.
[25] M. Rudemo. 1973. State estimation for partially observed Markov chains. *J. Math. Anal. Appl.* 44, 3 (1973), 581–611.
[26] D. D. Salvucci. 2006. Modeling driver behavior in a cognitive architecture. *Human factors* 48, 2 (2006), 362–380.
[27] D. D. Salvucci, E. Boer, and A. Liu. 2001. Toward an integrated model of driver behavior in cognitive architecture. *Transportation Research Record: Journal of the Transportation Research Board* 1779 (2001), 9–16.
[28] D. Sandberg and M. Wahde. 2008. Particle swarm optimization of feedforward neural networks for the detection of drowsy driving. In *Proceedings of IEEE International Joint Conference on Neural Networks and IEEE World Congress on Computational Intelligence*. IEEE, 788–793.
[29] A. Sathyanarayana, P. Boyraz, and J. Hansen. 2008. Driver behavior analysis and route recognition by hidden Markov models. In *Proceedings of IEEE International Conference on Vehicular Electronics and Safety*. IEEE, 276–281.
[30] C. Schmidt, F. Oechsle, and W. Branz. 2006. Research on trajectory planning in emergency situations with multiple objects. In *Proceedings of IEEE Conference on Intelligent Transportation Systems Conference*. IEEE, 988–992.
[31] R. L. Stratonovich. 1960. Conditional markov processes. *Theory of Probability & Its Applications* 5, 2 (1960), 156–178.
[32] I. Teodorescu. 2009. Maximum likelihood estimation for Markov Chains. *arXiv preprint arXiv:0905.4131* (2009).
[33] C. Tran, A. Doshi, and M. M. Trivedi. 2012. Modeling and prediction of driver behavior by foot gesture analysis. *Computer Vision and Image Understanding* 116, 3 (2012), 435–445.
[34] W. Wang, J. Xi, and H. Chen. 2014. Modeling and recognizing driver behavior based on driving data: A survey. *Mathematical Problems in Engineering* 2014 (2014).
[35] J. Ward, G. Agamennoni, S. Worrall, and E. Nebot. 2014. Vehicle collision probability calculation for general traffic scenarios under uncertainty. In *Proceedings of IEEE Intelligent Vehicles Symposium*. IEEE, 986–992.
[36] J. Xu, H. Shu, and Y. Shao. 2018. Modeling of Driver Behavior on Trajectory-Speed Decision Making in Minor Traffic Roadways With Complex Features. *IEEE Transactions on Intelligent Transportation Systems* (2018).
[37] R. Zeng, W. Sheng, and D. Yang. 2015. Collision probability computation based on vehicle to vehicle communication. In *Proceedings of IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems*. IEEE, 1462–1467.