Interpretable and Realtime Predictions of Social Interactions for Autonomous Vehicles

by

Cyrus Anderson

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Robotics)
in The University of Michigan
2021

Cyrus Anderson
andersct@umich.edu
ORCID iD: 0000-0003-1464-7676

To Hitomi

ACKNOWLEDGEMENTS

Were it not for all the people who supported me, I would not have made it this far. My advisors, Professor Matthew Johnson-Roberson and Professor Ram Vasudevan, supported me greatly through many discussions and encouragement that helped me get back on the road after getting stuck. They have also played a major role in shaping how I think about robotics and more broadly. I am immensely grateful for having had them as mentors. Thank you also to members of my doctoral committee, Professor Necmiye Ozay and Professor Lionel Robert for all of your guidance, support, and insights.

My labmates in the Ford Center for Autonomous Vehicles (FCAV), as well as in the Deep Robot Optical Perception (DROP) lab and the Robotics and Optimization for the Analysis of Human Motion (ROAHM) lab have been a great source of positive energy and insights. Mani worked with me when I first entered the lab and helped me to get my bearings. He and Ming-Yuan have readily listened to my wild ideas and been great support ever since. I am grateful to Matt and Alexa also for making the space in many mornings to talk about recent events or anything else, and always the insights on coffee. A large part of the experience would not be present without mentioning food. As much has progressed in lab, outside of lab my culinary skills have also undergone major change. Thank you to Carl, for helping provide that spark, and for the massive support in working with the various tools in lab. I also thank Jinsun and Junming, who also helped to make the lab potlucks a delicious success, and who have been a great pleasure to work with too. Xiaoxiao has a deadly pie recipe, has been an amazing collaborator, and a great mentor when I was still finding my way. Wonhui, for listening to my culinary escapades, and helping me to find my direction in research. Without help from Charlie and Trinh, much of this work would have taken considerably longer as well.

I also am thankful to be part of the Robotics community at the University of Michigan. It takes significant effort to cultivate such a supportive and vibrant environment, and I am glad to have been able to participate. Thank you to Damen, Denise, and Dan for all of your support in the program as well. This work also would not be possible without the support of the Ford Motor Company, via the Ford-UM

Alliance under awards N022884 and N028603.

Thank you to my friends and family, for all of the support over the years. And especially thank you to my loving partner, from whose example I am always learning, Hitomi.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Autonomous vehicles present an opportunity to transform transportation. The benefits range from increased access to mobility and time freed from driving, to greater safety due to automation. These robots are powered by the coordination of various systems to perceive and navigate through the world. Crucially, the autonomous vehicle operates in an open environment alongside fellow road users with whom it will interact regularly. Predictions of fellow road users' intents and future motion guide these interactions and specify a large part of the autonomous vehicle's behavior. Spurred by advances in deep learning, recent prediction methods have increasingly begun to consider how interactions affect future motion in ever more varied environments. The corresponding gains in accuracy translate to more anticipatory and less reactive autonomous vehicle behavior. One drawback is an increase in complexity, which can lead to less interpretable predictions and behavior. Achieving realtime performance and handling missing data caused by adverse sensing conditions present additional challenges.

To support autonomous vehicle behavior that is transparent and predictable, this thesis develops interpretable prediction methods. Model-based approaches provide the vehicle for building interpretable predictions, and novel inference procedures are developed to generate the predictions in realtime. Adopting a probabilistic framework enables natural handling of missing data and affords the flexibility to model interactions in varied environments beyond those described by existing interpretable methods. Experiments on real highway traffic and urban data demonstrate the developed methods' effectiveness.

CHAPTER I

Introduction

1.1 Autonomous Vehicles

Transportation forms a part of everyday life. Drivers in the U.S. alone are estimated to have driven over 3 trillion miles in total in 2018 [2]. Humans excel at navigating, and readily adapt to a wide variety of scenarios. These scenarios range from driving through construction zones, to the more mundane structure of signaled intersections, and the lack of structure in shared spaces. Negotiating with other road users for priority adds further complexity to the driving task. Autonomous vehicles (AVs) present the possibility of automating many of these drives. Immediate impacts include freeing the time that would otherwise be spent focusing on the road, and increased mobility for impaired persons [3]. The advent of AVs also brings benefits such as lower carbon emissions due to ridesharing [4], and fewer traffic accidents due to distracted driving and human error.

Autonomous vehicles are formed from a collection of systems working in unison. Data from cameras and other sensors are processed by the perception system to identify the locations of other road users in the world. Planning systems find a sequence of control inputs to safely drive the vehicle along a desired path. Between these lies the prediction system. Just as human drivers learn to watch their surroundings to avoid situations that would lead to a collision, predictions facilitate this capability in autonomous vehicles. The role of the prediction system is to use observations of road users to estimate where they may be in the future. This information can then be used in planning at a high level to decide whether to yield, or at a low level to constrain the feasible region for driving. The uncertainty inherent in road users' motion makes prediction over long time horizons a challenging task. Interaction adds further complexity by increasing the need to consider multiple road users' observations jointly to make predictions for even a single road user. For a common example of interaction between vehicles, a driver's actions may depend greatly on those of the vehicle they are driving behind. Beyond these points, it is important that the prediction system interface well with the perception and planning systems. The various needs that predictions aim to fulfill are described next.

1.2 Role of Predictions in Autonomous Driving

Predictions rely on various data streams produced by perception systems. It is rarely the case, however, that these are without errors or uncertainty. A common source of error stems from inaccuracy in road users' estimated positions. More severe, is the missing data caused by failures to detect road users, which may be caused by lighting conditions, or occlusions that hide road users from the perception system's sensors [5, 6, 7, 8]. Predictions based on environmental maps [9, 10, 11] may need to ensure that changes such as road construction do not render the maps unusable, or rely further on perception systems to construct these maps in realtime. Uncertainties in other perception systems such as pose estimation may also affect the predictions [12, 13]. Given the uncertainty inherent in perceiving the world, prediction systems should be equipped to handle the various kinds of errors that will occur while driving. A separate use of predictions is to directly account for the perception system's limited detection capabilities. Blind corners represent one such scenario that human drivers regularly face. Predicting how vehicles may appear from outside the detectable region can enable planners to account for these blind spots [14, 15, 16].

In contrast to perception, planning is downstream of predictions. Planning uses predictions to specify the risk due to other road users' future motion in routing the autonomous vehicle through less risky regions [14]. Another use is to specify regions of the road to avoid entirely [17, 18]. If these predicted regions fully contain the positions that road users could potentially occupy, then finding a route through the remaining portion of road will guarantee collision-free navigation. For this reason, planning may value conservative predictions that overestimate road users' possible occupancy. One way to accomplish this is to predict road users' occupancy with assumptions only on how their motion is physically constrained [19, 20, 21]. Too much conservatism, however, will prevent normal operation alongside other road users. This may come in the form of freezing in place [22] when conservative predictions eliminate all possible routes, or the sudden application of emergency maneuvers. Long-term predictions in particular are subject to greater uncertainty and will exacerbate this issue. On the other hand, conservatism is less useful in long-term predictions where the possibility of collision remains distant. In these settings, methods that focus on accuracy can be used to complement the more conservative methods [23]. This accuracy often comes at the cost of making greater assumptions about road user behavior. Rule-based behavioral assumptions, such as that pedestrians rarely enter the road outside of crosswalks, are one example [24]. Assuming that road user behavior follows a specified model, or one built from recorded data, is a widespread approach to achieving accurate predictions. This last approach will be the main topic of this thesis. Not all decisions made by autonomous vehicles will require predictions of road users' exact positions. For planning at a high level whether to yield, it may be sufficient to know that a pedestrian intends to cross the street. This information may be provided by specialized intent prediction methods [25, 26] or come as part of position based predictions [27, 28]. Whatever the form that predictions take, they will guide both planning and the autonomous vehicle's interaction with fellow road users.

1.3 Trajectory Prediction

Trajectory prediction methods use a road user's observed positions to predict future positions. Positions are typically 2D coordinates in the world viewed from a bird's eye perspective, to match the output of AV tracking systems [29]. The time between observations is often fixed which allows for the use of timesteps in place of times. An example from the DUT dataset [30] in Figure 1.1 shows three seconds of observed positions made at a rate of 10 Hz and the next five seconds of positions at the same rate. Besides the road user's observed positions, prediction methods may use additional sources of information provided by the AV's sensors. Information about the surrounding environment may be available in the form of high-definition maps or satellite images. This is especially useful for using knowledge of lane centerlines to predict vehicle trajectories, and identifying locations where pedestrians are likely to cross the street such as crosswalks. Contextual information is also provided by other road users' observed positions, which is valuable for prediction methods that model the interactions between road users.

Let x_t denote the road user's position at timestep t, and c denote the available contextual information. Assuming k observations are made before predicting future positions until the final timestep T, the prediction task amounts to sampling future trajectories from the distribution

$$p(\{x_t\}_{t=k+1}^{T} | \{x_t\}_{t=1}^{k}, c). \tag{1.1}$$

This distribution models the future trajectory of the pedestrian in Figure 1.1 as a random sample from the distribution, with the observed positions and contextual information known. Building a prediction model entails choosing what contextual information c is needed and specifying the distribution (1.1). A standard approach is to parameterize the prediction model by a variable $\theta \in \mathbb{R}^n$, typically referred to as the model parameters. This provides the flexibility to calibrate the model to

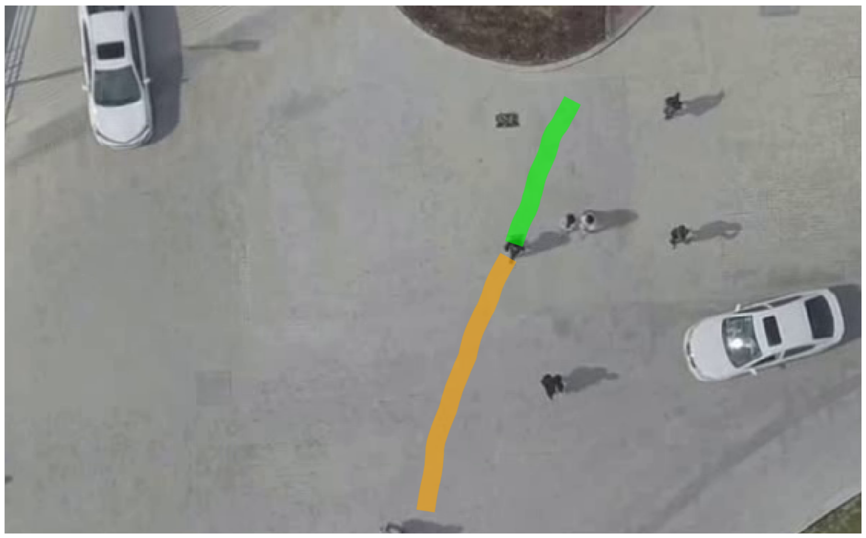


Figure 1.1: A road user in the DUT trajectory dataset. Three seconds of observed positions are shown as a solid green line. The following five seconds of positions are shown as a solid orange line.

recorded trajectory data. The next section outlines two main groups of prediction models based on the differences in specifying the predictive distribution (1.1).

1.4 Model-based and Model-free Predictions

Specifying the form of the predictive distribution (1.1) is a choice of the modeler. The methodology used to make the choice, however, may often be classified as model-free or model-based. The main difference lies in the interpretation of the model parameters.

The model-based approach focuses on building an interpretable model of road user behavior. One set of models assume that the road user is a rational agent who plans future actions. Here, the road user is modeled as choosing actions to minimize costs incurred during travel. This results in an optimal control problem and its solution yields the predictive distribution [31, 32]. The road user's preferences correspond to the costs being minimized in the control problem. While this formulation offers insight into why one action may have been chosen over another, each road user's actual preferences are rarely known. An alternative is to directly approximate the road user's decision-making. This also obviates the need to solve an optimal control problem. Each element of the model parameters $\theta \in \mathbb{R}^n$ often corresponds to a physical quantity, such as the road user's desired speed, gap size, or goal [33, 34]. Maintaining this interpretability most often results in fewer model parameters with $n < 100$. It should be noted that this interpretability is in an engineering sense of understanding what factors shape each prediction from the modeler's perspective, rather than necessarily that of legibility which aims to make the robot's intentions clear to those around it through its motion or other actions [35].

Advances in computing coupled with the creation of high quality datasets has spurred the development of model-free methods. The model-free methodology focuses on building models flexible enough to explain the wide range of behaviors observed in datasets. This is accomplished by the use of models capable of approximating any function, and typically thousands of model parameters to improve the approximation. Deep neural networks (DNNs) [36] and support vector machines [37] are commonly used classes of models having the property of universal function approximation. The modular structure of DNNs and their ability to readily incorporate additional contextual information has led to their widespread use. Fitting models with more parameters than data may result in models that are overly attuned to noise in the dataset, but the use of large datasets and the choice of model can offset this problem. In evaluating these methods we use the term 'experiment' in the machine-learning sense of evaluating performance across many scenarios in collected datasets.

1.5 Sources of Uncertainty

Without clear signaling, a pedestrian who is about to cross the road may be difficult to distinguish from a pedestrian who is about to slow down and yield. Similarity between the trajectory in each case naturally introduces uncertainty. By identifying the subtle differences unique to each trajectory such as speed or distance to curbs, prediction models can eliminate uncertainty. Modeling interactions brings its own set of challenges. Various road users may be in a driver's view, but to what extent the driver is paying attention to each is not clear from this information alone. To add to this, recording gaze information is often performed with instrumentation including in-vehicle cameras or eye-tracking glasses [38]. As a result gaze is rarely observed. Interactions may also occur between road users at varying distances. For instance, a couple walking together will likely continue to walk together. Their yielding to a vehicle on the far side of an intersection provides an interaction over a longer distance. Due to the wide variation, modeling interactions remains an open problem.

1.6 Realtime Computing

Accuracy is a general demand of prediction models, and informs more socially aware decisions. Use on an autonomous vehicle also adds the requirement of realtime predictions. Realtime predictions allow for quickly responding to the predicted future state of the world, increasing the time available for emergency maneuvers. This is especially valuable in highway driving where waiting one second means that the vehicle may have already moved several car lengths along the highway. Even at lower speeds, autonomous vehicles must rapidly process new observations of road users appearing at blind corners or from behind vehicles parked in the street.

1.7 Gaps in the Literature

This thesis focuses on the following problems:

1. Collecting large datasets is costly and limits the testing and development of prediction methods for new environments.

2. State-of-the-art methods sacrifice model interpretability to account for complex interactions in general scenarios. Simultaneously achieving realtime performance presents an additional challenge.

3. Existing interpretable methods make assumptions that limit their applicability to general scenarios.

1.8 Contributions

This thesis seeks to build upon existing prediction methods with the following contributions:

1. Novel simulation framework to leverage small amounts of data to train deep neural networks for pedestrian trajectory prediction on large synthetic datasets, enabling the application of deep neural networks when there is little annotated data, or where the costs of collecting and labeling real data is high. (Chapter II)

2. Low-latency probabilistic model of highway interactions capable of predicting with as little as 400 ms of observations. Development of a realtime and consistent inference procedure enables the estimation of model parameters from observations rather than by manual specification. (Chapter III)

3. A realtime probabilistic model to predict pedestrian trajectories accounting for interactions with vehicles in scenes where traditional traffic devices may not be present. Development of a novel and tractable training procedure allows for estimating the model's parameters while avoiding the auxiliary simulations or manually specified parameters called for in previous model-based works. (Chapter IV)

4. A kinematic model for generating interaction-aware predictions in general highway driving scenarios including multiple vehicle lane changes. The inference procedure developed for this model enables tractable inference in the presence

realistic sensor conditions including significant occlusions and late detection. (Chapter V)

1.9 Summary

Autonomous vehicles rely on predictions of fellow road users' future motions to guide planning for safe and human-like behavior. As a result predictions may specify a large part of vehicle behavior. To support autonomous behavior that is transparent and predictable, this thesis develops interpretable prediction methods. Since interactions between road users influence their future motion, these are also a central focus of this thesis. Especially valuable for planning is that interactions with the autonomous vehicle itself will also influence fellow road users' future motion. In terms of perception, prediction methods must gracefully handle measurements subject to sensor noise or even the lack of measurements due to limited sensor range or occlusions. Building such ideal prediction methods may be hindered by the lack of annotated data, or fitting procedures. Even after fitting a model to data, lack of a realtime procedure to generate predictions from the model may hinder its deployment to autonomous vehicles. The following chapters present frameworks and prediction methods to address these issues along with experiments to evaluate their effectiveness.

CHAPTER II

Stochastic Sampling Simulation for Pedestrian Trajectory Prediction

2.1 Introduction

In crowded urban environments, mobile robots, such as autonomous vehicles (AVs) and social robots, must navigate safely and efficiently while in close proximity to many pedestrians on the road. To avoid collision and ensure a smooth ride, it is crucial for the robots to accurately predict where a nearby pedestrian may move to next. However, the motion and actions of each pedestrian may depend on the behavior of others, which makes it difficult to forecast [39, 40, 41, 1]. Our goal is to predict all possible future locations and trajectories of pedestrians with probability estimates accounting for social interactions.

Classical approaches to forecasting pedestrian trajectories include Kalman filters, Gaussian Processes [**?**], and inverse optimal control [31], which estimates a model for each pedestrian based on past behavior to forecast the future. These approaches have traditionally focused only on predicting single pedestrians without considering the social interactions between different pedestrians. These earlier results have been improved upon by extending the frameworks with hand-crafted features to model social interactions [22].

More recently, deep neural networks (DNNs) have been used to successfully make long-term pedestrian trajectory prediction accounting for social interactions [40, 42, 1, 43, 44]. Instead of using hand-crafted features, they rely on vast amounts of annotated trajectories to learn social interactions directly from the data. However, it is often difficult, if not impossible, to obtain such large datasets with accurate annotation without resorting to either an enormous effort of manual labeling [45, 46] or heavily constructed experiments with instrumented participants [47], both of which are expensive and prone to error. Therefore, it would be valuable to develop a training scheme that requires very small amounts of labeled real-world data, yet still produces satisfactory prediction results for test datasets.

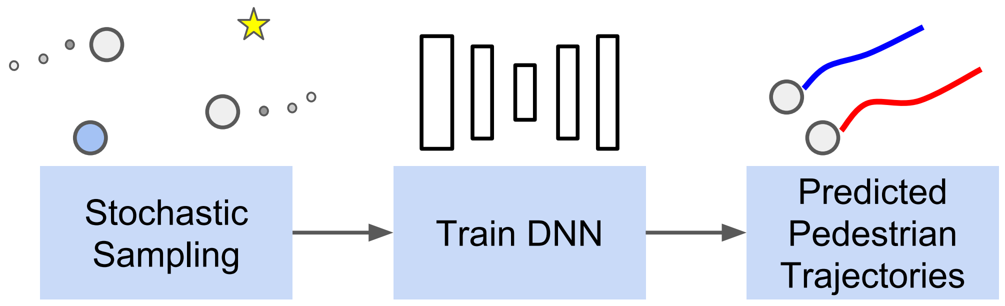


Figure 2.1: System overview. We propose using a novel stochastic sampling-based simulation system to train a deep neural network (e.g., Social GAN [1]) to make socially acceptable pedestrian trajectory predictions.

To address this problem, we propose the use of automatically-annotated, realistic pedestrian simulations to train deep neural networks for 2D (top-down view) trajectory prediction. Generating synthetic data for training DNNs has shown promising results for various applications, including image classification [48], object detection [49], and pose estimation [50]. However, to the best of our knowledge, this chapter is the first that proposes techniques for synthesizing pedestrian trajectories.

Figure 2.1 shows an overview of our system. We develop a novel stochastic sampler that can generate tens of thousands of realistic pedestrian trajectories based on limited real-world annotations. We then use these samples to train state-of-the-art DNNs to predict pedestrian trajectories. In this chapter, we use Social GAN [1] as our main prediction network, which provides state-of-the-art predictions and takes into account the interactions of all pedestrians in the scene. We envision our method being used to train high-performing prediction algorithms (such as DNNs) when there is little annotated data, or where the costs of collecting and labeling real data are high.

Our main contributions include (1) a novel nonparametric model of pedestrians, (2) a method to sample realistic pedestrian trajectories from this model, and (3) experiments training on these sampled trajectories and predicting on benchmark pedestrian datasets, such as the ETH dataset [51, 45] and the UCY dataset [52]. We demonstrate that Social GAN trained on the realistic sampled pedestrian trajectories alone achieves performance surpassing that achieved by training on real-world human-annotated data.

The chapter is organized as follows. Section 2.2 describes related work in data augmentation and simulation as well as related work in trajectory prediction networks. Section 2.3 describes our proposed simulation model and synthetic trajectory generation process. Section 2.4 presents pedestrian prediction results on benchmark datasets. Section 2.5 concludes this chapter.

2.2 Related Work

In this section, we first describe DNN-based pedestrian trajectory prediction methods and our motivation for using the Social GAN. Then, we describe related work in synthetic data generation for training DNNs, such as using physics simulations and domain randomization. Finally, we describe related work in data augmentation and methods for generating synthetic datasets based on real data annotations.

2.2.1 DNN-Based Pedestrian Trajectory Prediction Methods

In recent literature, DNN-based methods, particularly methods based on Long Short-Term Memory (LSTM) networks, have shown successful results in pedestrian trajectory prediction applications [40, 53, 54, 1, 55, 56, 57]. As our method does not consider scene geometry, we refrain from using SS-LSTM [55] or scene-LSTM [57], which incorporates scene information. Instead, we select Social GAN [1] as our main prediction network, which provides state-of-the-art prediction results and takes into account the interactions of all pedestrians in the scene. Most recently, a convolutional neural network (CNN)-based trajectory prediction approach was proposed [44]. This approach uses highly parallelizable convolutional layers to handle temporal dependencies and predict future pedestrian positions. However, this CNN-based approach only handles individual trajectory information and does not consider social interactions as Social GAN does. Therefore, we chose Social GAN as our basic prediction network.

The Social GAN uses a pooling mechanism together with a Generative Adversarial Network (GAN) to learn the social interactions of pedestrians and produces probabilistic trajectory outcomes. The current Social GAN is a purely data-driven approach [1] and benefits from large quantities of annotated trajectory data. However, accurate annotations for large datasets are often difficult, expensive, or impossible to obtain [45, 46, 47]. In this chapter, we develop a stochastic sampling-based simulation system to automatically generate large amounts of annotated, simulated yet realistic pedestrian trajectories to use as training data for a Social GAN, and aim to produce prediction results comparable or better than the Social GAN prediction results trained on real data.

2.2.2 Physics Simulators and Domain Randomization

Physics simulators and engines can generate new data without the aid of existing data. Some studies have focused on crafting synthetic data as similar to real data as possible. In [49] and [58], for example, the Grand Theft Auto (GTA) game engine was utilized to produce automatically-annotated, photo-realistic images for object detection and semantic segmentation. In [59], the OpenRAVE simulator [60] was used to generate a large-scale database for grasps. This simulated grasps dataset was then used to train a DNN for binary classification (stable or unstable grasps) tasks.

In contrast, domain randomization (DR) methods [61, 62] were used to "bridge the gap" between simulation and reality. Domain randomization methods focus on bringing variability to the simulation, typically by varying global parameters (e.g., camera pose, shape and number of objects, texture, lighting) and adding noise to the simulated data without much regard to photo-realism. For example, the images synthesized in [63] contain car models and added geometric objects rendered over random background images. The aim here is to encourage DNNs to learn features invariant to the different kinds of noise added, as well as rendering artifacts and avoid over-fitting. Note that both physics simulations and DR methods do not rely on existing real datasets or annotations.

2.2.3 Data Augmentation and (Real-)Data-Driven Synthesis

Data Augmentation (DA) is another approach to enlarge and enhance training datasets by performing a variety of transformations, typically on images [48]. Data augmentation methods usually seek to increase the amount of training data by generating new artificial data from existing real data while preserving label information. Common forms of DA include image translation, horizontal flips/reflections, crops, and perturbations to color (intensity) values [48]. These methods have been widely used in image classification [48] and object detection [64]. Other applications include acoustic modeling [65] and natural language processing [66]. So far, we are not aware of any standard label-preserving transforms designed specifically for non-image-based pedestrian trajectories, due in part to the need to account for interactions among pedestrians and scene geometry.

In addition to image transformation, several methods have been proposed to transfer the style between real and synthetic data, adopting the GAN framework [67, 68, 69]. Rogez and Schmid [50] proposed a synthesis engine to augment existing real images with manual 2D pose labels into 3D poses using 3D Motion Capture (MoCap) data. In a way, these methods were trying to automatically learn the relationship, or transformation, between real and synthetic datasets instead of performing predefined transformation as DA usually does.

Our chapter is most similar to [50] in that we also use labeled real-datasets to generate a new, larger synthetic dataset for training. Unlike [50] which synthesizes images for pose estimation, we simulate realistic pedestrian trajectories. We also perform a variety of perturbations to generate our synthetic data, inspired by DA methods. We show that, with our proposed method, the DNN trained on synthetic data outperforms when trained on real data, even when the synthetic data is generated from a small amount of real annotations.

2.3 Stochastic Sampling-Based Simulation

In this chapter, the overall task is to predict future pedestrian trajectories given each pedestrian's previous positions considering social interactions. To do so, we aim to generate large amounts of synthetic pedestrian trajectories for training a Social GAN. A limited amount of real data is given to our simulation system so we can generate realistic trajectories based on how pedestrians actually walk from observed real datasets. In this section, we define the notations and pre-computation steps used in our method and then describe our simulation method in detail.

2.3.1 Notations and Pre-computation

Let $\mathbf{x}_{tk}$ denote the 2D position (top-down view) at time t for the k^{th} pedestrian, $\mathbf{x}_{tk} \in \mathbb{R}^2$. Since we utilize a possibly small, real dataset to generate our simulation dataset, we denote $\mathcal{D}_\mathcal{R}$ as the given real dataset and $\mathcal{D}_\mathcal{S}$ as the generated simulation dataset. In our system we aim to generate a much larger synthetic dataset, such that $|\mathcal{D}_\mathcal{S}| \gg |\mathcal{D}_\mathcal{R}|$, where $|\mathcal{D}|$ refers to the size of the dataset $\mathcal{D}$.

We denote the total number of unique pedestrians in the real dataset $\mathcal{D}_\mathcal{R}$ by K. At each frame (timestep) in $\mathcal{D}_\mathcal{R}$, we record the number of pedestrians in the scene as K_p. K is a known constant for $\mathcal{D}_\mathcal{R}$, while K_p may change from frame to frame as pedestrians are entering and exiting the scene. From K_p, we can compute the average number of pedestrians in a frame as μ_p and the variance of the number of pedestrians in the scene as σ_p^2.

From $\mathcal{D}_\mathcal{R}$, we can also compute the walking speed for each pedestrian, following

$$s_{tk} = \frac{||\mathbf{x}_{(t+1)k} - \mathbf{x}_{tk}||}{\Delta t}, \tag{2.1}$$

where $\mathbf{x}_{tk}$ denotes the 2D position at each timestep t for pedestrian $k = 1, ..., K$, Δt is the difference in time between two frames/timesteps (fixed), and $|| \cdot ||$ denotes Euclidean distance. Figure 2.2 shows a simple illustration for computing the speed for two pedestrians. Note that a pedestrian in $\mathcal{D}_\mathcal{R}$ may appear in sequences of varying lengths (due to entering and exiting the scene or due to available data). We denote the sequence length (length of observed timesteps) as T_k for pedestrian k in $\mathcal{D}_\mathcal{R}$. Also, note that the speed s_{tk} can vary in each step for real pedestrians.

Given the speed for each pedestrian at each timestep, we can compute the average walking speed for k^{th} pedestrian as $\bar{s}_k$. In our method, we assume that all persons

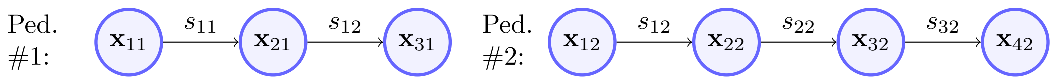


Figure 2.2: An illustration for computing pedestrian trajectory speed. Suppose we observed two pedestrians in $\mathcal{D}_\mathcal{R}$; pedestrian 1 appeared in a sequence of three timesteps ($T_1 = 3$), pedestrian 2 appeared in a sequence of four timesteps ($T_2 = 4$). The $\mathbf{x}_{tk}$ denotes the X-Y position at each timestep t for pedestrian k. The speed at each timestep can be calculated using (2.1).

walk with the same speed variation between timesteps and we compute σ_s^2 as the pooled variance across all $s_{tk}, \forall t, k$. Note that $\bar{s}_k$ changes for each pedestrian and σ_s^2 is the same for all pedestrians. Constraining the variance to be the same helps reduce the model dimensionality.

The above summary statistics reflect how pedestrians walk in the real dataset and are used later to generate synthetic trajectories in our sampling scheme.

2.3.2 Sampling Number of Pedestrian and Walking Speed

In our simulation, we use stochastic sampling to generate realistic pedestrian trajectories. In this section, we describe the method to sample the number of pedestrians and walking speeds for the simulated dataset.

Let n_p denote the number of pedestrians in a frame in the simulated dataset. In simulating a single set of pedestrians, we sample n_p based on statistics from the given real dataset. We already obtained the average number of pedestrian in a frame μ_p and the variance of the number of pedestrians in the scene σ_p^2 from Section 2.3.1. We assume the number of pedestrians at each time follows a normal distribution $\mathcal{N}(\mu_p, \sigma_p^2)$ left-truncated at zero, which we denote with $\mathcal{N}(\mu_p, \sigma_p^2, 0)$. We can then sample $n_p \sim \mathcal{N}(\mu_p, \sigma_p^2, 0)$.

Regarding walking speed, we model each pedestrian as walking at a desired constant speed. Denote $s^{(i)}$ as the speed of the i^{th} sampled pedestrian. We use the superscript to distinguish between the index of the stochastically sampled and real datasets. For each i, we first uniformly sample an average speed value, $\bar{s}^{(i)}$, from the pool of average speeds from real pedestrians, denoted as $\bar{s}^{(i)} \sim \mathcal{U}(\{\bar{s}_k\}_{k=1}^{K})$. The variance of speed is assumed to be the same as real pedestrians, σ_s^2. Then, we can sample $s^{(i)}$ based on a truncated normal distribution, $\mathcal{N}(\bar{s}^{(i)}, \sigma_s^2, 0)$.

2.3.3 Pedestrian Trajectory Sampling

Based on the number of pedestrians and walking speeds sampled above, we can determine the actual paths of the pedestrians. We generate the pedestrian trajectories by assigning the sampled speeds to these paths.

We represent the real dataset as a collection of trajectories, i.e., $\mathcal{D}_\mathcal{R} = \{f_k\}_{k=1}^{K}$, where f_k is the trajectory for the k^{th} pedestrian. The trajectory is given by the

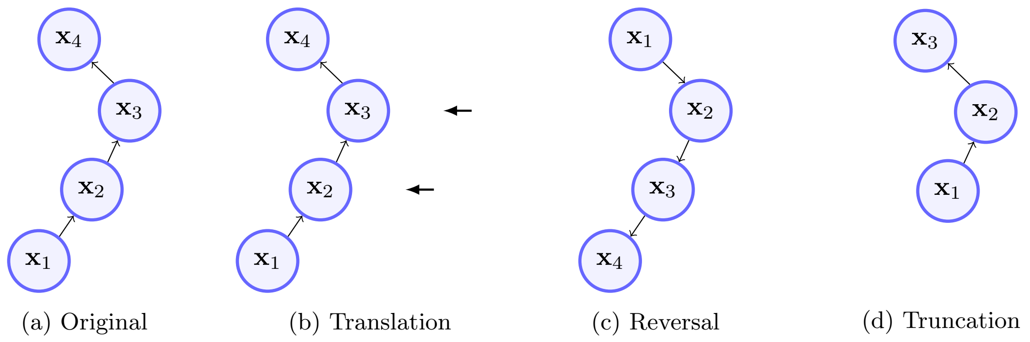


Figure 2.3: An example of the perturbations used to sample pedestrian paths. Suppose (a) is a sample trajectory from real data, (b) represents the "translation" perturbation, with the arrows representing horizontal displacement. (c) is an example of the "reversal" perturbation where the start and final locations as well as waypoints are flipped. (d) shows an example of "truncation" perturbation where the last location was truncated.

sequence of positions $f_k = \{\mathbf{x}_{tk}\}_{t=t_k}^{T_k}$, for pedestrian k present in the scene from time t_k to T_k. For each sampled pedestrian i, we first uniformly sample a trajectory, $f^{(i)}$, from the pool of all real pedestrian paths, denoted as $f^{(i)} \sim \mathcal{U}(\{f_k\}_{k=1}^{K})$. Then, we apply the following three types of user-defined perturbations to the $f^{(i)}$:

- Translation by an amount $\Delta\mathbf{x} \sim \mathcal{U}([-r, r] \times [-r, r])$, where r is the displacement in each axis of the 2D plane.
- Reversal with probability p_r: We reverse the start and ending locations as well as all the waypoints in between.
- Truncation by a random number of steps.

Fig 2.3 shows an illustration for the perturbations. The pedestrian then follows the path g, a piecewise linear spline fit [70] to the perturbed $f^{(i)}$. All synthetic pedestrians have fixed $N+1$ timesteps, and we denote a sampled set of trajectories as the set $\mathcal{X}^{\mathcal{S}} = \{\mathbf{x}_{li} | i = 1, ..., n_p, l = 1, ..., N+1\}$, for $N+1$ timesteps and n_p pedestrians in the scene.

We run Algorithm 1 for a user-defined M number of times to generate the entire large-scale simulated trajectory dataset. In practice we see that $M > 20$ produces datasets yielding competitive models, with better performance for larger M.

2.4 Experiments

To measure the effectiveness of the proposed method, we first generated synthetic datasets from the sampling-based simulation method described above. We employ Social GAN [1], a state-of-the-art deep learning trajectory prediction network architecture, to perform a prediction based on our simulated data. We evaluate our

Algorithm 1: Stochastic Sampling for Pedestrian Trajectory Simulation

```
   Input: $\mathcal{D}_{\mathcal{R}} = \{f_k\}_{k=1}^{K}$, $\{\bar{s}_k\}_{k=1}^{K}$, $N$, $\Delta t$, $\mu_p, \sigma_p^2$, $\sigma_s^2$
   Output: $\mathcal{X}^{\mathcal{S}}$
1  $n_p \sim \mathcal{N}(\mu_p, \sigma_p^2, 0)$
2  foreach $i = 1, ..., n_p$ do
3      $\bar{s}^{(i)} \sim \mathcal{U}(\{\bar{s}_k\}_{k=1}^{K})$
4      $s^{(i)} \sim \mathcal{N}(\bar{s}^{(i)}, \sigma_s^2, 0)$ // see Section 2.3.2
5      $f^{(i)} \sim \mathcal{U}(\{f_k\}_{k=1}^{K})$
6      $\tilde{f}^{(i)} \leftarrow$ peturb($f^{(i)}$) // see Section 2.3.3
7      $g \leftarrow$ spline($\tilde{f}^{(i)}$)
8      foreach $l = 1, ..., N+1$ do
9          $\mathbf{x}_{li} \leftarrow g(s^{(i)} \Delta t l)$
10     end
11 end
```

method on two widely used pedestrian trajectory datasets. The ETH dataset [51, 45] contains over 850 labeled frames of data in each of two distinct scenes, *ETH* and *Hotel*. The UCY dataset [52] also has two scenes, *Zara* and *University*, each with over 1500 labeled frames. Using leave-one-out cross validation, we evaluated the Social GAN's prediction outputs on each scene, having trained on data from the other three scenes.

2.4.1 Baselines

Since the focus of this chapter is synthetic trajectory dataset generation, we compare Social GAN prediction results trained on the following four methods:

- **Real:** Train on all the available frames of real data in each scene (approximately 5,000 frames in total).
- **Synth-Large:** Train on a large synthetic dataset. We sampled simulated trajectories 500 times ($M = 500$) using Algorithm 1 with $N = 20$ timesteps for each scene. When sampling from *University* dataset, we sampled $M = 100$ times due to the large numbers of pedestrians in the scene. This yields over 20,000 simulated, labeled frames in every train-test cross validation split.
- **Synth-Equal:** Train on a synthetic dataset that has the same size as the real dataset (which is much smaller than the size of Synth-Large). We sampled simulated trajectories such that the number of frames of simulated pedestrian trajectories is equal to that of the real data for each scene.
- **Real + Synth-Large:** Train on the combined data from Real and Synth-Large. This is to evaluate the effect of including real data in training.

The Synth-Large dataset has over 15k more frames in in each cross validation split than using 100% of real data ("Real-100%"). In addition, we examined the effect of using an even smaller number of labeled frames of real data. We randomly selected 20% of the real data from each scene (around 1.2k frames) and used this to train the Social GAN. We also generated a separate set of synthetic datasets based only on the 20% real data and reported prediction results as well. These results were reported under "20%" columns in Table 2.2.

2.4.2 Evaluation Metrics

Since Social GAN makes probabilistic predictions, we treat the predicted position for the i^{th} pedestrian at timestep t as a random variable $\mathbf{y}_{ti}$. To thoroughly sample the predictive distribution for $\mathbf{y}_{ti}$, we made 100 probabilistic predictions of the pedestrian's possible locations. We denote the ground truth position as $\mathbf{x}_{ti}$. Similar to prior work [40, 1], we used the following error metrics:

Average Displacement Error (ADE)

Expected distance between the ground truth (GT) pedestrian location and the probabilistic prediction. We estimate this for the dataset by averaging across all N_p pedestrians in the dataset and all predicted timesteps ($t = 1, ..., T$) as

$$ADE = \frac{1}{N_p \times T} \sum_{i=1}^{N_p} \sum_{t=1}^{T} \mathbb{E}[||\mathbf{y}_{ti} - \mathbf{x}_{ti}||], \tag{2.2}$$

where $\mathbb{E}[\cdot]$ refers to the expected value.

Minimum Displacement Error (MDE)

Minimum distance between the GT pedestrian location and our predictions, averaged across pedestrians and timesteps. Denoting the jth probabilistic prediction by $\mathbf{y}_{ti}^{(j)}$, the MDE is given by

$$MDE = \frac{1}{N_p \times T} \sum_{i=1}^{N_p} \sum_{t=1}^{T} \min_j \{||\mathbf{y}_{ti}^{(j)} - \mathbf{x}_{ti}||\}. \tag{2.3}$$

Final Displacement Error (FDE)

Expected distance between the GT pedestrian location at the final time step T and the predicted final position. This is averaged across pedestrians, written as

$$FDE = \frac{1}{N_p} \sum_{i=1}^{N_p} \mathbb{E}[||\mathbf{y}_{Ti} - \mathbf{x}_{Ti}||]. \tag{2.4}$$

The ADE provides a measure of spread in the predictions, in that a model producing a large spread will necessarily have a large ADE. Unless the model makes predictions near the pedestrian, a small spread in predictions will not ensure a low ADE. The MDE reflects recall in the "best case", measuring the closest prediction to the pedestrian. The FDE is equivalent to ADE measured only at the final timestep. Ideally, we want the ADE, MDE, and FDE to all have low values to show that all the predictions are close to each pedestrian GT location.

2.4.3 Training Parameters

The Social GAN network architecture is trained with a learning rate of 0.001 and batch sizes of 64 for 200 epochs, following the training procedure in [1] for each experiment. In alignment with this, we use a timestep of $\Delta t = 0.4$ s when sampling pedestrian trajectories. Predictions are made by observing pedestrian trajectories for 8 timesteps (3.2 s) and making predictions for the next 8 timesteps ($T = 8$).

For each scene we separately calculate the mean and standard deviation for the number of pedestrians at each timestep (μ_p and σ_p), and the standard deviation about their desired speeds (σ_s). These values are also used when sampling from the smaller amounts of real data, as these can be reliability estimated with much less effort than that needed to build a dataset. The summary statistics calculated for each scene are given in Table 2.1. Note that the *University* scene alone ("Univ" row) contains larger mean and variance for number of pedestrians compared with the rest.

Table 2.1: Summary statistics for each scene.

Dataset	μ_p	σ_p	σ_s
ETH	6.15	4.46	0.35
Hotel	5.60	3.41	0.15
Zara	7.36	3.95	0.25
Univ	26.77	20.31	0.27

2.4.4 Comparison of Prediction Performance

Table 2.2 shows the ADE, MDE, and FDE comparison results across all cross validation datasets, predicted using Social GAN trained on various dataset generation methods. Training on large amount of sampled data in "Synth-Large" achieves the best performance, producing lower prediction errors than training on 100% real data. Figure 2.5 shows a qualitative comparison for both of these models. This lower error performance holds true for "Synth-Large-20%" as well, where we sampled from only 20% of the real data to make the synthetic dataset. We can attribute part of this high performance to the increased amount of sampled data used for training. "Synth-Large" outperforms "Synth-Equal", where the only difference is the amount of sampled data (15k more frames in “Synth-Large”). The realistic variations contained in the additional labeled frames allows for learning a better representation for the true distribution of pedestrian trajectories. This performance difference is especially pronounced in the ADE. We observe similar performance when trained on 100% versus 20% real data, where using more training data increases the performance.

When adding the real dataset to the sampled dataset (in “Real+Synth-Large”), we do not observe strictly increased performance compared to training on “Synth-Large” alone. While the ADE have increased slightly when adding the real dataset, the MDE has decreased. The lack of large performance increases makes intuitive sense, since the synthetic data is sampled from the real data and the pedestrian statistics from real data is largely contained in these stochastically sampled trajectories.

In the next section, we will show that adding real data promotes the expression of uncertainty in the DNN, which aids in lowering the MDE. We also show that the small decreases in MDE depend more on the variations in the sampled data than the amount of data sampled through an ablation study.

Table 2.2: Prediction performance across all datasets and methods (best in **bold** and second best underlined). The lower the errors, the better the performance. All errors are reported in meters. The μ refers to the mean error value across all datasets for each of the ADE, MDE, and FDE evaluation metrics. Each row corresponds to results on a test dataset. For example, the first row reports the ADE values when testing on the *ETH* dataset while trained on the other three datasets.

		Real		Real+Synth-Large		Synth-Equal		Synth-Large	
Metric	Dataset	20%	100%	20%	100%	20%	100%	20%	100%
ADE	ETH	0.95	0.82	0.82	0.77	0.96	0.80	0.79	**0.75**
	Hotel	0.83	0.63	0.48	0.64	0.70	0.57	**0.43**	**0.43**
	Zara	0.96	0.39	0.36	0.39	0.68	0.37	**0.30**	0.35
	Univ	0.72	0.55	**0.37**	**0.37**	0.46	0.38	0.38	0.38
μ		0.86	0.60	0.51	0.54	0.70	0.53	**0.47**	0.48
MDE	ETH	0.56	0.51	0.45	**0.38**	0.55	0.42	0.45	0.40
	Hotel	0.47	0.17	0.15	**0.12**	0.17	0.16	0.13	**0.12**
	Zara	0.44	0.10	0.12	**0.09**	0.14	0.11	0.13	0.11
	Univ	0.37	0.21	0.18	0.15	**0.13**	0.15	0.20	0.17
μ		0.46	0.25	0.23	**0.19**	0.25	0.21	0.23	0.20
FDE	ETH	1.79	1.61	1.65	1.55	1.76	1.56	1.59	**1.50**
	Hotel	1.55	1.29	0.99	1.33	1.25	1.10	**0.84**	0.86
	Zara	1.77	0.81	0.75	0.84	1.27	0.75	**0.62**	0.72
	Univ	1.31	1.06	0.78	**0.77**	0.91	**0.77**	0.78	0.78
μ		1.60	1.19	1.04	1.12	1.30	1.05	**0.96**	**0.96**

2.4.5 Ablation Study

In our ablation study, we removed the dataset fitting terms σ_s and σ_p from the sampler to see their effect on the performance. Upon removing these terms, we sampled from their respective densities without variance, which is equivalent to setting σ_s or σ_p to zero.

We sampled large datasets from 100% of the real data following the same procedure as for generating Synth-Large, once with $\sigma_s = 0$ and again with both $\sigma_s = 0, \sigma_p = 0$. We compared these "reduced models" to the Synth-Large results with full pedestrian statistics. These reduced models both attain an average ADE of 0.47 m and FDE of 0.95 m, compared to 0.48 m and 0.96 m of the full model. On the other hand, both have higher MDE. Removing σ_s from the sampler increases MDE from 0.20 m to 0.22 m; further removing σ_p increases this to 0.23 m.

We also report an additional quantile-based metric to evaluate their performances. Recall that for each pedestrian at each predicted timestep, Social GAN produced 100 probabilistic predictions. This quantile-based metric is defined by sorting the predictions by their distance to the ground truth pedestrian locations and calculating the average distance for each quantile. Figure 2.4 shows the quantile-based distance metric across all training datasets. Ideally, we want the curve to have low distance value across all quantiles. Naturally, the higher the quantile value, the higher the distance (since we sorted the distances in ascending order). The further the curve is shifted towards the top-left, the better the performance. Since we have 100 predictions, the "quantile" here is equivalent to "percentile".

The lowest point on each curve is equivalent to MDE, since it represents the closest (minimum) distance. The full sampler (with non-zero σ_s and σ_p values) performs the best below the 50% percentile (when quantile is less than 0.5 on the plot). The model with no σ_s, on the hand hand, outperforms that the no σ_s, σ_p model at the minimum distance as well as in the middle quantiles. All three synthetic-based models outperform the model trained on real data by a great margin across all percentiles.

The more gentle slope for the real model in Figure 2.4 reflects a greater spread in distances to the ground truth locations as well as uncertainty in the probabilistic predictions. Expressing this uncertainty is important for safety-critical applications such as autonomous driving, since we would like to avoid hitting any predicted locations a pedestrian may potentially be. Training on the sampled data without real-pedestrian statistics (no σ_s and σ_p) reduces this expression of uncertainty and results in the steeper slopes. Including real-pedestrian statistics and calibrating the sampler to the real data before sampling can help recover this uncertainty, as shown in Figure 2.4 by the less steep slopes for the full model, where σ_s and σ_p were added.

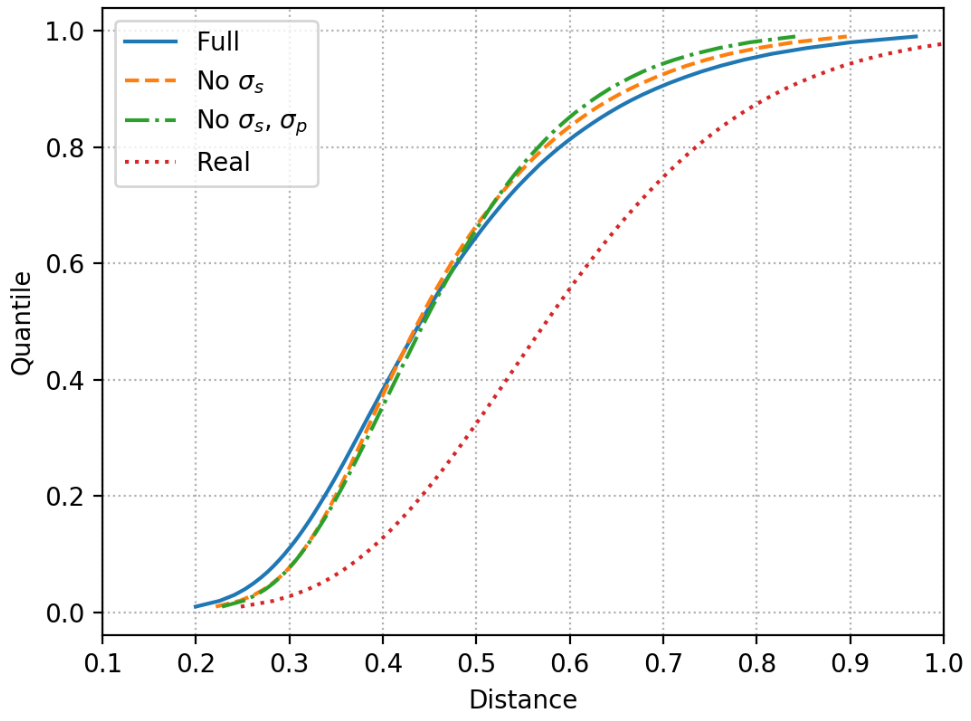
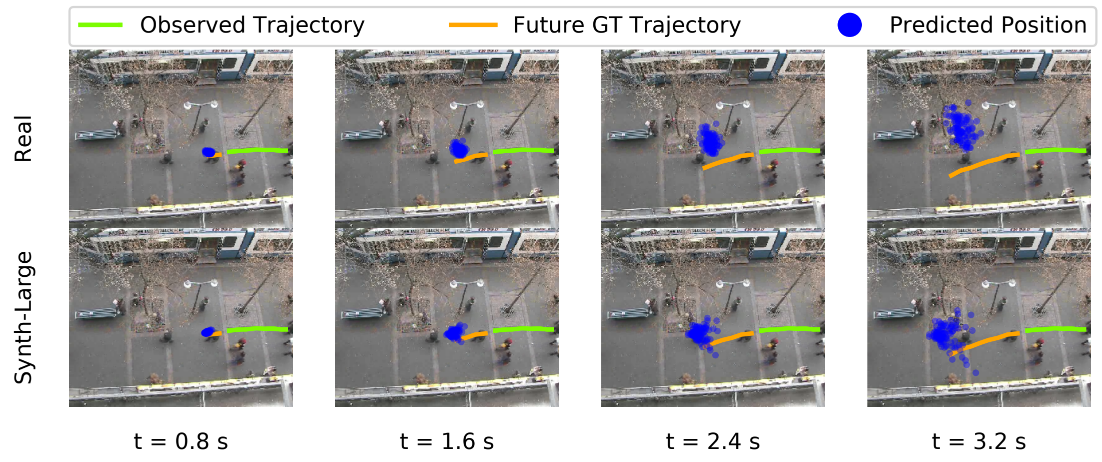


Figure 2.4: Performance curve for the ablation study using quantile-based metric. The model trained on synthetic trajectory samples has lower distance error than trained on real dataset across all percentiles. Removing the real-pedestrian statistics terms from the sampler reduces the expression of uncertainty.

2.5 Conclusion

In this chapter, we presented a novel stochastic sampling method for simulating realistic pedestrian trajectories. We developed a model to extract pedestrian number and walking speed from a small real dataset, and used this information to sample synthetic pedestrian trajectories. We trained a Social GAN on the sampled datasets and evaluated the prediction results on a variety of benchmark datasets of pedestrian trajectories. We show improved prediction performance when trained on large amounts of synthetic data generated by the proposed sampling scheme when compared with trained on real datasets. We also performed an ablation study on the effect of the pedestrian statistics and show that our extracted pedestrian parameters can represent how pedestrians walk in real dataset and allow the DNN to more accurately model the true distribution of pedestrian trajectories.

Future directions include extending the sampling method to incorporate scene geometry, and training a DNN that utilizes the scene information such as [57] on the synthetic datasets. Sampling from the space of interactions, such as sampling the outcomes of pedestrian yielding, is another direction.



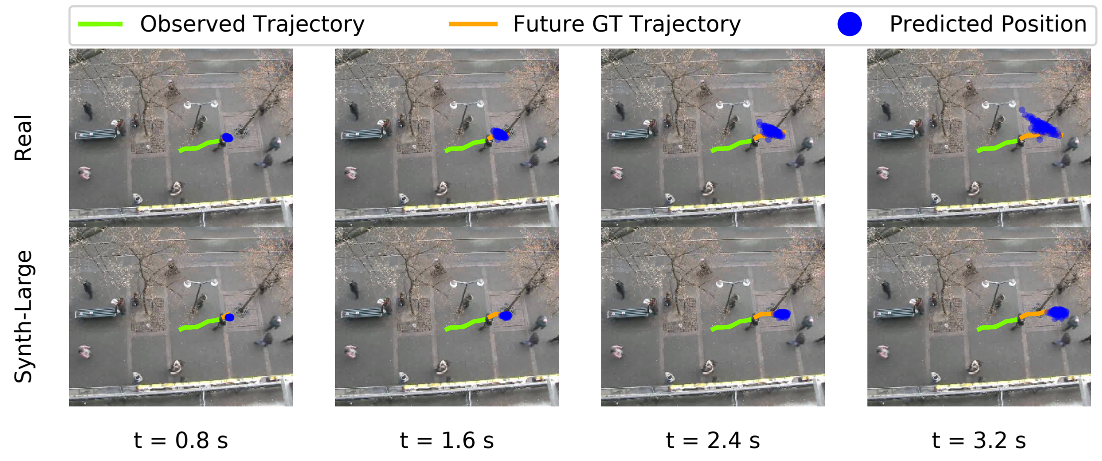


Figure 2.5: Two examples of prediction results from Social GAN trained on “Real-100%” (100% real data only) and on “Synth-Large-100%” (large synethic data sampled from all real data). We show the pedestrian’s predicted position for 2, 4, 6, and 8 timesteps into the future. The Social GAN jointly predicts future positions for all timesteps. The green solid line represents the observed trajectory (in the past). The orange solid line represents the ground truth trajectory for future timesteps. The blue dots represent the 100 probabilistic predictions of pedestrian locations. The ”Synth-Large” predictions are closer to the ground truth position and can obtain decreased error across all evaluation metrics.

CHAPTER III

Low-Latency Trajectory Predictions for Interaction Aware Highway Driving

3.1 Introduction

Merging in dense traffic necessitates cooperating with other drivers. Successful cooperation in turn relies on predicting others' actions. Predicting a vehicle's trajectory, however, is complicated by its possible interactions with surrounding vehicles [71, 72]. Recent works based on deep neural networks (DNNs) have proven effective at modeling these interactions [73, 74, 75, 76, 77, 1, 78, 9, 79]. These methods utilize a fixed number of observations of surrounding vehicles to infer which trajectories are likely. A drawback to this is the duration of time needed to collect observations before making predictions, ranging from 3 s up to 5 s. This window of time determines the minimum delay between first seeing a vehicle and predicting its trajectory. While observation windows of 3 s may be tenable in low speed environments, the high speeds in highway driving call for faster reaction times. Occlusions and sensor limitations such as maximum range also impact the quality of any observations of the surrounding vehicles. This motivates the need for predictions that can be made in short time and with few observations. In this chapter we aim to strike a balance between the richness of interactions modeled and the number of observations needed to make predictions. Merging onto the highway in particular poses a challenge to autonomous vehicles (AVs). In addition to the delay induced by observation windows, limited ramp length will further restrict the time available to make decisions. Due to the heightened need of fast reaction times when merging onto the highway, we focus on these scenarios. The main contributions of this chapter are:

1. a novel and low-latency probabilistic highway interaction model capable of predicting with as little as 400 ms of observations;

2. a realtime and consistent inference procedure that enables the estimation of model parameters from observations rather than by manual specification;

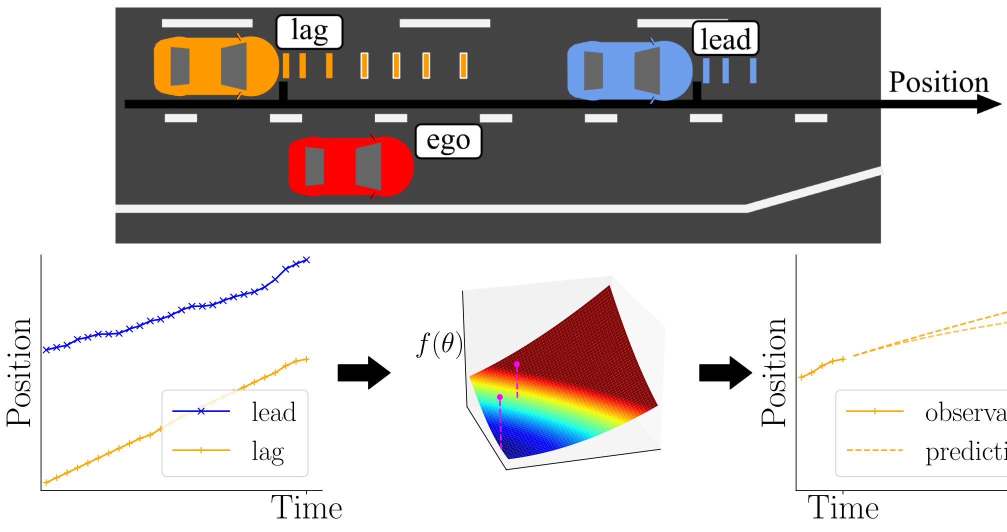


Figure 3.1: Overview of prediction method. The ego vehicle (red) seeks to predict the trajectories of the front and rear (lead and lag) vehicles to enable a safe merge between them. Observations of both vehicles (blue and orange marks) are used to define a likelihood function over possible controllers θ for the lag vehicle. Solving a convex problem yields an estimate that is used to sample realistic trajectories.

3. evaluation on merge scenarios in the real-world NGSIM dense highway traffic dataset [80].

We first extend the deterministic car-following model proposed by Wei et al. [81] to the probabilistic setting. Instead of choosing model parameters by hand, we treat them as unknown random variables and estimate them from observed velocities of the lead and lag vehicles depicted in Figure 3.1. Though the resulting estimation problem is nonconvex, we prove that it is equivalent to a semidefinite program and solve it in realtime with an off-the-shelf solver. The estimate for the model parameters is then used to sample realistic trajectories.

The chapter is organized as follows. Section 3.2 describes related works in interaction-based trajectory prediction for traffic participants in general scenarios and those focused on ramp merging. Section 3.3 describes the interaction model we use to make predictions and the inference procedure used to determine the probabilities of different outcomes. We evaluate our model on the NGSIM dataset and perform an ablation study in Section 3.4 before concluding in Section 3.5.

3.2 Related Work

We first describe prediction methods that may operate on highly restricted observation windows, but do not take into account interactions. Methods that focus on modeling interactions, but operate on longer observation windows, are described in the next section. In the last section we describe methods designed specifically to account for the interactions and restricted observation windows in ramp merging scenarios.

3.2.1 Single Agent Prediction

Classical methods specify a simple kinematic model to predict trajectories, such as constant velocity or constant yaw rate and acceleration [82]. Other methods have employed learning based approaches such as Gaussian mixture models [83], or hidden Markov models to ensemble simple kinematic models [84]. More similar to the method proposed in this chapter, Houenou et al. [85] combine predictions from a simple kinematic model and weight these with penalty terms on future accelerations. These methods, however, do not account for interactions between different vehicles on the road. This can lead to inconsistent predictions in common scenarios such as a fast vehicle needing to slow down for a vehicle in front.

3.2.2 General Interaction-Based Trajectory Prediction

More recently, DNNs have been used to model the interactions between multiple vehicles. Early works account for interaction but do not make probabilistic predictions [79, 74]. Treating all other agents as obstacles and predicting occupancy grids offers another approach, but this loses individual tracking labels and is limited by coarse grid size [73]. Directly predicting the parameters of a known distribution has been employed in most probabilistic methods, using a bivariate normal distribution [75, 76, 77, 78]. The works of [75, 76], however, require additional labels for maneuver types. Another method, Traphic [77], requires no such labels, but performs a potentially slow social pooling operation for each agent at each timestep. TrafficPredict [78] uses an attention based mechanism to extract features for all pairwise interactions which scales poorly with the number of potentially interacting agents. Recent works have avoided social pooling and attention mechanisms to reduce computational complexity [1, 9]. Social GAN [1] introduces a permutation invariant pooling layer to account for distant interactions while using a Generative Adversarial Network architecture to predict all timesteps in a single forward pass. Multi-Agent Tensor Fusion [9] instead uses a global pooling layer to avoid pooling for each agent separately, as well as preserve spatial structure. The method proposed in this chapter does not account for lane changes, but for ramp merging scenarios most vehicles on the highway will cooperatively merge to an inner lane if at all merging, to avoid interacting with vehicles entering the highway [86, 87].

Other works not based on neural networks have relied on manually defined cost functions to specify vehicles' behavior [88], or solving integer linear programs [89].

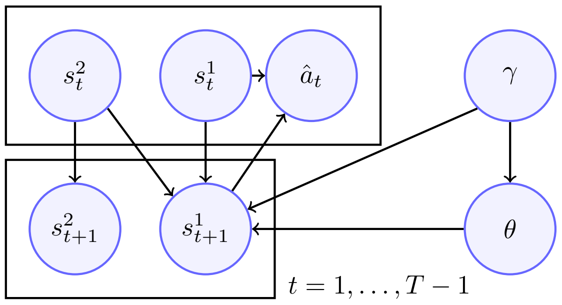


Figure 3.2: Proposed interaction model for predicting lag vehicle behavior. The lag vehicle state s^1 depends on the lead vehicle's state s^2, its own controller θ, and hyperparameters γ. Noisy estimates of lag vehicle acceleration $\hat{a}_t$ are calculated from state measurements.

These methods, however, have not performed as well as those based on neural networks.

3.2.3 Ramp Merging Trajectory Prediction

In this chapter we focus on modeling the interactions between the lag vehicle and lead vehicle shown in Figure 3.1. A complementary body of research has instead focused on modeling the interactions between the lag vehicle and ego vehicle [81, 90, 91]. In these works the goal is to infer whether or not the lag vehicle will yield to make space for the ego vehicle, or not yield. The lag vehicle is then modeled as following a controller specific to the yield intent. The controllers have been modeled as Markov chains [90] and with the Intelligent Driver Model (IDM) [92] with manually chosen parameters [91].

3.3 On-Demand Trajectory Predictions

Here we define our model of interactions between the lag and lead vehicles. We then show how this is used to predict trajectories. Section 3.3.1 states the trajectory prediction problem in our probabilistic setting. Section 3.3.2 defines the interaction-based controller whose parameters we aim to estimate, and the dynamics of the system. The full probabilistic model with novel regularization terms is defined in Section 3.3.3. The realtime inference procedure for predicting trajectories with this model is described in Section 3.3.4.

3.3.1 Problem Statement

We consider the ramp merging scenario in Figure 3.1 depicting the ego vehicle merging onto the highway. To enable safe merging we are interested in predicting the longitudinal positions of the two (lead, lag) vehicles in the target lane. Let $s_t^i = (x_t^i \ v_t^i)^\intercal \in \mathbb{R}^2$ denote vehicle state, consisting of longitudinal position x_t^i and velocity v_t^i at timestep t. We denote the state of the lag vehicle by s_t^1 and state of the lead vehicle by s_t^2. We observe the state of both vehicles $s_t = (s_t^{1\intercal}\ s_t^{2\intercal})^\intercal$ at timesteps $t = 1, ..., k$ and predict x_t^1 until the final timestep, $t = k+1, ..., T$. Additionally we will use the subscript notation $i : j$ to refer to the set of variables indexed by $i, i+1, \ldots, j$. In making probabilistic predictions this amounts to sampling

$$x_{k+1:T}^1 \sim p(x_{k+1:T}^1 | s_{1:k}). \tag{3.1}$$

We now explain our focus on the lag vehicle. With several assumptions summarized in the graphical model shown in Figure 3.2, we decompose the joint prediction of both vehicles into two parts. The first part predicts the lead vehicle's trajectory and the second predicts the lag vehicle's trajectory conditioned on that of the lead vehicle. We start with the assumption of not having observations for the vehicle in front of the lead vehicle. We model the lag vehicle behavior as dependent on the state of the lead vehicle, yet we do not model the state of the lead vehicle as dependent on its own lead. One reason for this inconsistency is that occlusions and limited sensor range may prevent us from obtaining such observations. Aside from this, the proposed model could be extended to account for the lead vehicle's own lead by repeated decomposition, but there is a point at which we cannot see further vehicles ahead. We thus present the simplest model here. Next, assume that the lead vehicle's actions do not depend on the lag vehicle's state as in common car-following models. Furthermore, assume measurements of each vehicle's position and velocity are noiseless, while the acceleration measurement $\hat{a}_t$ of the lag vehicle has zero mean Gaussian noise with a small variance σ_a^2. This is reasonable when the former measurements have low variances but acceleration is approximated from velocity via finite differences. For example, given velocity measurements with variance σ_v^2 and timestep size Δt, acceleration then has variance $\mathrm{var}(\hat{a}_t) = \mathrm{var}(v_{t+1} - v_t)/\Delta t^2 = 2\sigma_v^2/\Delta t^2$. The small timestep will magnify the variance as in the NGSIM dataset. Using the independence assumptions in graphical model shown in Figure 3.2 we may write

$$\begin{aligned} &p(s_{k+1:T} | s_{1:k}) \\ &= p(s_{k+1:T}^1 | s_{1:k}, s_{k+1:T}^2) p(s_{k+1:T}^2 | s_{1:k}) \\ &= p(s_{k+1:T}^1 | s_{1:k}, s_{k+1:T}^2) p(s_{k+1:T}^2 | s_{1:k}^2) \end{aligned} \tag{3.2}$$

which provides the problem decomposition. Throughout the remainder of this chapter we focus on the prediction problem for the lag vehicle posed as

$$s_{k+1:T}^1 \sim p(s_{k+1:T}^1 | s_{1:k}, s_{k+1:T}^2) \tag{3.3}$$

from which we obtain the predicted positions.

3.3.2 Interaction Model

Here we describe the controller used to model the interactions between the lead and lag vehicle. The controller is based on balancing two goals. The first is to match the speed of the lead vehicle, and the second is to maintain a desired gap to the lead vehicle. Let g_t denote the current gap between the lead and lag vehicles. This gap is calculated from their positions as $g_t = x_t^2 - x_t^1 - l$, where l is the length of the lead vehicle. We denote the desired gap by g_*. Denoting k_v and k_g as the proportional control gains for the desired speed and desired gap, respectively, we denote the parameters that define this controller by $\theta = (k_v \; k_g \; g_*)$. The controller proposed in [81] sets the lag vehicle's acceleration with

$$h(s_t, \theta) = k_v(v_t^2 - v_t^1) + k_g(g_t - g_*) \tag{3.4}$$

according to a manually chosen θ. In this chapter we treat θ as unknown and our main focus is to estimate it. We assume that the parameters are nonnegative, hence $\theta \in \mathbb{R}_+^3$. We will use the notation $0_{m\times n}$ to denote the matrix of zeros with m rows and n columns. Given current state of the lag vehicle and controller parameters θ the next state is given by

$$s_{t+1}^1 = C s_t^1 + \begin{pmatrix} \Delta t^2/2 \\ \Delta t \end{pmatrix} h(s_t, \theta), \tag{3.5}$$

where

$$C = \begin{pmatrix} 1 & \Delta t \\ 0 & 1 \end{pmatrix}. \tag{3.6}$$

Given the lead vehicle's states we write the entire system dynamics as

$$\begin{pmatrix} s_t \\ 1 \end{pmatrix} = \begin{pmatrix} A(\theta) \\ 0_{2\times 5} \\ 0_{1\times 4} \; 1 \end{pmatrix} \begin{pmatrix} s_{t-1} \\ 1 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ s_t^2 \\ 0 \end{pmatrix}, \tag{3.7}$$

where

$$A(\theta) = \begin{pmatrix} C & 0_{2\times 3} \end{pmatrix} + \begin{pmatrix} \Delta t^2/2 \\ \Delta t \end{pmatrix} \begin{pmatrix} -k_g \\ -k_v \\ k_g \\ k_v \\ -k_g(l+g_*) \end{pmatrix}^{\mathsf{T}}. \tag{3.8}$$

3.3.3 Regularized Prediction

The difficulty in sampling trajectories in (3.3) stems from not knowing the lag vehicle's controller θ. Direct estimation of θ can assign significant probability to controllers that produce unrealistic behaviors. In this section we define regularization terms to promote more realistic behaviors. We can express the distribution of trajectories in (3.3) given the known hyperparameters γ, which we define later, as

$$\begin{aligned} &p(s^1_{k+1:T}|s_{1:k}, s^2_{k+1:T}, \gamma) \\ &= \int_{\mathbb{R}^3_+} p(s^1_{k+1:T}, \theta|s_{1:k}, s^2_{k+1:T}, \gamma) d\theta \\ &= \int_{\mathbb{R}^3_+} p(s^1_{k+1:T}|s_{1:k}, s^2_{k+1:T}, \theta, \gamma) p(\theta|s_{1:k}, s^2_{k+1:T}, \gamma) d\theta \\ &= \int_{\mathbb{R}^3_+} p(s^1_{k+1:T}|s_{1:k}, s^2_{k+1:T}, \theta, \gamma) p(\theta|s_{1:k}, \gamma) d\theta, \end{aligned} \tag{3.9}$$

where the last equality follows from the conditional independence expressed in Figure 3.2. We begin by defining the second term in (3.9) which can be written using Bayes' rule and calculating the accelerations derived from velocities as

$$p(\theta|s_{1:k}, \gamma) \propto p(\hat{a}_{1:k-1}, s_{2:k}|s_1, \theta, \gamma) p(\theta|s_1, \gamma). \tag{3.10}$$

For the first term in (3.10) we impose a recursive structure independent of γ to mirror standard Markov chain structure as

$$p(\hat{a}_{1:k-1}, s_{2:k}|s_1, \theta, \gamma) = \prod_{i=1}^{k-1} p(\hat{a}_i, s_{i+1}|s_i, \theta), \tag{3.11}$$

which combined with the system dynamics in (3.7) and the assumed Gaussian noise for acceleration yields

$$\begin{aligned} p(\hat{a}_{1:k-1}, s_{2:k}|s_1, \theta, \gamma) = \prod_{i=1}^{k-1} p(\hat{a}_i|h(s_i, \theta)) = \\ = \prod_{i=1}^{k-1} \mathcal{N}(\hat{a}_i; h(s_i, \theta), \sigma_a^2). \end{aligned} \tag{3.12}$$

Using this term only, we could estimate θ that fits the observed data. We now introduce the hyperparameters to address weaknesses of this initial model. One problem is an occasionally large and unrealistic estimate of the desired gap g_*. For this we regularize g_* to be close to a given mean gap g_0. Additionally, the car-following model was originally designed for dense traffic where the lead vehicle is near the lag vehicle. The model is thus vulnerable to overfitting the lag vehicle's behavior to a distant lead vehicle's accelerations. To address this, we regularize the proportional parameters to zero more as the distance between the lead and lag vehicles increases. We introduce the scalars α, β to control the precision of the normal priors placed on the desired gap and proportional parameters. Letting $\gamma = (\alpha, g_0, \beta)$, we define the second term in (3.10) as

$$-\log p(\theta|s_1, \gamma) = \alpha(g_* - g_0)^2 + \beta g_0^2(k_v^2 + k_g^2). \tag{3.13}$$

We now define the first term in (3.9) to regularize the future behavior of the controller, in contrast to (3.12) which focuses on the fit to observations. Let $\chi_{\{v \preceq 0\}}(v)$ denote the characteristic function which equals zero for the real vector v having all positive components and equals infinity elsewhere. The negative log-likelihood is defined to be

$$-\log p(s_{k+1:T}^1|s_{1:k}, s_{k+1:T}^2, \theta, \gamma) = \chi_{\{v \preceq 0\}}(v), \tag{3.14}$$

where $v = (v_{k+1}^1, ..., v_T^1)^\intercal$. Collecting the likelihoods specified in (3.12)-(3.14), the negative log-likelihood for a given θ in (3.9) is

$$\begin{aligned} f(\theta) = \frac{1}{2\sigma_a^2}\sum_{i=1}^{k-1}(\hat{a}_i - h(s_i, \theta))^2 + \alpha(g_* - g_0)^2 + \\ + \beta g_0^2(k_v^2 + k_g^2) + \chi_{\{v \preceq 0\}}(v). \end{aligned} \tag{3.15}$$

The σ_a^2 term contributes only to the weighting of the data fit term relative to the regularization terms. We thus may set $\sigma_a^2 = 1$ for convenience and determine the other hyperparameters relative to this. Given parameters θ' we can obtain a predicted trajectory via the dynamics given in equation (3.7). This trajectory has exact probability equal to

$$\frac{\exp\left(-f(\theta')\right)}{\int_{\mathbb{R}_+^3} \exp\left(-f(\theta)\right) d\theta} \tag{3.16}$$

under the model. Calculating this probability by numerical integration, however, is problematic due to the large number of function evaluations. Sampling can be used to obtain a consistent approximation, but we cannot sample θ directly from $\exp\left(-f(\theta)\right)$ because it does not correspond to a distribution for which efficient samplers exist. In the next section we construct a consistent and efficient sampler for the likelihood specified by (3.15).

3.3.4 Efficient Sampling

Our approach to sampling from (3.15) has two main steps. We first solve an optimization problem to find a set of parameters $\hat{\theta}$ that has high likelihood. We then employ importance sampling to sample from (3.15).

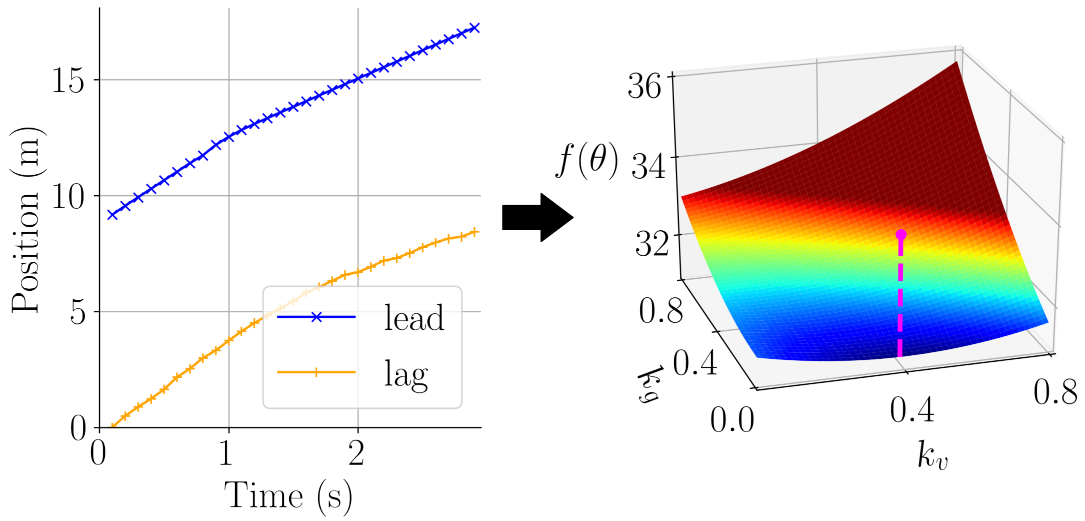


Figure 3.3: Likelihood surface of merging scenario. The projection $f(\theta)$ onto k_v and k_g appears smooth and unimodal along these parameters. The proposed inference procedure finds the global minimum (magenta) at $\hat{\theta} = (0.39, 0, 8.63)$.

Importance Sampling Given $\hat{\theta}$

We sample parameters via $\theta \sim q(\theta; \hat{\theta})$ in the higher likelihood region around $\hat{\theta}$ where q is a distribution chosen to have support over $\mathbb{R}^3_+$ and admit efficient samplers. These samples are then weighted with their importance weights

$$w = \frac{\exp\left(-f(\theta)\right)}{q(\theta; \hat{\theta})} \tag{3.17}$$

and normalized by the sum of the weights to complete the importance sampling. Since q has support over $\mathbb{R}^3_+$, the importance sampling is consistent, and depending on the choice of q this procedure may also be efficient for sampling all the high-likelihood θ. Figure 3.3 shows a typical likelihood surface of $f(\theta)$ using observations from NGSIM. The smooth surface and unimodality admit efficient sampling.

Optimizing to Find $\hat{\theta}$

There are multiple possible optimization problems that could be solved to find a high-likelihood $\hat{\theta}$ from $f(\theta)$. Minimizing $f(\theta)$ over $\theta \in \mathbb{R}^3_+$ directly is one choice but we see that $\forall t > k$ s_{t+1} depends on θ through both $A(\theta)$ and s_t in (3.7) since we only observe up to timestep k. This implies that including the future behavior regularizer (3.14) produces a nonconvex problem. To ensure our predictions can be made in realtime, we instead optimize over all other terms in $f(\theta)$. Optimizing over the chosen terms and noting $\theta = (\theta_1\ \theta_2\ \theta_3) = (k_v\ k_g\ g_*)$ yields

$$\hat{\theta} = \underset{\theta\in\mathbb{R}^3_+}{\operatorname{argmin}} \frac{1}{2}\sum_{i=1}^{k-1}(k_v(v_i^2 - v_i^1) + k_g(g_i - g_*) - \hat{a}_i)^2 + \\ + \alpha(g_* - g_0)^2 + \beta g_0^2(k_v^2 + k_g^2) \tag{3.18}$$

$$= \underset{\theta\in\mathbb{R}^3_+}{\operatorname{argmin}} \frac{1}{2}\sum_{i=1}^{k-1}(\theta_1(v_i^2 - v_i^1) + \theta_2(g_i - \theta_3) - \hat{a}_i)^2 + \\ + \alpha(\theta_3 - g_0)^2 + \beta g_0^2\theta_1^2 + \beta g_0^2\theta_2^2 \tag{3.19}$$

$$= \underset{\theta\in\mathbb{R}^3_+}{\operatorname{argmin}} \frac{1}{2}||D\begin{pmatrix}\theta_1\\ \theta_2\\ \theta_3\\ \theta_2\theta_3\end{pmatrix} - b||_2^2 \tag{3.20}$$

$$= \underset{\theta\in\mathbb{R}^3_+, u\in\mathbb{R}_+}{\operatorname{argmin}} \frac{1}{2}||D\begin{pmatrix}\theta\\ u\end{pmatrix} - b||_2^2 \text{ s.t. } \theta_2\theta_3 = u \tag{NC}$$

where we collect the variables and write the optimization as a least squares problem with the rewritten known terms being $D \in \mathbb{R}^{k+2,4}$ and $b \in \mathbb{R}^{k+2}$, whose rows correspond to the k–1 fitting terms and three regularizing terms. To remove the quadratic component in the objective function, in the last equality, we introduce a new decision variable u to represent this quadratic variable and then define a new equality constraint. From this point we first consider solving the problem (NC) where θ is no longer constrained to the nonnegative orthant:

$$\begin{aligned} &\underset{\substack{\theta\in\mathbb{R}^3\\ u\in\mathbb{R}}}{\text{minimize}}\ \frac{1}{2}||D\begin{pmatrix}\theta\\ u\end{pmatrix} - b||_2^2 \\ &\text{s.t. } \theta_2\theta_3 = u \end{aligned} \tag{NC1}$$

This problem admits a convex relaxation that exactly recovers the global minimum, shown next. We begin by defining $x = (\theta; u) \in \mathbb{R}^4$. To rewrite the nonconvex constraint $\theta_2\theta_3 = u$ in terms of x, we introduce $E \in \mathbb{S}^4$ and $c \in \mathbb{R}^4$ defined by

$$E = \frac{1}{2}\begin{pmatrix}0 & 0 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 0\end{pmatrix}, \tag{3.21}$$

and

$$c = \begin{pmatrix}0\\ 0\\ -1\\ 0\end{pmatrix}. \tag{3.22}$$

We now have

$$r(x) = x^\intercal E x + c^\intercal x = \theta_2\theta_3 - u, \tag{3.23}$$

whereby the constraint may be written as $r(x) = 0$. It is now possible to apply Lemma A.3 in Appendix A to show that solving the convex relaxation of (NC1) given by

$$\begin{aligned} \underset{\substack{X\in\mathbb{S}^4\\ x\in\mathbb{R}^4}}{\text{minimize}}\ & \frac{1}{2}\operatorname{tr}(D^\intercal D X) - b^\intercal D x + \frac{1}{2}b^\intercal b \\ \text{s.t. }\ & X \succeq xx^\intercal \\ & \operatorname{tr}(EX) + c^\intercal x = 0 \end{aligned} \tag{P}$$

for x is equivalent to solving (NC1) for x. We can observe that (P) is a semidefinite program by defining

$$Y = \begin{pmatrix} X & x \\ x^\intercal & 1 \end{pmatrix}, \tag{3.24}$$

and noting that by using the Schur complement of 1 in Y:

$$Y \succeq 0 \iff 1 > 0,\ X - xx^\intercal \succeq 0. \tag{3.25}$$

This shows that the $X \succeq xx^\intercal$ constraint in (P) is a semidefinite constraint. That (P) is a semidefinite program follows from the fact that the objective and remaining constraint are linear in Y.

At this point, solving (P) exactly recovers the global minimum of (NC1). Provided the solution to (P) happens to satisfy $x \succeq 0$, the solution for (NC1) is also feasible for (NC). Since (NC1) and (NC) share the same convex objective function, the solution's feasibility implies it is also optimal for (NC). Moreover, the same solution is the global minimum of the entire negative log-likelihood (3.15) whenever it additionally satisfies the nonconvex constraint on future velocities (3.14). Since there is the possibility that the solution to (P) will not satisfy $x \succeq 0$, we instead approximate (NC) with the additional nonnegative constraint on (P) given by:

$$\begin{aligned} \underset{\substack{X\in\mathbb{S}^4\\ x\in\mathbb{R}^4}}{\text{minimize}}\ & \frac{1}{2}\operatorname{tr}(D^\intercal D X) - b^\intercal D x + \frac{1}{2}b^\intercal b \\ \text{s.t. }\ & X \succeq xx^\intercal \\ & \operatorname{tr}(EX) + c^\intercal x = 0 \\ & x \succeq 0 \end{aligned} \tag{P1}$$

The proposed method for sampling trajectories is summarized in Algorithm 2. Lemma A.3 also tells us that we need sufficiently many observations to ensure D is full rank. This occurs at a minimum of two distinct observations, and in practice we find four to be sufficient.

Algorithm 2: Probabilistic Trajectory Prediction for Ramp Merging

```
  Input: $s_{1:k}$, $s^2_{k+1:T}$, $\gamma$, $n$
  Output: $s^{1,(i)}_{k+1:T}, p^{(i)}$, for $i = 1, \ldots, n$
1 Solve convex problem (P1) for $\hat{\theta}$
2 foreach $i = 1, ..., n$ do
3     Sample $\theta_i \sim q(\theta; \hat{\theta})$
4     Generate $s^{1,(i)}_{k+1:T}$ via (3.7)
5     $w_i \leftarrow \exp(-f(\theta_i, s_{1:k}, s^{1,(i)}_{k+1:T}, s^2_{k+1:T}, \gamma))/q(\theta_i; \hat{\theta})$ via (3.15)
6 end
7 $\forall i = 1, \ldots, n \; p^{(i)} \leftarrow w_i / \sum_{i=1}^{n} w_i$
```

3.4 Experiments

To evaluate the proposed method's ability to predict trajectories in dense traffic for ramp merging, we test it on the NGSIM dataset [80]. The NGSIM dataset includes full trajectory data recorded at 10 Hz for two highways, I-80 and US-101, during peak usage. Since our focus is ramp merging for AVs, we extract relevant pairs of lead and lag vehicles. These pairs are those between which a vehicle entering the highway has merged, or the pair behind such a pair. We are most interested in predicting the behavior of the lag vehicle at the most crucial moment–when it can see the potentially merging ego vehicle. For each pair the start of the prediction window $t = k + 1$ begins when the merging vehicle first passes the lag vehicle. The end of the prediction window $t = T$ occurs either when the ego vehicle passes the lead vehicle or first enters the target lane in case of a merge. All pairs are observed for 3.2 s before the prediction window. This choice of observation window allows us to compare to methods that use more traditional window lengths. We extract 420 pairs from the I-80 data and 292 pairs from the US-101 data.

3.4.1 Model Specifications

We set g_0 equal to the mean of the observed gaps. For the precision values, we found that values in $[0.5, 2]$ achieve a good balance between performance and probability calibration. Following this, we set α and β to 1 for all experiments. For importance sampling we define $q(\theta; \hat{\theta})$ as the multivariate normal distribution $\mathcal{N}(\hat{\theta}, I_3)$ truncated to $\mathbb{R}^3_+$ and draw 1,000 samples. This variance was found to be sufficiently large to sample effectively. The convex problem (P1) is solved with CVXOPT v1.2.3 [93], an open-source solver for convex optimization.

3.4.2 Baselines

We compare to state-of-the-art methods for ramp merging and general highway prediction in addition to a simplified version of the proposed model:

- **Constant Velocity (CV):** The average velocity is used to predict future positions.
- **IDM-based (IDM)**[91] **:** The IDM car-following model [92] is parameterized based on the identified lead vehicle. Unlike other methods it uses the future trajectories of the lead vehicle and the ego vehicle.
- **Social GAN (SGAN)**[1] **:** Shown to achieve state-of-the-art performance on NGSIM when compared to other neural networks [77] despite originally being designed for joint prediction of pedestrian trajectories.
- **Multi-Agent Tensor Fusion (MATF)**[9] **:** Achieves state-of-the-art performance on NGSIM using a global pooling layer to capture distant interactions while maintaining spatial structure.
- **No Regularization (Proposed-NR):** The proposed method without regularization terms, corresponding to only the first term of (3.15).

Each DNN is trained once on each highway dataset. For making predictions on a given scenario, the model that has not seen it during training is used to make predictions. To evaluate these probabilistic predictions from SGAN and MATF we draw 100 samples.

The proposed method uses observations for the lead and lag vehicles, but SGAN and MATF have traditionally been evaluated with observations for all vehicles on the freeway [77, 9]. For comparison we include this standard evaluation, denoted by SGAN* and MATF*. In practice, however, we will accurately detect only nearby vehicles. To reflect this case, we evaluate SGAN and MATF with the same observations as the proposed method, augmented with observations of other vehicles we may reasonably detect from the viewpoint of the ego vehicle. We add observations for three additional vehicles: the lead vehicle's lead, the ego vehicle, and the ego's own lead vehicle.

3.4.3 Evaluation Metrics

Let $\hat{x}_{i,t}$ be the random variable corresponding to the probabilistic prediction of the lag vehicle's longitudinal position at timestep t in the ith scenario. The true position is denoted $x_{i,t}$. Since the time horizon varies between scenarios, we denote

Table 3.1: Predictive performance of each method for the NGSIM dataset (best in **bold** and second best underlined). Average distance error (ADE) and root mean squared error (RMSE) are shown as ADE/RMSE in meters. The proposed method achieves the lowest error for short-term predictions, and outperforms the DNNs when the observations are limited to nearby vehicles.

	3.2 s observed							
	Extra information							
t (s)	IDM	SGAN*	MATF*	CV	SGAN	MATF	Proposed-NR	Proposed
0.8	0.65/0.89	0.46/0.75	0.44/0.68	0.67/0.92	0.67/0.96	0.49/0.75	0.37/**0.60**	**0.33**/**0.60**
1.6	1.95/2.60	1.10/1.63	1.00/**1.44**	1.47/1.97	1.47/2.01	1.20/1.73	1.08/1.56	**0.95**/1.62
2.4	3.47/4.73	1.87/2.60	**1.56**/**2.17**	2.34/3.42	2.34/3.13	2.08/2.93	1.99/2.77	1.67/2.47
3.2	4.73/6.20	2.78/3.71	**2.04**/**2.81**	3.42/4.44	3.35/4.39	3.01/4.24	3.01/4.14	2.54/3.61
4.0	5.57/7.29	3.81/4.99	**2.67**/**3.60**	4.63/5.91	4.46/5.77	4.19/5.87	4.25/5.74	3.54/4.88
4.8	5.97/7.72	4.90/6.26	**3.22**/**4.34**	5.94/7.60	5.65/7.27	5.42/7.52	5.63/7.55	4.67/6.31

N_t as the number of scenarios with time horizon $T \geq t$. To evaluate the accuracy of the probabilistic trajectory predictions we evaluate the following metrics:

- *Average Distance Error (ADE):* The expected distance between the prediction and the true position, used in [1, 77, 9, 78]. ADE is calculated at timestep t as:

$$ADE(t) = \frac{1}{N_t} \sum_{i=1}^{N_t} \mathbb{E}[|x_{i,t} - \hat{x}_{i,t}|]$$

- *Root Mean Squared Error (RMSE):* The square root of expected squared error between the prediction and the true position, used in [76, 75, 74, 77, 9]:

$$RMSE(t) = \sqrt{\frac{1}{N_t} \sum_{i=1}^{N_t} \mathbb{E}[(x_{i,t} - \hat{x}_{i,t})^2]}$$

The ADE tells us how prediction errors are distributed on average. Predicting a distribution over positions that has a mean close to the actual position will result in lower error. The RMSE is similar but assigns more weight to larger errors due to the squared term within the expectation. A method with ADE lower than RMSE suggests that it predicts more extreme cases, but assigns lower probability to these.

3.4.4 Performance in Dense Traffic Scenarios

The performance for each method is shown in Table 3.1. The model-based method IDM makes overly conservative predictions about braking which hurt its performance. MATF* achieves the lowest errors for long-term predictions by utilizing positional information of all vehicles on the freeway. SGAN* also achieves low error with the same information. Using the same DNNs to make predictions having observed only the more immediate vehicles, however, results in worse performance. The proposed

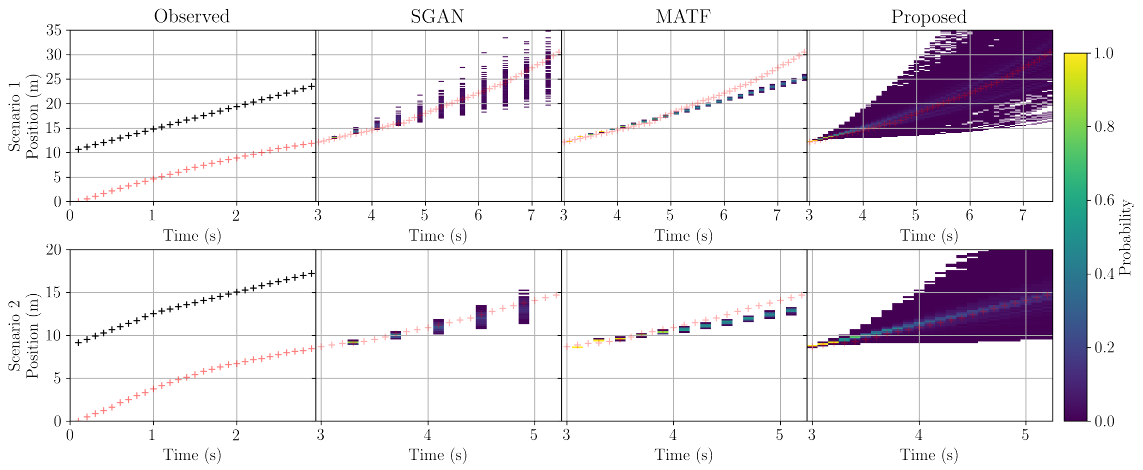


Figure 3.4: Predictions of each method on two ramp merging scenarios. The red crosses show the lag vehicle's position and the black crosses show the observed positions of its lead vehicle. The probabilistic predictions for each method are displayed after the end of the observation window. The color bar (right) provides the probability corresponding to each color. The proposed method predicts the lag vehicle's positions accurately despite using less information than the other interaction-based methods.

method is evaluated without knowledge of the lead's own lead vehicle, or the vehicles in the merge lane, yet still outperforms the DNNs. The drop in DNN performance suggests that rather than learning to predict based on cues in nearby drivers' behavior, they have learned how traffic waves propagate along highways. Comparing the errors made in the short-term and the long-term, we observe that while the DNNs perform competitively with CV in the short-term, they are better at capturing behavior in the long-term. Fitting the car-following model directly as Proposed-NR is competitive with MATF, though less so at longer-term predictions. Adding the regularization terms enables the proposed method to outperform the baselines, excepting those with full observations of the freeway. Even compared to these methods, the proposed method predicts the short-term with lower error. This makes sense if MATF* learned to focus on the long-term rather than the short-term, relying heavily on observations of vehicles much farther ahead.

3.4.5 Performance with Limited Observations

Previously we compared performance ensuring that each method had sufficiently many observations in each scenario. For the real scenarios that AVs will encounter, however, we cannot guarantee that such time will be available. Even within the NGSIM dataset alone, nearly 8% of the scenarios having between 400 ms and 3.2 s of observations are removed from our evaluation to ensure traditional observation windows may be used. In this section we make predictions on the same scenarios as before but limit ourselves to 400 ms of observations. Both SGAN and MATF operate on downsampled data. SGAN operates at 2.5 Hz, so for the limited observation window of 400 ms it sees only one observation. To supply the method with the traditional 3.2 s of observations, we extrapolate using the constant velocity model from the 400 ms of original observations at 10 Hz. We also extrapolate for MATF which operates at 5 Hz.

The ADE and RMSE for a subset of the methods is shown in Table 3.2. We see that the performance for both SGAN and MATF remains largely unchanged between using the traditional and limited observation windows. This has also been observed in pedestrian trajectory prediction [94] where only the first few observations were found to significantly affect predictive performance of a DNN. The proposed method's performance decreases for the longer-term predictions, but still outperforms the baselines.

Table 3.2: ADE/RMSE with limited observations. CV predicts competitively in the short-term while MATF does so in the long-term. The proposed method retains its performance despite using a short observation window.

	400 ms observed			
t (s)	CV	SGAN	MATF	Proposed
0.8	0.43/0.78	0.65/0.95	0.51/0.82	**0.32/0.59**
1.6	1.13/1.81	1.46/2.04	1.23/1.84	**0.92/1.48**
2.4	2.01/2.95	2.35/3.20	2.04/2.92	**1.68/2.59**
3.2	3.13/4.38	3.34/4.45	2.97/4.19	**2.63/3.88**
4.0	4.43/6.05	4.40/5.75	4.09/5.74	**3.69/5.27**
4.8	5.89/7.89	5.60/7.26	5.15/7.24	**4.87/6.82**

Table 3.3: Compute time and probability calibration. The proposed method with 1000 samples and CV predict in realtime, while SGAN and MATF with 100 samples do not. SGAN and the proposed method show calibrated probability estimates, while MATF and CV match in calibration.

	CV	SGAN	MATF	Proposed
Compute time (s)	0.002	0.549	0.908	0.028
Calibration	0.65	0.18	0.65	0.17

3.4.6 Speed

Sudden and critical scenarios in autonomous driving demand methods that operate in realtime. Table 3.3 shows the time taken to make predictions for a single scenario. SGAN and MATF are benchmarked on GTX 1080 GPU, and the other methods on Intel Core i7-6800K CPU at 3.40 GHz. Although SGAN and MATF do not make realtime predictions with 100 samples, they could do so by reducing the number of samples. This would improve speed at the cost of less accurate probability estimates. The proposed method's low number of parameters aids in making realtime predictions despite using 1000 samples.

3.4.7 Probability Calibration

Probabilistic predictions attach a probability to each predicted outcome and enable planners to calculate risk. The calculated risk, however, depends on estimates of probability since the true distribution over future outcomes is unknown. The constant velocity model can be viewed as a probabilistic method with degenerate probability estimates. It predicts a single outcome and assigns it full probability. In some sense these probability estimates are not accurate, because vehicles often take different trajectories. We measure this accuracy with a metric for the calibration of regression methods [95]. This measures the squared error between each confidence interval's probability and the empirical probability of outcomes within the interval being realized. The calibration scores for each method in Table 3.3 mirror the predictive distributions in Figure 3.4. SGAN and the proposed method have calibrated probability estimates, while MATF tends to underestimate the uncertainty in its predictions.

3.5 Conclusion

We propose a novel probabilistic extension for a car-following model and introduce regularization terms to enforce realism in predicted behaviors. Through experiments we demonstrate that these terms lead to increased prediction accuracy for real ramp merging scenarios in dense traffic. Comparing our model to existing methods on the NGSIM dataset shows that it achieves state-of-the-art performance. Furthermore, the proposed model maintains comparable performance when limited to using very few observations. There are multiple limitations to the proposed model that provide grounds for future work. The model considers interactions only between the lag and lead vehicles. Combining this with approaches that consider interactions between the lag and ego vehicles [81, 90, 91] provides one direction for future work. Accounting for lane changes provides another direction.

CHAPTER IV

Pedestrian Prediction in Shared Spaces for Autonomous Vehicles

4.1 Introduction

Pedestrians and drivers interact closely in a wide range of environments. Various road markings and signals regulate their interactions, and these features have been leveraged by many model-based prediction methods to better predict pedestrians. Environments such as shared spaces, however, aim to regulate traffic through natural social interactions rather than traffic devices. Shared spaces are specifically designed to minimize separation between pedestrians and drivers to promote negotiation between the two groups of road users [96]. This focus on social interactions limits the applicability of prediction methods relying on the existence of traditional traffic devices. Recent model-free prediction methods have instead focused on accurately predicting pedestrians in arbitrary environments. Deep neural networks (DNNs) have proven especially effective at leveraging large datasets to learn the various interactions amongst pedestrians, between pedestrians and the environment, and between pedestrians and vehicles. This superior performance and generality comes at a price. Black-box methods sacrifice both interpretability and speed with ever larger numbers of parameters. In this chapter we aim to strike a balance between existing model-based and model-free methods, borrowing techniques from each. We introduce a probabilistic method called Off the Sidewalk (OTS) to predict pedestrian trajectories in environments where sidewalks and other traffic devices may or may not be present. As in model-free methods we leverage existing trajectory data to learn functions for pedestrian attention and vehicle influence. At the same time, we focus on modeling only interactions between pedestrians and vehicles. While this focus ignores interactions amongst pedestrians, we find that modeling this type of interaction alone enables the proposed method to achieve state-of-the-art performance. The simplified modeling of interactions also yields a model that is more interpretable and faster than state-of-the-art DNNs. The main contributions of this chapter are:

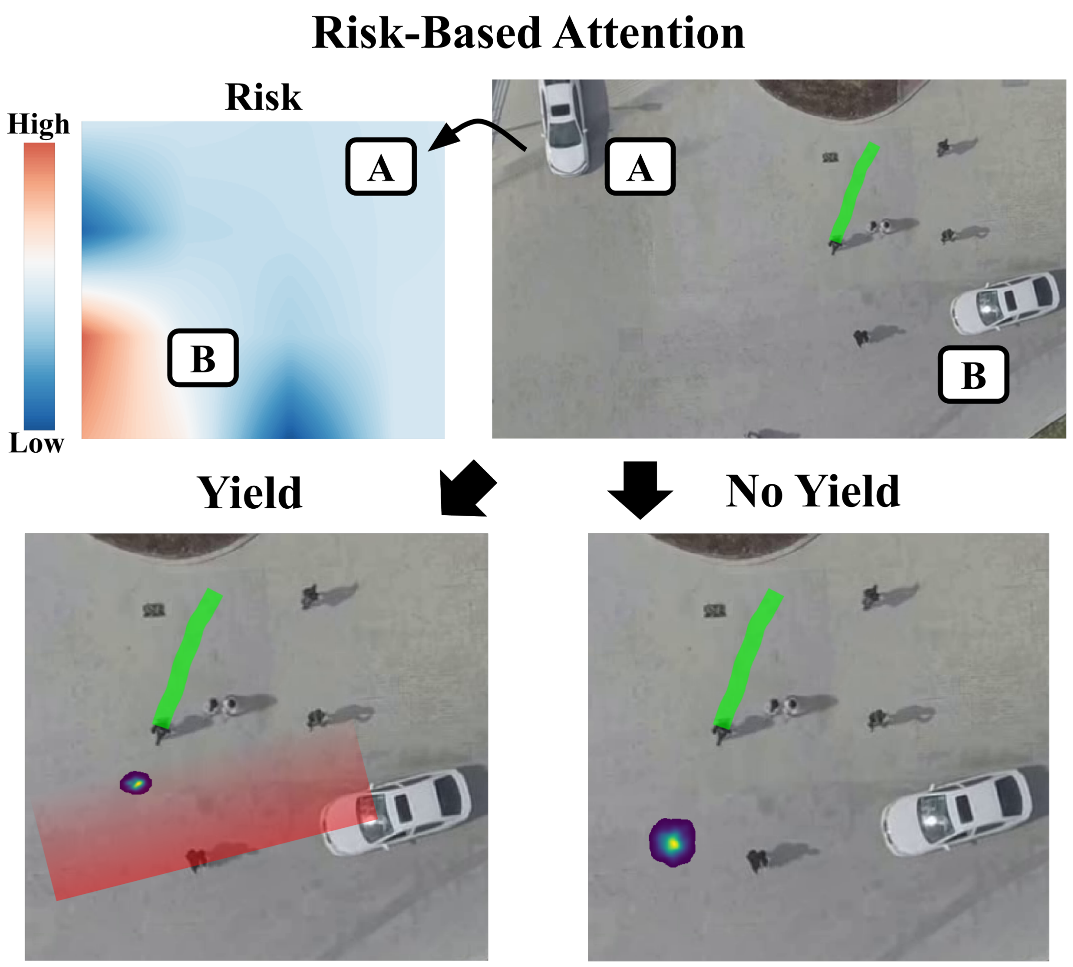


Figure 4.1: Steps of the proposed interaction model. The pedestrian pays attention to each vehicle and yields in proportion to estimated risk. A learned vehicle influence then predicts how yielding pedestrians adjust their speed, while non-yielding pedestrians continue at their desired velocity. The predicted distribution over future positions is shown for each case.

1. a novel and realtime probabilistic method OTS to predict pedestrian trajectories in scenes where traditional traffic devices may not be present;
2. a tractable training procedure that avoids the auxiliary simulations or manually specified parameters called for in previous model-based works;
3. evaluation on real-world interactions at shared spaces and urban intersections in the DUT [30] and inD [97] datasets.

The proposed method OTS predicts individual pedestrians in two steps, shown in Figure 4.1. Risk-based attention is used to predict which vehicle holds the pedestrian's attention, and whether the pedestrian yields to the chosen vehicle. To model risk in the absence of informative features such as curbs, we rely entirely on the pedestrian's position and velocity relative to the vehicle. For yielding pedestrians, a learned vehicle influence predicts how yielding adjusts the pedestrian's speed. Similar to social forces [33], vehicle influence is based on the pedestrian's distance to the vehicle's anticipated motion, but is learned from labeled data. Since labels for attention and yielding are typically unavailable, the resulting training problem may have many modes. We employ pseudo-likelihood methods to decompose the problem into simpler parts that are readily solved.

The chapter is organized as follows. Section 4.2 describes related methods for predicting pedestrians' trajectories. Section 4.3 describes the model of interactions between pedestrians and vehicles used to predict trajectories. In Section 4.4 we evaluate the model on the DUT and inD datasets, concluding in Section 4.5.

4.2 Related Work

We first describe methods that predict trajectories by explicitly modeling pedestrians' interactions with other road users or the environment. Methods that learn models of interaction directly from large datasets are described in the next section.

4.2.1 Model-Based Methods

Recent works have had success with modeling the evolution of the pedestrian's position as a Markov process [98, 99, 28, 27]. Methods based on solving Markov Decision Processes [31, 32] and non-Markovian models such as Interacting Gaussian Processes [22] have also been proposed. The former, however, do not scale to account for interactions between road users, and the latter have not achieved the same performance as more recent methods [40]. We adopt the Markov process approach in this chapter. In these approaches the pedestrian at each timestep chooses whether to continue a nominal trajectory or stop for an oncoming vehicle. Previous works have modeled this decision at signaled intersections [98] and marked crosswalks [27, 28]. In these settings they have leveraged scene features to estimate the pedestrian's risk associated with continuing. More general scenes containing at least curbs have been examined in Kooiji et al. [99], but this chapter addresses pedestrian motion only in one dimension. Vehicle interactions are incorporated by measuring risk presented by the oncoming vehicle. Measures include vehicle speed and distance [28], and minimum separation distance [99, 27]. Blaiotta [27] additionally considers the time remaining before the minimum distance is attained. The focus of this chapter is on shared scenes, which lack the informative features provided by traditional road infrastructure. Here, risk depends only on the minimum distance and time features.

Once the decision to yield is made, many works model the pedestrian's speed as a binary option of stopping or walking [99, 27, 28]. We propose to learn a vehicle influence function that specifies how pedestrians adjust their current speed when yielding, rather than stopping. The learned influence shown in Figure 4.3 (right) captures the phenomenon that many pedestrians slow down before stopping. This is crucial to detecting the intent to yield early. Similar to social forces [33, 100], the influence depends on the pedestrian's distance to the vehicle, but our formulation enables a much simpler estimation using linear least-squares.

4.2.2 Deep Learning Methods

In contrast to traditional methods that rely heavily on manually chosen features, model-free methods learn features directly from large labeled datasets. This automatic feature selection has contributed to the recent successes of deep learning methods [79, 57, 55]. Unlike most model-based methods which estimate uncertainty, these initial works make only deterministic predictions. Subsequent works have addressed this by predicting the parameters of the normal distribution [40, 77]. These works also use social pooling layers, which extract features for nearby pedestrians [40] or road users [77] based on a grid of specified size. The fixed size of the grid, however, could fail to account for distant interactions. Many methods have addressed this by replacing social pooling with soft attention, which models each pairwise interaction between road users [53, 56, 101, 78]. The risk-based attention used in this chapter is similar to soft attention. Of the above DNNs using soft attention, only TrafficPredict [78] models interactions between pedestrians and vehicles.

Speed has been an area of focus for these works since the number of pairwise interactions computed for soft attention quickly grows with the number of road users. Social pooling also entails a costly pooling step for each road user. Social GAN [1] introduces permutation invariance to replace these slower operators. This method reduces the computational burden to a single application of the proposed permutation invariant pooling module. One drawback, however, is that the permutation invariant operators do not preserve the uniqueness of interaction features for each road user. Multi-Agent Tensor Fusion (MATF) [9] addresses this by introducing a global pooling layer that preserves uniqueness. Though not as efficient as Social GAN, MATF achieves state-of-the-art performance without the computational burden of soft attention.

4.3 Interaction-Based Trajectory Predictions

We formulate the problem of predicting pedestrian trajectories in Section 4.3.1. Section 4.3.2 introduces the probabilistic method, Off the Sidewalk (OTS), used to model pedestrians' interactions with vehicles and predict pedestrian trajectories.

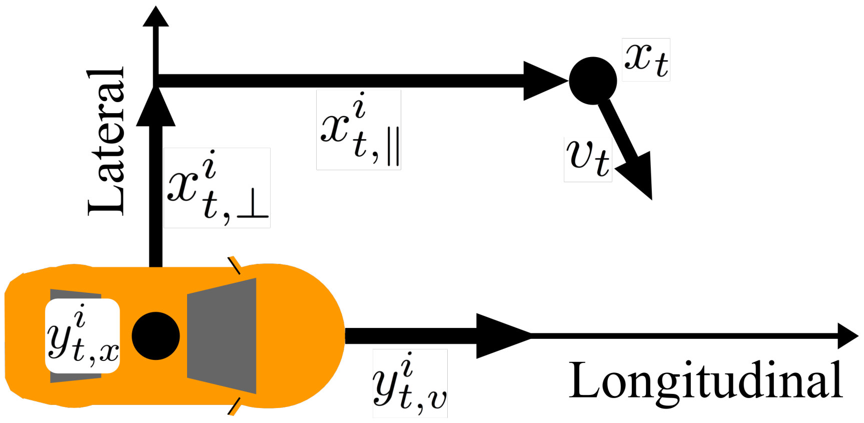


Figure 4.2: Reference frame for the ith vehicle at timestep t. The pedestrian's position in this frame is decomposed into the orthogonal components $x^i_{t,\perp}$ and $x^i_{t,\|}$. The positions and velocities used in the world frame are shown for reference.

Estimation of the model parameters is described in Section 4.3.3, followed by implementation details in Section 4.3.4.

4.3.1 Problem Statement

We receive noisy observations of pedestrian position and aim to predict the true position at future timesteps. We denote the noisy observation at timestep t by $\hat{x}_t \in \mathbb{R}^2$ and the corresponding true position by $x_t \in \mathbb{R}^2$. Given observations over timesteps $t = 1, \ldots, k$ and a final timestep of T, we write the prediction task as sampling future trajectories

$$\{x_t\}^T_{t=k+1} \sim p(\{x_t\}^T_{t=k+1} | \{\hat{x}_t\}^k_{t=1}). \tag{4.1}$$

In this chapter we focus on modeling interactions of a single pedestrian with multiple vehicles. Like other model-based works [99, 27, 28], we assume vehicle position and velocity for each timestep is known and deterministic. While this assumption is not true when pedestrians and drivers continuously respond to each others' actions, it does hold in a scenario of significance to AVs. In particular, the assumption holds when the AV is planning its own future trajectory, with no intent of aborting the execution before the final timestep of prediction. In this case the AV knows its own trajectory and does not modify it in response to the pedestrian's actions. We examine this scenario in Section 4.4.4.

4.3.2 Pedestrian-Vehicle Interaction Model

Let $x_t, v_t \in \mathbb{R}^2$ denote the pedestrian's position and desired velocity at timestep t. We now turn to defining the variables used to model the pedestrian's interaction with vehicles. The first step is to choose the pedestrian's vehicle of focus. Let $r_t \in$

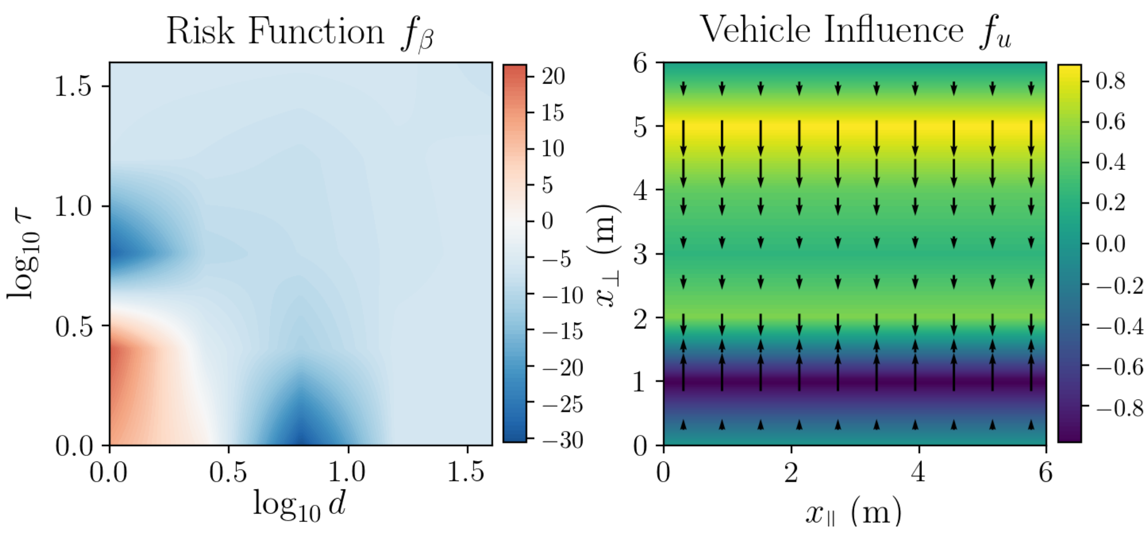


Figure 4.3: Learned functions for the DUT dataset. The learned risk function (left) predicts the decision boundary for pedestrians' yielding to lie along the white contour, over low values of minimum distance d and time remaining τ. The learned vehicle influence (right) resembles a curb roughly 3 m away from the vehicle. Arrows show the movement of a yielding pedestrian with desired velocity of 1 m/s.

$\{1 \dots n_v\}$ denote which of the n_v vehicles currently has the pedestrian's attention. The next step is whether or not the pedestrian yields to vehicle r_t. We define the binary variable $q_t = 0$ for yielding, and $q_t = 1$ for continuing at the desired velocity v_t. To aid in defining the extent of interactions, we introduce $R_t \subseteq \{1, \dots, n_v\}$, the set of vehicles the pedestrian may pay attention to at timestep t. Let the current position and velocity of the ith vehicle be given by $y^i_{t,x}, y^i_{t,v} \in \mathbb{R}^2$, respectively. Also let $x^i_{t,\perp}$ and $x^i_{t,\parallel}$ denote the lateral and longitudinal components of the pedestrian's position in the ith vehicle's reference frame. This is shown in Figure 4.2. We define a maximum lateral distance $u_{\max}$ to limit the extent of interactions. Any vehicles beyond this distance are ignored by the pedestrian. Additionally ignoring vehicles behind the pedestrian or not crossing the pedestrian's path, we define

$$R_t = \{i \in \{1, \dots, n_v\} |\ x^i_{t,\parallel} \geq -l, \quad |x^i_{t,\perp}| \leq u_{\max},\ v_t^\intercal z < 0\}, \tag{4.2}$$

where l corresponds to half the vehicle length and z corresponds to the unit vector for the lateral axis in Figure 4.2. The positions for which yielding may occur correspond to a subset of a quadrant in front of the vehicle, as in Figure 4.1. We now define vehicle influence over these positions. Vehicle influence is modeled as a piecewise-linear function of $x^i_{t,\perp}$ that is symmetric about zero, as $f_u : \mathbb{R} \to \mathbb{R}$. The function is linear in $u \in \mathbb{R}^{n_u}$, with n_u denoting the number of grid points. The function grid is parameterized by its maximum distance $u_{\max}$ with the n_u grid points evenly spaced within $[0, u_{\max}]$. Given a pedestrian yielding to vehicle i, the pedestrian's velocity is defined

$$v_{\text{yield}} = f_u(x^i_{t,\perp})v_t. \tag{4.3}$$

Similar to Social Forces [33] each position specifies a yielding velocity for the pedestrian. Shown in Figure 4.3, the model has learned that yielding pedestrians slow down before stopping closer to the vehicle's path. We may now write the pedestrian's next transition as

$$x_{t+1} = x_t + [q_t v_t + (1 - q_t) f_u(x^{r_t}_{t,\perp}) v_t]\Delta t, \tag{4.4}$$

where Δt is the size of each timestep.

Now defining the distributions for each variable, we first assume normally distributed noise for the observations as

$$\hat{x}_t \sim N(x_t, \sigma_x^2), \tag{4.5}$$

with variance σ_x^2. Desired velocity is modeled as a driftless random walk with normally distributed innovations. Its transition is given by

$$v_{t+1} \sim N(v_t, \sigma_v^2), \tag{4.6}$$

where σ_v^2 is the variance of the innovations. The pedestrian decisions for r_t and q_t depend on risk features, which we define next. Under a constant velocity, the remaining time before the pedestrian and vehicle $i \in R_t$ reach their minimum separation distance is given by

$$\tau^i_t = \frac{(x_t - y^i_{t,x})^\intercal (y^i_{t,v} - v_t)}{\|y^i_{t,v} - v_t\|_2^2}. \tag{4.7}$$

Since $i \in R_t$, the ith vehicle is closing the distance to the pedestrian and this time is positive and finite. The minimum distance itself is given by

$$d^i_t = (\|y^i_{t,x} - x_t\|_2^2 - (\tau^i_t)^2 \|y^i_{t,v} - v_t\|_2^2)^{\frac{1}{2}}. \tag{4.8}$$

When both the remaining time τ^i_t and minimum separation distance d^i_t are low, we would expect the perceived risk to be high. On the other hand, a high value for either would suggest low risk. We aim to learn this relationship from data with a piecewise-linear function similar to the vehicle influence. The function is defined on a regular grid over $[b_0, b_1]^2 \subseteq \mathbb{R}^2$ with n_b^2 evenly spaced points. Denote the piecewise function $f_\beta : \mathbb{R}^2 \to \mathbb{R}$, which is linear in the model parameter $\beta \in \mathbb{R}^{n_b^2+1}$ and includes a constant term. We define the current risk perceived by the pedestrian as

$$\text{risk}^i_t = f_\beta(\log_{10} \tau^i_{t-1}, \log_{10} d^i_{t-1}). \tag{4.9}$$

The arguments to f_β are in the log scale, to match the intuition that risk changes more rapidly nearer to collisions. Figure 4.3 shows that this is reflected in the learned parameters. Having defined risk, we now define the distributions for r_t and q_t. First, when there are no vehicles presenting risk, $R_t = \emptyset$. For this case we take $q_t = 1$ since no yielding will occur. When there are possibly multiple vehicles, the pedestrian pays attention in proportion to risk. For $i \in R_t$ we define the distribution of r_t with the softmax function as

$$p(r_t = i | x_{t-1}, v_{t-1}, \beta) = \frac{\exp \text{risk}_t^i}{\sum_{j \in R_t} \exp \text{risk}_t^j}. \tag{4.10}$$

Given vehicle r_t has the pedestrian's attention, the binary decision to yield is distributed as

$$p(q_t = 0 | x_{t-1}, v_{t-1}, r_t, \beta) = \frac{\exp \text{risk}_t^{r_t}}{1 + \exp \text{risk}_t^{r_t}}. \tag{4.11}$$

Compared to choosing a vehicle based on relative risk, the decision to yield is based on absolute risk. We place weak Gaussian priors on the parameters u and β to ensure their estimation is well-posed. Let chosen scalars α_u and α_β denote the strength of these priors. The negative log likelihoods are given by

$$\begin{aligned} -\log p(u) &= \alpha_u \|u\|_2^2 \\ -\log p(\beta) &= \alpha_\beta \|\beta\|_2^2 \end{aligned} \tag{4.12}$$

which correspond to zero-mean Gaussian priors, with precision proportional to α_u and α_β. Additionally we restrict the domain of each element of u to the interval $[-1, 1]$. This allows for the interpretation that pedestrians only decrease speed in response to vehicle influence. Although pedestrians may temporarily increase speed while crossing in front of a fast moving vehicle, we approximate this by a lack of yielding rather than with the vehicle influence. The next section describes how to estimate the model's unknown parameters σ_v^2, u, and β.

4.3.3 Model Estimation

We optimize a likelihood function to estimate model parameters. To keep quantities concise, we introduce additional notation. For each of the variables x_t, v_t, r_t, q_t, let its bolded version denote the entire time series, such as $\mathbf{x} \equiv \{x_t\}_{t=1}^{t_f}$, where t_f is the final timestep observed. Additionally, let $s_t = (x_t, v_t)$. Using the Markov structure of the model, we write the joint distribution of a single pedestrian's data

$$
\begin{aligned}
&L_{\text{full}}(\mathbf{x}, \mathbf{v}, \mathbf{r}, \mathbf{q}, \sigma_v^2, u, \beta) = \\
&= p(u)p(\beta) \prod_{t=2}^{t_f} p(\hat{x}_t | s_{t-1}, r_t, q_t, u) p(q_t | s_{t-1}, r_t, \beta) \\
&\qquad\qquad\qquad\qquad p(r_t | s_{t-1}, \beta) p(v_t | v_{t-1}, \sigma_v^2)
\end{aligned} \tag{4.13}
$$

From the transition equation (4.4) there are many interacting terms. The decision variables r_t and q_t are also discrete. These features suggest that the full likelihood L_{full} may have many modes that can trap optimization procedures at poor local optima. We instead work to separate this likelihood into commonly solved problems. This is accomplished by first removing the need to estimate each r_t. We first note that the set of possible vehicles R_t effectively specifies r_t when it is either empty or consists of a single vehicle. Since R_t depends on the pedestrian's position and desired velocity, we use the observed positions and a moving average of observed velocities over two seconds in their place to produce the estimate $\hat{R}_t$. We remove the likelihood's dependency on $\mathbf{r}$ by ignoring data for all pedestrians having any timestep t with $|\hat{R}_t| > 1$. This pseudo-likelihood technique of ignoring component likelihoods will reduce the efficiency of our parameter estimates [102]. Using a large dataset to estimate the parameters, however, allows us to safely ignore this loss. As each remaining r_t is specified by $\hat{R}_t$, we remove the $p(r_t | x_{t-1}, v_{t-1}, \beta)$ term which no longer contributes any information. Defining the set of timesteps with no candidate vehicles as $Q = \{t | \hat{R}_t = \emptyset\}$, we also have that $\forall t \in Q, q_t = 1$. Examining the transition equation (4.4), we can simplify likelihood terms for $t \in Q$ as

$$
p(\hat{x}_t | x_{t-1}, v_{t-1}, r_t, q_t = 1, u) = p(\hat{x}_t | x_{t-1}, v_{t-1}). \tag{4.14}
$$

Rewriting the joint distribution of the included pedestrian in terms of Q now yields

$$
\begin{aligned}
&L_Q(\mathbf{x}, \mathbf{v}, \mathbf{q}, \sigma_v^2, u, \beta) = \\
&= \prod_{t=2}^{t_f} p(v_t | v_{t-1}, \sigma_v^2) \prod_{t \in Q} p(\hat{x}_t | s_{t-1}) \\
&p(u)p(\beta) \prod_{t \notin Q} p(\hat{x}_t | s_{t-1}, r_t, q_t, u) p(q_t | s_{t-1}, r_t, \beta)
\end{aligned} \tag{4.15}
$$

The likelihood given by (4.15) has eliminated the dependency on u and β for the first two products. In fact, the first two products form a Kalman smoothing problem. Given noisy observations $\hat{x}_t$ for $t \in Q$, we estimate the true position x_t for $t \in Q$, v_t for all timesteps, and the variance σ_v^2. Using $\hat{x}_t$ as an unbiased estimate of x_t for $t \notin Q$, we are now in a position to use the remaining likelihoods. We use these estimates in place of their unknown values to estimate the model parameters u and β. Denoting $\{q_t\}_{t \notin Q}$ by $\mathbf{q_c}$, the final likelihood we use is given by

$$\begin{aligned} &L_c(\mathbf{q_c}, u, \beta) = \\ &p(u)p(\beta) \prod_{t \notin Q} p(\hat{x}_t | s_{t-1}, r_t, q_t, u) p(q_t | s_{t-1}, r_t, \beta) \end{aligned} \tag{4.16}$$

Taking the negative log likelihood of (4.16) yields

$$\begin{aligned} &l_c(\mathbf{q_c}, u, \beta) = \\ &\sum_{t \notin Q} \frac{\Delta t^2}{2\sigma_x^2} \| q_t v_t + (1 - q_t) f_u(x_{t,\perp}^{r_t}) v_t - \frac{\hat{x}_{t+1} - x_t}{\Delta t} \|_2^2 \\ &- \log p(q_t | x_{t-1}, v_{t-1}, r_t, \beta) + \alpha_u \|u\|_2^2 + \alpha_\beta \|\beta\|_2^2 \end{aligned} \tag{4.17}$$

If the decision to yield q_t were known for each timestep, we would have two separate problems. Fixing q_t for $t \notin Q$ in addition to the estimated values of x_t and v_t, only u is unknown in the first summand. Since the piecewise-linear function f_u is linear in u, the first summand and the u prior form a linear least squares problem for u with box constraints. There is no closed-form solution due to the restricted domain of u being $[-1, 1]^{n_u} \subseteq \mathbb{R}^{n_u}$, but it is readily solved by off-the-shelf linear programming solvers. For the second summand, β is the only unknown variable. The piecewise-linear function f_β is linear in β, so the second summand and the β prior form a logistic regression for β. Since the q_t are not labeled in common datasets, we use block coordinate descent. For blocks we use (u, β) and $\mathbf{q_c}$. We start with a random initial value for each unknown q_t and solve the above subproblems for u and β. Fixing u and β makes the sum separable, so the optimal value of q_i does not depend on that of q_j for $i \neq j$. Choosing the optimal q_t then consists of choosing the binary value that results in lower loss for the summand at timestep t. Repeating these steps to convergence yields the final parameter estimates for u and β.

4.3.4 Implementation

We parameterize the risk function f_β by a regular grid over $[0, 1.6]^2 \in \mathbb{R}^2$ with a stride of 0.4. Since the function inputs are in the log scale, the range includes real distances up to roughly 40 m. Values outside the range are clipped to the nearest gridded point. The vehicle influence f_u is parameterized by points in $[0, 6] \in \mathbb{R}$ with 1 m spacing. This sets the influence's maximum range $u_{\max}$ to 6 m. The priors on the learned functions' parameters (4.12) are set to be weak with $\alpha_u = \frac{1}{20^2}$ and $\alpha_\beta = \frac{1}{10^2}$. Few pedestrians are identified as yielding within the 5-6 m range during the training process. Lack of yielding in the range results in the farthest grid point of f_u having its parameter estimate being pulled to the prior value of zero, shown in Figure 4.3. This

Table 4.1: Predictive performance on DUT and inD datasets. Evaluation metrics are shown as ADE/RMSE in meters. The proposed method OTS outperforms the baselines for both short-term and long-term predictions.

Dataset	DUT					inD				
t (s)	CV	SGAN	MATF-S	MATF	OTS	CV	SGAN	MATF-S	MATF	OTS
1	0.39/0.38	0.62/0.66	1.65/1.87	0.63/0.72	0.22/0.30	0.50/0.50	0.98/1.09	1.01/1.12	0.42/0.50	0.12/0.37
2	0.84/0.82	0.86/0.96	3.19/3.61	1.22/1.40	0.49/0.64	1.10/1.13	1.58/1.79	2.04/2.26	0.86/1.03	0.37/0.83
3	1.31/1.28	1.21/1.43	4.86/5.53	1.87/2.15	0.78/1.01	1.79/1.85	2.24/2.56	3.16/3.48	1.40/1.68	0.67/1.35
4	1.81/1.75	1.67/2.02	6.44/7.38	2.58/2.97	1.09/1.37	2.57/2.64	2.96/3.39	4.36/4.82	2.00/2.40	1.02/1.92
5	2.31/2.22	2.20/2.73	7.98/9.16	3.37/3.85	1.41/1.74	3.42/3.50	3.74/4.28	5.65/6.24	2.65/3.20	1.42/2.53

suggests that the maximum range of 6 m for vehicle influence is sufficiently large. The same effect appears in the risk function's parameters for the risk of vehicles that remain far from the pedestrian. In setting the amount of observational noise, we follow the inD dataset guideline that its typical positioning error is less than 0.1 m. We thus set σ_x to 0.05 m. For inference we find that importance sampling effectively samples the posterior distribution. This enables us to avoid using slower particle filter steps as in other works [99, 27]. Inference with the proposed method relies on receiving the sequence of vehicle positions to make predictions. In most scenarios the positions are known up to the current time, but not into the future. For this case we assume vehicles move at a constant velocity and extrapolate their future positions.

4.4 Experiments

We test the proposed method's ability to predict pedestrian trajectories in the DUT [30] and inD [97] urban datasets. The DUT dataset contains nearly 1800 pedestrians' interactions with vehicles at two scenes. One scene is a marked crosswalk and the other is a shared space. Drivers and pedestrians in both scenes negotiate for priority of passage. While the inD dataset contains no data collected at a shared space, it contains over 11500 road users' trajectories across four unsignalized intersections.

4.4.1 Baselines

We compare to baselines including state-of-the-art methods for pedestrian prediction based on DNNs:

- **Constant Velocity (CV) :** The pedestrian is assumed to travel at a constant velocity.
- **Social GAN (SGAN)**[1] **:** A GAN architecture using a permutation invariant pooling module to capture pedestrian interactions at different scales.
- **Multi-Agent Tensor Fusion (MATF)**[9] **:** A GAN architecture using a global pooling layer to combine trajectory and semantic information.
- **Off the Sidewalk (OTS) :** The probabilistic interaction model introduced in Section 4.3.

Each learning model, including the proposed method, is trained once on each dataset to make predictions on the unseen dataset. Aside from CV, all of the compared methods make probabilistic predictions. For evaluation we sample 100 trajectories from each to compare against the pedestrian's true trajectory. The proposed method operates on observations made at 10 Hz while the other learning baselines operate at lower frequencies. The observations made at 10 Hz are downsampled to 5 Hz for MATF. SGAN is originally designed for 2.5 Hz, but is trained and evaluated at 2 Hz for the sake of comparison as in previous work [78]. All baselines are trained to make 3 s of observations and 5 s of predictions. We also compare to the Multi-Agent Tensor Fusion method trained without semantic information. The method trained with semantic information is denoted MATF-S and the method without is denoted MATF.

4.4.2 Evaluation Metrics

Let $x_{i,t}$ denote the position of the ith pedestrian evaluated in the dataset at timestep t. The corresponding prediction denoted $\hat{x}_{i,t}$ is a random variable since each method is probabilistic. Let N denote the total number of pedestrians evaluated in the dataset. We compare methods with the following metrics:

- *Average Distance Error (ADE):* The expected Euclidean distance between the true position and prediction, used in [27, 40, 78, 1, 9]. ADE at timestep t is:

$$ADE(t) = \frac{1}{N}\sum_{i=1}^{N}\mathbb{E}[\|x_{i,t} - \hat{x}_{i,t}\|_2]$$

- *Root Mean Squared Error (RMSE):* The square root of expected squared error between the true position and prediction, used in [77, 9]. RMSE at timestep t is:

$$RMSE(t) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\mathbb{E}[\|x_{i,t} - \hat{x}_{i,t}\|_2^2]}$$

The ADE measures the distance between the mean of the predicted distribution over the pedestrian's position, and the true position. This can also be viewed as the mean of the error distribution. In contrast, the RMSE is a measure of the second moment of the error distribution. Larger errors thus influence the RMSE more than the ADE.

4.4.3 Pedestrian Prediction

Table 4.1 shows each method's performance on both urban datasets. The proposed method OTS outperforms previous works in both the shared spaces of DUT and unsignalized intersections of inD. SGAN and MATF provide the next best long-term predictions for DUT and inD, respectively. Comparing the performance of MATF-S and MATF shows that learning interactions with semantic data does not necessarily transfer from one scene to another. Despite DUT containing a marked crosswalk

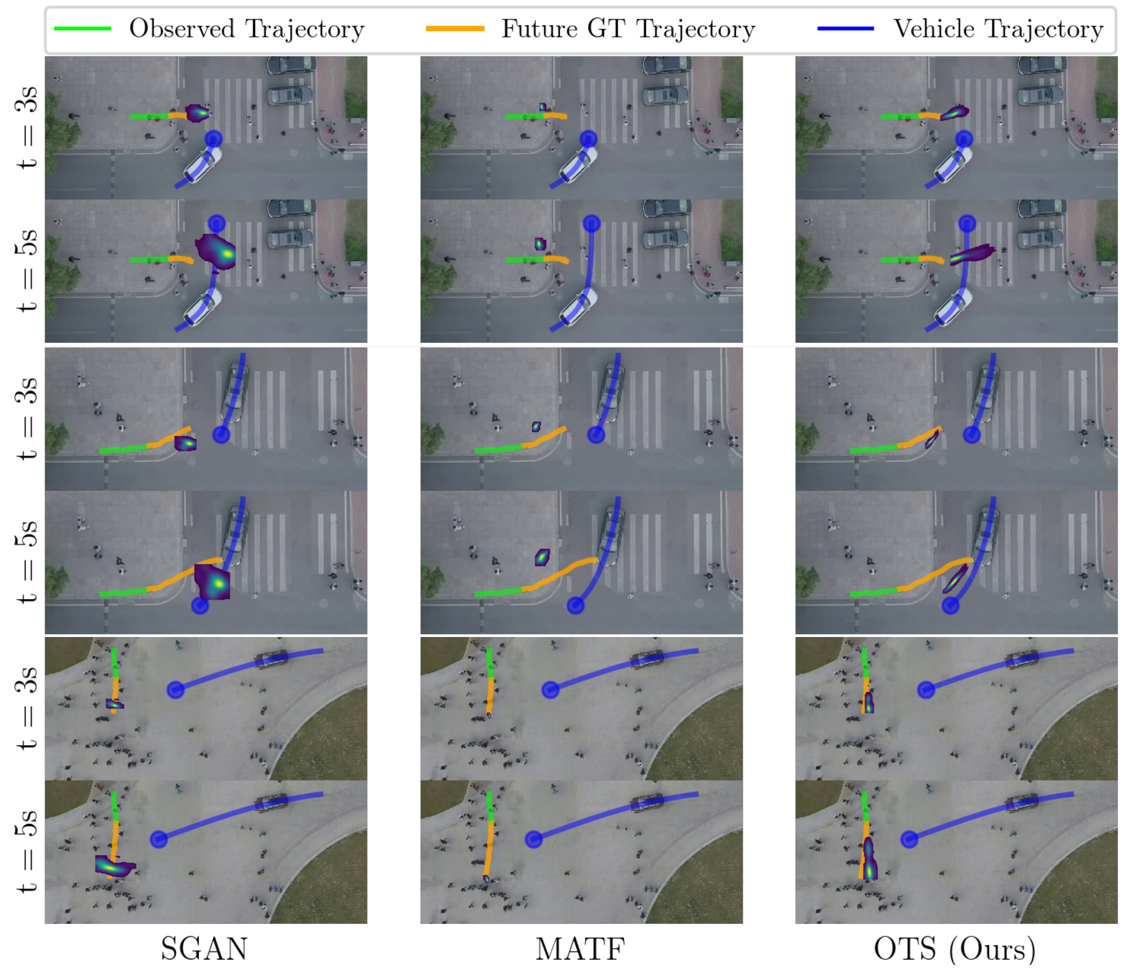


Figure 4.4: Examples of predictions on scenes from DUT. Each method is trained on trajectory data from inD and observes 3 s of each road user's trajectory (pedestrian in solid green) before predicting the next 5 s. Predictions for 3 s and 5 s into the future are shown for each method with likelihood according to the viridis color scale. The orange solid line represents the ground truth trajectory of the pedestrian up to the predicted timestep. The blue solid line represents the vehicle trajectory up to the predicted timestep. The proposed method OTS captures uncertainty in the pedestrian's actions in both traditional crosswalk scenes and shared spaces.

scene similar to those in inD, MATF achieves better performance than MATF-S on inD without using this information. Qualitative examples of predictions are shown in Figure 4.4. The learned risk and vehicle influence functions enable OTS to predict accurately in both traditional crosswalk scenes and shared spaces.

4.4.4 Autonomous Vehicle Planning Scenario

Predictions inform the AV of surrounding road users' future positions. These predictions are then used in motion planning to choose the vehicle's future trajectory. Given that the AV knows its own nominal trajectory, it is possible to use this additional information when predicting how surrounding pedestrians interact with

Table 4.2: Performance for simulated autonomous vehicle scenarios. Evaluation metrics are shown as ADE/RMSE in meters. Predictions made with future trajectory information (OTS-AV) achieve lower error than those made without (OTS).

Dataset	DUT		inD	
t (s)	OTS	OTS-AV	OTS	OTS-AV
1	0.23/0.30	0.22/0.29	0.28/0.38	0.28/0.38
2	0.48/0.63	0.47/0.61	0.61/0.85	0.60/0.84
3	0.74/0.99	0.72/0.95	0.99/1.37	0.98/1.36
4	1.02/1.34	0.98/1.30	1.40/1.95	1.39/1.93
5	1.29/1.69	1.25/1.64	1.85/2.57	1.83/2.55

the AV. We simulate this scenario in the DUT and inD datasets. For evaluation we limit predictions to scenes containing a single moving vehicle. The single vehicle fills the role of the AV, and its future trajectory is used alongside pedestrian observations for prediction. Since the proposed method considers the vehicle trajectory as given, we may use the new information with no changes. We do not compare to the baseline methods in this scenario since each is built only for making predictions based on observations up to the current time. Performance of the proposed method using the trajectory is denoted OTS-AV and shown against the standard OTS in Table 4.2. The trajectory information boosts the performance of OTS-AV, particularly for long-term predictions.

4.4.5 Speed

Previous sections have focused on predictive accuracy, but speed is vital to making a timely response in critical driving scenarios. Here we benchmark the average time to make predictions for a single scenario. Using the open source implementation of each baseline on a GTX 1080 GPU, SGAN finishes computation in 0.43 s and MATF in 0.84 s. The average time for OTS is 0.03 s on a single core of an Intel Core i7-6800K CPU clocked at 3.40 GHz. In contrast to the deep learning methods, having fewer than 40 parameters helps to make OTS responsive.

4.5 Conclusion

We propose a novel probabilistic method to predict pedestrians' trajectories in general scenes such as shared spaces, in addition to more traditional scenes. Experiments on these scenes demonstrates that OTS achieves state-of-the-art performance. The focus on interactions between an individual pedestrian and vehicles is both the method's strength and weakness. The benefits include interpretable model parameters and realtime performance. On the other hand, other types of interactions such as group interactions between pedestrians are ignored. Modeling these types of pedestrian behaviors provides an avenue for future research.

CHAPTER V

A Kinematic Model for Trajectory Prediction in General Highway Scenarios

5.1 Introduction

Highway driving places autonomous vehicles at high speeds and in close proximity to other drivers. Conservative behaviors can completely eliminate the risk of collision in some scenarios [21], but many demand close interaction with less conservative human counterparts. We focus on prediction as an aid to assessing risk and navigating these scenarios. Motion in the short-term is largely constrained by vehicle dynamics and simple kinematic models predict with fair accuracy in this setting [89]. Further motion, however, receives greater influence from the driver's intent to maneuver or interact with other vehicles. State-of-the-art methods train deep neural networks (DNNs) on large trajectory datasets to extract the subtle differences in motion to predict each maneuver. While DNNs are capable of predicting trajectories in general scenarios, their complexity hinders interpreting both the learned parameters and the predictions. This complexity is especially problematic for AVs that drive on behalf of human passengers. Such a responsibility calls for models where the cause of erroneous predictions can be explained. In offering transparency, this can help to make unknown risk known. A complementary focus is to identify predictions that are not well-founded before their use, which can be accomplished by testing whether observed behaviors fall within modeling assumptions [103, 104]. In the context of autonomous driving, testing can signal when the AV has encountered anomalous behavior, or that the current model is inadequate for the situation. While such model validation has seen extensive development for model-based methods, there has been less for model-free methods. The interpretable and model-based nature of kinematic methods is encouraging, but they fail to account for maneuvers and interactions that are not explicitly modeled. Their performance for long-term predictions in general scenarios has correspondingly not reached that of DNNs. In this work we aim to build a method that is both interpretable and achieves high performance in general

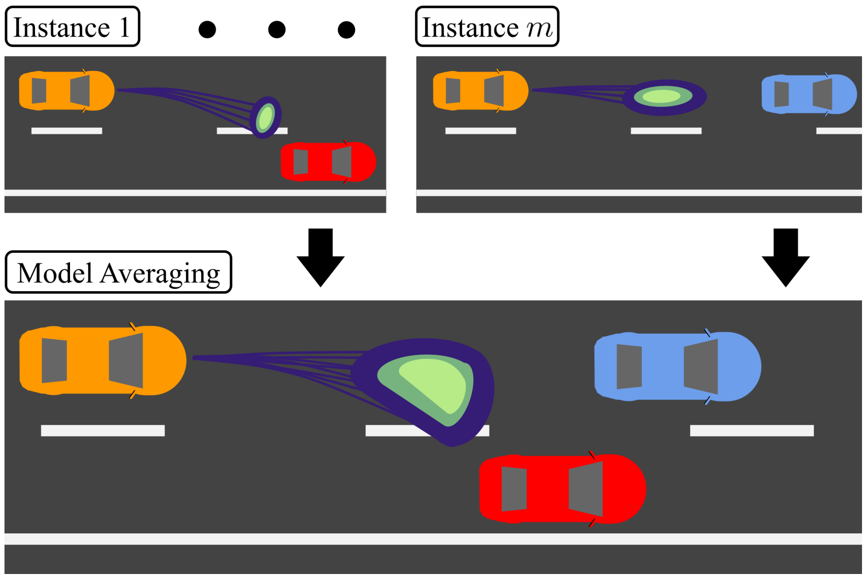


Figure 5.1: Separate instances of the proposed kinematic model *(top)* predict trajectories based on different interactions. Combining these models enables predicting trajectories in general scenarios *(bottom)*.

scenarios. We approach this by first building a kinematic model that describes both lane change and car-following behavior. The resulting model assumes that a target lane and a vehicle to follow have been specified. To predict trajectories in general scenarios, we apply Bayesian model averaging over a suite of these models specified with different target lanes and vehicles. The main contributions of this work are:

1. a novel kinematic model for generating interaction-aware predictions in general highway driving scenarios;
2. a tractable inference procedure;
3. experiments on the highway traffic datasets NGSIM [80] and highD[105] in varied sensing conditions.

This chapter is organized as follows. Section 5.2 describes related methods for trajectory prediction. Section 5.3 introduces the proposed kinematic model and its extension for prediction in general highway scenarios. We compare the proposed method to state-of-the-art approaches in Section 5.4 and conclude in Section 5.5.

5.2 Related Work

Methods that predict trajectories by specifying a kinematic model are described first. The following section describes methods that rely on learning from large trajectory datasets to specify a model for prediction.

5.2.1 Kinematic Methods

The most basic kinematic models assume drivers move at a constant velocity, acceleration, or yaw rate [106, 107, 108, 89]. These models achieve high accuracy over short time horizons due to their close approximation of true vehicle dynamics in common scenarios. Over long time horizons, they often fail to predict drivers' intent to maneuver or interact with other vehicles. One line of research has led to explicit models for lane change maneuvers based on single-integrator models [107, 108] and optimizing over quintic polynomials [85]. Another has instead focused on modeling interactions specifically in the case of car-following behavior. Given a specified leader vehicle to follow, these methods predict the follower will maintain a desired distance and velocity [81], time gap [107], or a minimum distance subject to constraints on acceleration [92]. While these works rely on manually chosen parameters, one approach to improve performance has been to estimate the parameters from observations made online [109, 106, 110]. Online estimation adapts each model to each individual's driving, but generally requires solving a nonconvex problem. Proposed solutions include using particle filters [109], general purpose optimization routines [106], and convex relaxations based on semidefinite programs [110]. The computation involved in each approach hinders their inclusion in larger methods that model general highway driving. In addition, car-following models say little about how to specify the leading vehicle. This is problematic in cases where the leading vehicle may merge out of the follower's lane, or when another vehicle begins to merge into the space between the two. To resolve this, we instead treat the identity of the leader vehicle as unknown, and effectively estimate it online.

The above works largely focus on predicting specific maneuvers. Prediction for general highway driving scenarios has been addressed by combining the results of maneuver-specific models. Taking a linear combination is pursued within the framework of interactive multiple models [108, 89] and leads to a unimodal prediction when the predictions being combined are unimodal. Bayesian model averaging [107] instead weights each component prediction by its relative evidence and produces multi-modal predictions. We adopt this latter approach to better predict multi-modal behavior.

5.2.2 Data-Driven Methods

In predicting arbitrary maneuvers, kinematic models are hampered by their need to explicitly model each maneuver and interaction considered. Data-driven methods overcome this difficulty by leveraging large datasets to learn models for maneuvers and interactions from recorded trajectories. Initial works aimed to learn maneuvers in a model-free framework via Gaussian mixture models (GMMs) [83, 89] or long short-term memory (LSTM) networks [74], and addressed interactions with manually specified cost functions [88, 89]. The need to manually specify cost functions, however, leads to the same difficulties faced by the kinematic models. Subsequent works have aimed to learn models for maneuvers and interactions simultaneously, with many making use of the rich capacity for representation offered by deep neural networks (DNNs). This has also spurred researchers to adapt DNNs to the task of trajectory prediction. Deep neural networks in their base form are deterministic functions of their input, and some works have focused on accounting for the interactions between road users without modeling the uncertainty associated with future trajectory prediction [79, 11]. To adapt DNNs to describe uncertainty while still maintaining determinism, works have instead predicted the parameters corresponding to probability distributions. Distributions predicted by DNNs include discrete distributions over a finite number of trajectories [111] and normal distributions for each predicted timestep [40, 77]. Deo *et al.* [75] achieve both a multi-modal and continuous description of uncertainty by predicting a normal distribution for each type of maneuver from a given class. This makes predictions following a Gaussian mixture model, but relies on maneuvers to be labeled. Other works have aimed to predict GMMs without the aid of labeled data [112, 113]. Training a mixture model without the structure provided by maneuver labels, however, may collapse separated modes into a single mode during the training process [114]. Chai *et al.* [112] propose a two-step procedure to train a mixture model and DNN separately but are unable to benefit from end-to-end training. Occupancy grids present an alternative approach to predict multi-modal distributions [115, 114] but entail a trade-off between the discretization error of a coarse grid and the increased computation imposed by a fine grid. While the above methods predict the parameters that exactly specify a given possibly discrete distribution, a number of methods instead rely on learning a latent distribution from which predicted trajectories are drawn as samples [116, 117, 10, 1, 9, 118]. Several of these works use the variational autoencoder framework to learn a latent distribution over interactions, and model each pairwise interaction between road users [116, 117, 10]. To reduce the computation involved in the pairwise models, one solution has been to consider interactions only between road users within a fixed distance [10]. Social GAN [1] instead proposes a novel pooling module to examine interactions without the need to form all pairs, thereby removing the issues of computational complexity and manual choice of distance threshold. The price of this pooling module is that the spatial relations between road users are lost when using the single pooled result to make predictions. Zhao *et al.* [9] achieve state-of-the-art performance by preserving spatial relations within a tensor that models an inertial frame. Since sampling each model's latent distribution can be costly, DiversityGAN [118] introduces a low-dimensional space to more efficiently sample rare events such as lane changes. Constructing this space, however, depends on labeled data.

5.3 Probabilistic Trajectory Predictions

The proposed kinematic model consists of separate longitudinal and lateral components. Section 5.3.1 states the trajectory problem for which it is built, and provides an overview of the model. The following sections describe the longitudinal component (Section 5.3.2) and the lateral component (Section 5.3.3). The extension to predicting trajectories in general highway scenarios is described in Section 5.3.4, and a tractable inference procedure is detailed in Section 5.3.5.

5.3.1 Problem Statement

We observe the target vehicle's position during a window lasting n timesteps and aim to predict its position until a final timestep T. Let the subset of observed timesteps be given by $S \subseteq \{1, ..., n\}$, where we assume $1 \in S$ without loss of generality. The position at timestep t in the ground plane is denoted $p(t) \in \mathbb{R}^2$. As in other works [106, 107, 108] we assume the position is given in terms of longitudinal $p_1(t)$ and lateral $p_2(t)$ coordinates along the road with $p(t) = (p_1(t), p_2(t))^\intercal$. In the general case, positions can be transformed to this coordinate system using low-degree polynomial models of the road [119]. We also assume we observe the positions of n_v other vehicles over the same observation window. The jth vehicle's position at timestep t is denoted $p^j(t)$. For convenience we denote the collection of other vehicles' observations by $V = \{\{p^j(t)\}_{t=1}^{n} \,|\, j = 1, \ldots, n_v\}$, the target vehicle's observed positions by $p_S = \{p(t)\}_{t \in S}$, and future positions by $p_T = \{p(t)\}_{t=n+1}^{T}$. We now write the prediction task as sampling the target vehicle's future trajectories based on the observed data:

$$p_T \sim \Pr(p_T \,|\, p_S, V). \tag{5.1}$$

The kinematic model used to make predictions relies on driving that is away from the limits of handling, such that vehicle dynamics for lateral and longitudinal motion are approximately decoupled. Since this applies to most highway driving, we model each component as a separate double-integrator. Given Δt as the size of each timestep and the component in $i = 1, 2$, motion is given by:

$$x_i(t+1) = A x_i(t) + B(u_i(t) + \varepsilon_i(t)), \tag{5.2}$$

where

$$A = \begin{pmatrix} 1 & \Delta t \\ 0 & 1 \end{pmatrix}, B = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \tag{5.3}$$

with $u_i(t)$ as a modeled control input and $\varepsilon_i(t)$ is the Gaussian white noise with variance σ_i^2 that is specific to each component. This model may produce negative velocities in stop-and-go traffic, so for the propagation of state over the prediction horizon $t=n,\dots T$ we set longitudinal velocities that would become negative to zero. In addition, the surrounding vehicles' positions are assumed given only over the observation window. For prediction, their states are propagated according to (5.2) with zero control input.

5.3.2 Longitudinal Motion Model

The vehicle's longitudinal motion is described by a car-following model, based on maintaining a desired gap $g*$ and desired velocity $v*$ similar to previous work [81, 110]. We treat these quantities as unknown, and aim to infer them from observed interactions with the leading vehicle. Since the desired values likely vary in time, we approximate them as constant and assume the vehicle is closest to achieving these at the end of the observation window. This results in the priors:

$$g^* \sim N(p_1^i(n) - p_1(n), \sigma_g^2), \tag{5.4}$$

$$v^* \sim N(v_1(n), \sigma_v^2), \tag{5.5}$$

where the ith vehicle is given as the leading vehicle. The longitudinal control is assumed to be calculated over a fixed horizon k_f. Let $u=(u_0,\dots,u_{k_f-1})\in\mathbb{R}^{k_f}$ and let $x_{1,0}$ denote the current longitudinal state of the target vehicle. In addition, let $x_{1,0}^i=(p_{1,0}^i, v_{1,0}^i)^\intercal$ denote the leading vehicle's current state. The longitudinal control $u_1(t)$ is given by the control law $u_1(t) = u_0$ where u solves:

$$\begin{aligned} &\underset{u\in\mathbb{R}^{k_f}}{\text{minimize}}\ \|u\|_2^2 \\ &\text{s.t. } x_1(k_f) = (p_{1,0}^i + k_f\Delta t v_{1,0}^i - g^*, v^*)^\intercal \\ &\qquad x_1(0) = x_{1,0} \\ &\forall\, i = 0,\dots,k_f-1\ x_1(i+1) = Ax_1(i) + Bu_i \end{aligned} \tag{5.6}$$

wherein the driver assumes the leading vehicle moves at a constant speed. In the case of no lead vehicle, $u_1(t)$ is given by:

$$u_1(t) = \frac{v^* - v_1(t)}{k_f}. \tag{5.7}$$

The fact that the control is depends linearly on the state and the variables g^*, v^* is later used during inference.

5.3.3 Lateral Motion Model

Real drivers decide at discrete times to change lanes. Rather than explicitly modeling a switching process we will use an approximate form that assumes there exists exactly one lane change that is partially observed. We make the same assumption as in other works [107, 108] that after the lane change ends, the driver continues within the same lane. Since the duration of a typical lane change is between four and ten seconds [120], and a standard observation window is only three seconds, the majority of merges will be only partially observed. We start by defining the set $\mathcal{K}$ to contain the possible durations of timesteps that remain in the lane change maneuver. The unknown quantities in the model are the target vehicle's actual duration k remaining in the lane change at $t = 1$, and the desired lateral position p_m. As an uninformative prior we assume the duration is uniformly distributed:

$$k \sim U(\mathcal{K}). \tag{5.8}$$

The desired lane is assumed to be given with lateral center $\mu_p \in \mathbb{R}$, with which we assume the desired lateral position p_m is normally distributed about the center:

$$p_m \sim N(\mu_p, \sigma_p^2). \tag{5.9}$$

Since the target vehicle switches to continuing within the lane at some time, we now define the horizon used by the controller at each timestep t as:

$$k_t = \begin{cases} k - t, & \text{if } k - t > 2 \\ k_s, & \text{otherwise} \end{cases} \tag{5.10}$$

where k_s is the fixed horizon used for continuing within the same lane. We note that lane keeping behavior corresponds to choosing p_m within the current lane. To define the control input, let $u = (u_0, \ldots, u_{k_t-1}) \in \mathbb{R}^{k_t}$ and let $x_{2,0}$ denote the current lateral state. The lateral control $u_2(t)$ is given by the time-varying control law $u_2(t) = u_0$ where u solves:

$$\begin{aligned} \underset{u \in \mathbb{R}^{k_t}}{\text{minimize}} \ & \|u\|_2^2 \\ \text{s.t. } & x_2(k_t) = (p_m, 0)^\intercal \\ & x_2(0) = x_{2,0} \\ \forall\, i = 0, \ldots, k_t - 1 \ & x_2(i+1) = Ax_2(i) + Bu_i \end{aligned} \tag{5.11}$$

Though k is unknown, for inference we make use of the fact that the control is a linear function of state and p_m.

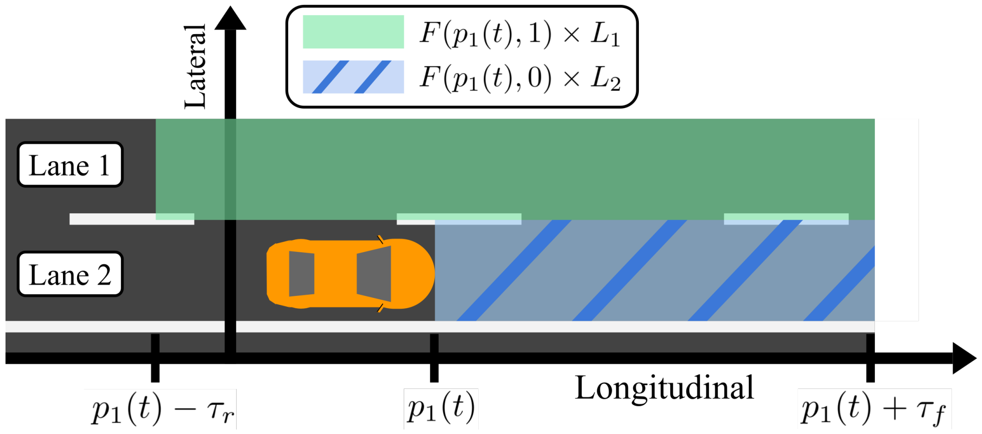


Figure 5.2: Target vehicle's field of view across two lanes (shaded) at timestep t. The view is specified in the lateral direction by the lane extents L_1 and L_2, and in the longitudinal direction by the current position $p_1(t)$, forward distance τ_f, and the rear distance τ_r.

5.3.4 General Highway Predictions

Combining the lateral and longitudinal models from the previous sections, we must specify a target lane and a leading vehicle to predict future trajectories. Trajectories may then be sampled from the model written as:

$$p_T \sim \Pr(p_T \mid p_S, V, i, j), \tag{5.12}$$

where the ith lane and the jth vehicle have been specified, and for notational convenience, we allow $j = \emptyset$ to denote the case of no lead vehicle for the longitudinal motion model. Although model (5.12) may be used to predict trajectories, it requires information that we are unlikely to know: the driver's desired lane and the vehicle to which they adjust their driving. In this section we remove such a need by first identifying a set C containing all relevant (i, j) lane and lead vehicle pairs to consider. Each pair is then used to specify a single model, and we apply Bayesian model averaging to combine them as:

$$\begin{aligned}\Pr(p_T \mid p_S, V) = \\ \sum_{(i,j)\in C} \Pr(p_T \mid p_S, V, i, j) \Pr(i, j \mid p_S, V)\end{aligned} \tag{5.13}$$

with predictions given by (5.1). This decomposition shows that the averaged model takes the trajectories predicted by each component model and weights them by the component's evidence. To build the set C we first introduce the sets that describe the target vehicle's field of view. Let the interval for ith lane's lateral values $[p_{i,l}, p_{i,u}]$ be given by L_i. For the longitudinal extent seen by the target vehicle, we introduce a forward distance τ_f and a rear distance τ_r. Here we assume that for the purposes of car-following, the target vehicle ignores those directly behind it. Then letting p denote the target vehicle's longitude, we define the extent as:

$$F(p, q) = [p - q\tau_r, p + \tau_f], \tag{5.14}$$

where q is a binary value used to exclude the rear portion of the target vehicle's lane. Figure 5.2 shows the target vehicle's view modeled over two lanes. Assume the first observation $p(1)$ corresponds to lane l. Given the ith lane we can now write its set of paired lead vehicles by:

$$\begin{aligned} G(i) = \{j \in \{1, \ldots, m\} \mid \exists t_1, t_2 \in S,\ & p_2^j(t_2) \in L_i, \\ & p_1^j(t_1) \in F(p_1(t_1), \mathbb{1}\{l \neq i\})\} \end{aligned} \tag{5.15}$$

which ensures each lead vehicle has been observed within each extent of the target vehicle's field of view at least once. Denoting the set of lanes adjacent to lane l by $\operatorname{adj}(l)$, we consider only lanes $i \in \{l\} \cup \operatorname{adj}(l)$. The set of lane and lead vehicle pairs is now given by:

$$C = \{i, j \mid j \in G(i)\} \cup \{i, \emptyset \mid G(i) = \emptyset\}, \tag{5.16}$$

where the lanes with no lead vehicles are paired in the second set. The next section describes a tractable method to predict trajectories using the combined model.

5.3.5 Inference

To sample trajectory predictions, we start by making use of the structure within the component longitudinal and lateral models given in Section 5.3.2 and Section 5.3.3 respectively. The longitudinal model depends on the desired gap g^* and velocity $v*$, and the lateral model depends on the target lateral position p_m and merge duration k. The key to efficient inference is that fixing k makes the model's control inputs linear functions of the remaining unknowns. To capitalize on this observation, we integrate over both $k \in \mathcal{K}$ and $(i, j) \in C$, and estimate the remaining parameters via Kalman filtering. This process is explained next. We first collect the random variables needed for prediction as $\theta = (x(n), g*, v*, p_m) \in \mathbb{R}^7$. To obtain a representation more amenable to inference, we rewrite (5.13) as:

$$\Pr(p_T \mid p_S, V) = \sum_{i,j,k} \int \Pr(p_T, \theta, i, j, k \mid p_S, V) d\theta. \tag{5.17}$$

The summand can be decomposed using the chain rule as:

$$\begin{aligned} &\Pr(p_T, \theta, i, j, k \mid p_S, V) = \\ &\qquad \Pr(p_T \mid \theta, i, j, k, p_S, V) \Pr(\theta \mid i, j, k, p_S, V) \\ &\qquad \Pr(i, j, k \mid p_S, V). \end{aligned} \tag{5.18}$$

The first term in the chain represents the prediction of future trajectories by propagating the current state estimate $x(n)$. Since the current state is included in θ, conditional independence implies that we may remove p_S. The second term represents the uncertainty in estimating the current state x_n along with the other unknown parameters, and is exactly the posterior distribution estimated by the Kalman filter. The final term can be rewritten using Bayes' rule:

$$\Pr(i, j, k \,|\, p_S, V) = \frac{\Pr(p_S \,|\, i, j, k, V) \Pr(i, j, k \,|\, V)}{\Pr(p_S \,|\, V)} \tag{5.19}$$

$$\propto \Pr(p_S \,|\, i, j, k, V) \Pr(i, j, k \,|\, V) \tag{5.20}$$

$$= \Pr(p_S \,|\, i, j, k, V), \tag{5.21}$$

where the final equality follows from assuming a uniform distribution over $(i, j, k) \in C \times \mathcal{K}$, independent of V. The resulting term is the marginal probability of the observations p_S under the model specified by i, j, k. We are now in a position to simplify (5.18) and apply (5.21) as:

$$\begin{aligned} &\Pr(p_T, \theta, i, j, k \,|\, p_S, V) \propto \\ &\quad \Pr(p_T \,|\, \theta, i, j, k, V) \Pr(\theta \,|\, i, j, k, p_S, V) \\ &\quad \Pr(p_S \,|\, i, j, k, V). \end{aligned} \tag{5.22}$$

This decomposition suggests the following procedure sample trajectories from (5.17). For each model component specified by $(i, j, k) \in C \times \mathcal{K}$, we use Kalman filtering to compute the last two terms in (5.22). A value of θ can be sampled from its posterior distribution, and propagating the sampled state $x(n)$ yields a sample of p_T. These predictions are then weighted by the marginal probability, and normalized by the sum of marginals across all components. This inference procedure is summarized in Algorithm 3. We note that if the constraint on longitudinal velocities to be nonnegative was removed, the propagation would also be possible within a Kalman filter for each model component. Performing inference with standard filtering recursions allows the predictions for most scenarios to be made with a small number of filtering steps. Scenarios for which C remains constant from the previous timestep will require only $|C \times \mathcal{K}|$ filtering steps, one for each component model, in addition to the steps used to propagate the trajectories into the future. Furthermore, each component model may be updated in parallel.

5.4 Experiments

We evaluate the proposed method's ability to predict trajectories with two highway traffic datasets. The first is the NGSIM [80] dataset, which contains over 9,000

```
Algorithm 3: Inference for Trajectory Prediction
   Input: $p_S$, $V$, $C$, $\mathcal{K}$, $n$, $T$
   Output: Sampled trajectories $\{w^{(l)}, p_T^{(l)}\}$
 1 Let $m = |C \times \mathcal{K}|$
 2 for $l = 1, \ldots, m$ do
 3     Take $l$th $(i, j, k)$ from $C \times \mathcal{K}$
 4     Compute the posterior and marginal probabilities in (5.22) using a Kalman filter with
         model (5.2) specified with $(i, j, k)$.
 5     Sample $\theta^{(l)} \sim \Pr(\theta \mid i, j, k, p_S, V)$
 6     Propagate $\theta^{(l)}$ to obtain $p_T^{(l)}$
 7     $\hat{w}^{(l)} \leftarrow \Pr(p_S \mid i, j, k, V)$
 8 end
 9 $\forall\, l = 1, \ldots, m$ $w^{(l)} \leftarrow \hat{w}^{(l)} / \sum_{i=1}^{m} \hat{w}^{(i)}$
```

unique vehicles recorded at 10 Hz during dense and occasionally stop-and-go traffic at two highways in California. The second dataset is highD[105], which puts greater focus on general driving conditions. It contains over 110,000 unique vehicles and is recorded at 25 Hz across six German highways near Cologne. In each experiment we aim to predict five seconds into future based on a three second window of observations, as in other works [75, 77, 9].

5.4.1 Model Specifications

To match NGSIM we use a timestep of 0.1 s for the proposed model. For lane change duration $\mathcal{K}$ we use a grid of values between zero and 12 seconds with a spacing of 0.5 s. Each driver is also assumed to plan their control inputs for non-merge situations over a ten second horizon, with $k_s = k_f = 100$. Since lane widths on highways are commonly between 3.5 m and 4.5 m, the prior uncertainty σ_p for desired offset to the lane center is set to 1.5 m. The uncertainties σ_g, σ_v for desired gap and velocity are both set to 2 m to provide some prior information. Previous work has shown that providing a small amount of information in the prior distribution can aid in predictions when estimating car-following models online [110]. For the error introduced into control inputs in (5.2), we set the lateral error σ_2 to 0.05 m for all experiments. Since NGSIM consists primarily of dense traffic, we set longitudinal error σ_1 to 0.2 m for NGSIM and otherwise equal to σ_2. The view distances τ_f and τ_r are set to 50 m and 10 m respectively.

5.4.2 Baselines

We compare to prediction methods including DNNs that achieve state-of-the-art performance on the NGSIM dataset:

- **Constant Velocity (CV):** Vehicle motion is modeled by constant velocity.

Table 5.1: Performance on NGSIM and highD datasets shown as average / final time error in meters (best in **bold** and second best underlined).

Bird's Eye View Predictions							
Dataset	Metric	CV	Social LSTM	Social GAN	MATF	Proposed-NI	Proposed
NGSIM	QDE (0.2)	1.99/3.86	**1.70**/3.23	1.82/**3.20**	2.40/4.59	2.01/3.93	1.75/3.42
	ADE	3.56/6.90	4.10/7.86	2.85/5.39	**2.50/4.76**	3.69/7.36	3.14/6.18
	RMSE	4.47/8.64	5.17/9.86	3.76/7.08	**3.41/6.48**	4.67/9.25	4.08/7.97
highD	QDE (0.2)	1.20/2.37	1.31/2.61	2.11/3.89	1.94/3.75	1.05/2.17	**0.99/2.08**
	ADE	2.56/5.04	2.42/4.87	3.03/5.89	2.11/4.02	1.79/3.74	**1.51/3.16**
	RMSE	3.14/6.22	3.64/7.09	6.91/12.57	4.74/8.76	2.24/4.75	**1.92/4.04**

Driver View Predictions							
Dataset	Metric	CV	Social LSTM	Social GAN	MATF	Proposed-NI	Proposed
NGSIM	QDE (0.2)	2.12/4.02	**1.83/3.41**	2.47/4.54	3.27/6.42	2.21/4.21	1.98/3.76
	ADE	3.80/7.20	4.24/8.06	3.50/**6.64**	**3.39/6.64**	4.01/7.79	3.53/6.78
	RMSE	4.86/9.08	5.39/10.15	4.61/8.60	**4.52/8.71**	5.18/9.85	4.67/8.78
highD	QDE (0.2)	**1.27/2.51**	1.55/3.07	2.91/5.26	2.50/4.84	1.50/3.13	1.30/2.70
	ADE	2.68/5.25	2.73/5.42	3.90/7.39	2.68/5.13	2.36/4.96	**1.88/3.90**
	RMSE	3.33/6.52	4.09/7.87	8.35/15.02	5.57/10.33	3.08/6.47	**2.44/5.06**

- **Social LSTM (SLSTM)**[40]**:** An LSTM framework models the influence of nearby vehicles using a grid to define a social pooling module.
- **Social GAN (SGAN)**[1]**:** A GAN architecture that uses a pooling operator to incorporate all road users' interactions at once to provide context for predictions.
- **Multi-Agent Tensor Fusion (MATF)**[9]**:** Drivers' spatial interactions are treated within a tensor that models single global frame to preserve context information.
- **No Interaction (Proposed-NI):** A variant of the proposed method where interactions are ignored. This treats the set of observations of surrounding vehicles as empty.

The open source implementation for each DNN is trained on a separate set of data than that used for evaluation. For NGSIM, each method is trained on data recorded at I-80 then evaluated on data at US-101, and vice-versa. The highD data are separated into one split containing highways labeled one to three and another split containing the remaining highways, labeled four to six. The two splits are then used in the same fashion as the two highway datasets in NGSIM. Since highD is recorded at 25 Hz, it is resampled to match NGSIM at 10 Hz. The DNNs operate on data at a lower frequency, so the input provided during evaluation and training is downsampled to 5 Hz for MATF and 2 Hz for Social LSTM and Social GAN.

5.4.3 Evaluation Metrics

We evaluate how well each method predicts the future with several metrics for probabilistic methods. Let $p_{i,t}$ denote the ith vehicle's true position at timestep t, with the random variable corresponding to its prediction as $\hat{p}_{i,t}$. For each DNN we sample 100 trajectory predictions to fully evaluate the posterior predictive distribution of $\hat{p}_{i,t}$. Let N be the total number of evaluated vehicles. We calculate the following metrics at each second in the five second prediction window, and compare their time average along with their final value:

- *Root Mean Squared Error (RMSE):* The square root of expected squared distance between the true position and prediction, used in [75, 77, 9]. RMSE at timestep t is given by:

$$RMSE(t) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\mathbb{E}[\|p_{i,t} - \hat{p}_{i,t}\|_2^2]} \tag{5.23}$$

- *Average Distance Error (ADE):* The expected distance between the true position and prediction, used in [40, 10, 1, 9]. ADE at timestep t is:

$$ADE(t) = \frac{1}{N}\sum_{i=1}^{N}\mathbb{E}[\|p_{i,t} - \hat{p}_{i,t}\|_2] \tag{5.24}$$

- *Quantile Distance Error (QDE):* The smallest distance traveled from the true position needed to reach a given fraction of the predictions. The value at timestep t with fraction q is:

$$QDE(q,t) = \frac{1}{N}\sum_{i=1}^{N} d_i, \tag{5.25}$$

  where the distance d_i is given by:

$$\begin{aligned} d_i = &\operatorname{argmin} d \\ &\quad \text{s.t. } \Pr(\|p_{i,t} - \hat{p}_{i,t}\|_2 \leq d) \geq q \end{aligned} \tag{5.26}$$

The quantile metric is equivalent to the minimum-of-K metrics [11, 40, 112, 10, 1, 9, 118] when predictions are weighted by the same probability. Measuring the quantile based metric favors predictions that place significant probability mass near the true position, without penalizing additional predictions that may be distant. From the perspective of autonomous vehicles, this is the most relevant metric when we prioritize conservative driving. In contrast, the expectation based metrics place more emphasis on predictions that cluster near the true position. The dependence of RMSE on the squared error also makes it more sensitive to distant predictions. This focus on overall closeness may be more desirable when the aim is to avoid distant predictions that could induce sudden and unwarranted emergency maneuvers.

5.4.4 Bird's-Eye View Predictions

Here we evaluate predictions made with complete observations of all vehicles, as if they were seen from a bird's-eye view. The performance of each method is shown in Table 5.1 *(top)*. The results show a trade-off between minimizing the quantile error and the expectation based errors. Social LSTM achieves low quantile error, it does so at the cost of higher average distances. On the other hand, MATF generates highly accurate but nearly deterministic predictions. The small difference between its quantile distance and average distance errors indicate that the predictions tend to deviate little from the mean prediction. Social GAN achieves more balance than the previous two methods in minimizing the different types of errors. Predictions made by the less deterministic methods are shown in Figure 5.3. Despite the proposed method's limited treatment of interactions, it performs competitively with the other methods, outperforming them for highD scenarios. Considering car-following interaction aids in predicting changes in speed when other vehicles merge into the same lane. Comparison to the ablated version also shows that considering interactions improves the predictions across all metrics.

5.4.5 Driver View Predictions

Realistic driving conditions create occlusions that prevent a clear view of other drivers. Autonomous vehicles are additionally subject to limited sensor range. In this section we simulate these conditions for each vehicle as if it were the autonomous vehicle. The simulated AV only observes vehicles within 50 m of its position along the longitudinal axis, that are not occluded by other vehicles. Occlusions are generated with a simplified model of detections. This treats vehicles as spherical obstacles with a radius of 2 m, and a vehicle is considered occluded if the line from its position to the simulated AV's position collides with any obstacles. We also ensure each observed vehicle is observed for at least one second in total during the observation window. Since the deep learning baselines assume full observations, we fill the missing values assuming constant velocity. The proposed method also assumes the longitudinal positions of surrounding vehicles are given, for which we smooth according to the model dynamics (5.2) with zero control input. Table 5.1 *(bottom)* shows that the more challenging nature of the partially observed case leads to a drop in performance across all methods. The proposed method degrades more gracefully

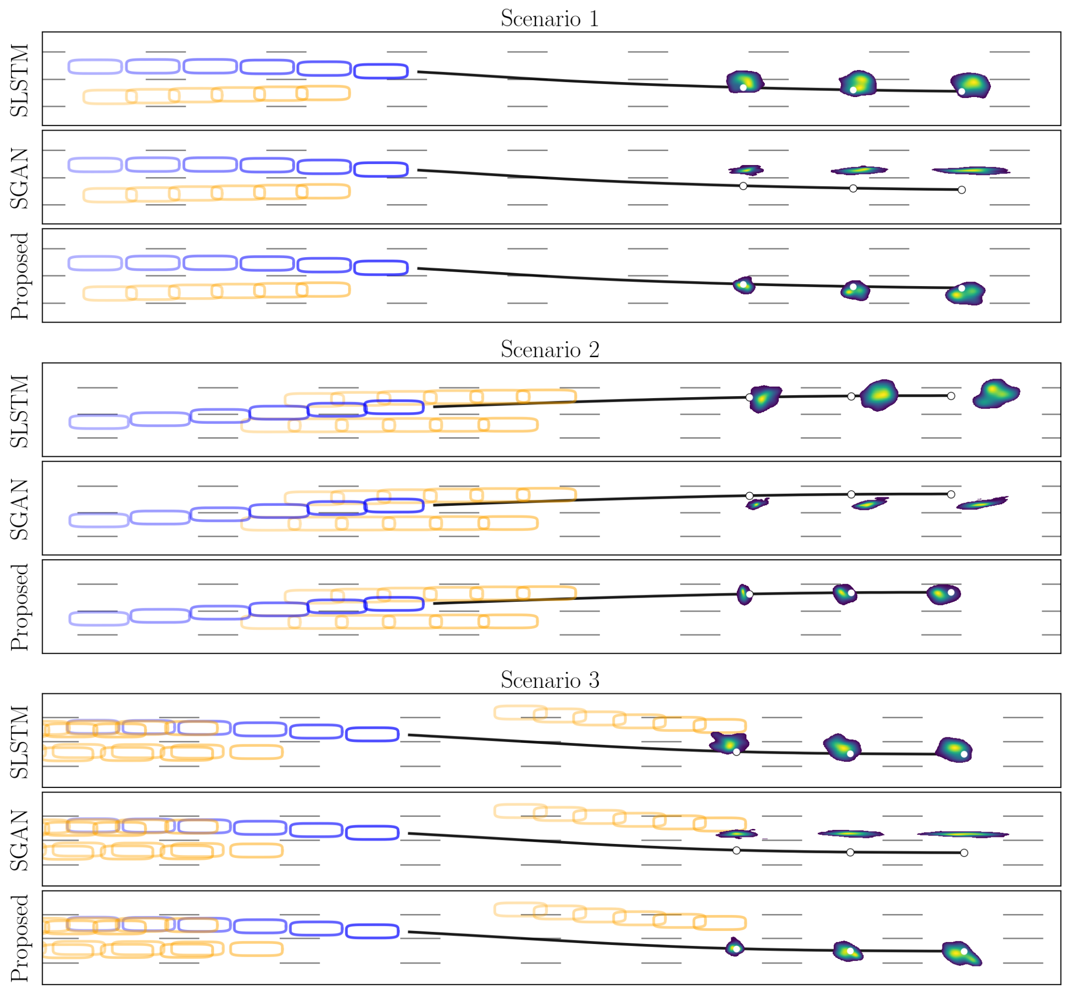


Figure 5.3: Examples of bird's-eye view predictions for dynamic scenarios from highD. Each method observes 3 s of each road user's trajectory (observations shown at each half second) before predicting the next 5 s into the future. The actual future trajectory is shown in black with white circles marking the driver's position at 3 s, 4 s, and 5 s. Each method's predictions are shown for these timesteps with likelihood given by the viridis color scale.

than the baselines, and closes the gaps in performance on NGSIM.

5.5 Conclusion

We propose a novel kinematic model to describe both car-following and lane changing behavior. This provides a means of obtaining interpretable trajectory predictions when the leading vehicle and desired lane are known. Through Bayesian model averaging, we extend the model to predict trajectories for general highway scenarios in which the designation of leader and follower vehicle may be more ambiguous, and the desired lane is not known. Experiments on the NGSIM and highD datasets demonstrate that the method is competitive and can outperform state-of-the-art prediction methods. These findings are shown to hold across varied sensing conditions including both perfect sensors and realistic sensors subject to occlusions. A benefit of the proposed model's interpretable nature also lies in identifying its weaknesses. The kinematic model does not impose constraints, which is less appropriate for describing vehicles' constrained lateral motion at low velocities. A possible remedy is to transition to a different model of motion at low velocities similar to [85]. The proposed model also assumes that longitudinal and lateral motion are approximately decoupled, which is unrealistic for aggressive maneuvers that are limited across both by friction. Incorporating these more realistic vehicle dynamics and examining other forms of interaction between drivers provide avenues for future research.

CHAPTER VI

Conclusion and Future Directions

6.1 Conclusion

This thesis has examined interacting road users and developed interpretable methods to predict their future trajectories across a range of scenarios in urban and highway environments. Beginning with building a prediction model, these methods have addressed fitting models to limited data and automating the fitting procedure. Real-time inference procedures developed for these models have then provided the means to quickly generate predictions and handle missing data due to limited sensor range and occlusions. Experiments on real-world pedestrian and highway traffic datasets have validated the methods' performance and ability to predict road user behavior across different settings.

6.2 Future Directions

The methods developed in this thesis touch on but leave ample room for development along the following directions.

Nominal Trajectories

While predictions typically rely on information accumulated up to the current time, autonomous vehicles may also have available information about the future in terms of a nominal trajectory plan. Without including this future information, the predicted future appears the same to the planner regardless of the actions it plans. Incorporating the future information thus has the potential to aid the planner in selecting plans in addition to making more accurate predictions. One caveat is that the nominal plan may not correspond to what the autonomous vehicle executes, and certain nominal plans may even lack the credibility needed to significantly change how surrounding road users act. Further developing this interface between prediction and planning provides one avenue for future research.

Model Validation and Abnormality Detection

The process of fitting prediction models to data often entails maximizing average performance rather than robustness to worst-case scenarios. This leads to a greater focus on describing common behaviors at the expense of rare ones. Quickly detecting that a given model is poorly suited to predicting a road user's behavior then, can enable the autonomous vehicle to act more cautiously to compensate for the identified uncertainty. Goodness of fit and other validation metrics offer means to gauge model performance without the need to compare predictions to future observations, and abnormality detection methods can assess when additional caution may be warranted. Integrating these with prediction methods could support additional evaluation metrics that account for both traditional prediction accuracy and the costs of compensating with planning behaviors.

Calibration to Weakly Labeled Data

Datasets facilitate the development of prediction methods for trajectories, but high level intent, such as whether a driver intends to change lanes or a pedestrian intends to cross the road, is often both useful and not labeled. The lack of labeling in part corresponds to ambiguity in road users' intent. A driver may be waiting to change lanes and only do so some time later. Fortunately, observing that the lane change was eventually completed implies the intent to change lanes was present for at least one instant in time. The lack of more specific time information makes this label for intent a weak label. Use of weak labels in model fitting procedures has the potential to broaden the class of models that can be reasonably fit.

APPENDIX

APPENDIX A

Proofs

We aim to show that the nonconvex problem given in (NC1) is equivalent to the convex reformulation in (P). We first state the dual semidefinite program (SDP) of (P):

$$\text{(D)} \qquad \begin{aligned} &\underset{s,\mu\in\mathbb{R}}{\text{maximize}}\ s \\ &\text{s.t.} \begin{pmatrix} \frac{1}{2}D^{\intercal}D + \mu E & -b^{\intercal}D + \mu c \\ (-b^{\intercal}D + \mu c)^{\intercal} & \frac{1}{2}b^{\intercal}b - s \end{pmatrix} \succeq 0 \end{aligned}$$

We use the following special case of [121, Theorem 6].

Corollary A.1. *Let $r : \mathbb{R}^n \to \mathbb{R}$ be defined as* (3.23)*. Suppose there exist vectors $x_1, x_2 \in \mathbb{R}^n$ such that $r(x_1) < 0 < r(x_2)$. If the nonconvex problem* (NC1) *has value that is bounded below, the dual SDP* (D) *always has an optimal solution (s^*, μ^*) with optimal value equal to the infimum of* (NC1)*. Furthermore the infimum of* (NC1) *is attained when the dual SDP possesses a feasible set that is not a singleton.*

Proof. This follows immediately from [121, Theorem 6]. $\square$

Remark A.2. Such x_1, x_2 can easily be found by taking $x_1 = (\frac{1}{2}, \frac{1}{2}, 2, 2)$ and $x_2 = (\frac{1}{2}, \frac{1}{2}, 2, \frac{1}{2})$, yielding $r(x_1) = (\frac{1}{2})2 - 2 < 0 < (\frac{1}{2})2 - \frac{1}{2} = r(x_2)$.

We now aim to show that the feasible μ are not unique to obtain equivalency.

Lemma A.3. *If $D \in \mathbb{R}^{m,n}$ with $m \geq n$ as defined in* (D) *has full rank, then the formulations* (NC1) *and* (P) *are equivalent and the optimal solution is attained.*

Proof. First note that D being full rank implies $D^{\intercal}D \succ 0$. For sufficiently small $u \in \mathbb{R}$, $\frac{1}{2}D^{\intercal}D + uI \succ 0$. For these u, $\frac{1}{2}D^{\intercal}D + \frac{u}{\|E\|_2}E \succ 0$ so the interior of $\{\mu \in \mathbb{R} : \frac{1}{2}D^{\intercal}D + \mu E \succeq 0\}$ is nonempty. Since there exist $s \in \mathbb{R}$ such that these μ are feasible for (D), and by the previous remarks, we can apply Corollary A.1 to obtain the desired result. $\square$

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi, “Social gan: Socially acceptable trajectories with generative adversarial networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 2255–2264.

[2] US Department of Transportation. (2019) Highway statistics 2018. Accessed on: 2020-05-5. Available: https://www.fhwa.dot.gov/policyinformation/statistics/2018/. [Online]. Available: https://www.fhwa.dot.gov/policyinformation/statistics/2018/

[3] ——. (2020) Automated vehicles for safety. Accessed on: 2020-05-5. Available: https://www.nhtsa.gov/technology-innovation/automated-vehicles-safety. [Online]. Available: https://www.nhtsa.gov/technology-innovation/automated-vehicles-safety

[4] B. Caulfield, “Estimating the environmental benefits of ride-sharing: A case study of dublin,” *Transportation Research Part D: Transport and Environment*, vol. 14, no. 7, pp. 527–531, 2009.

[5] A. Carlson, R. Vasudevan, and M. Johnson-Roberson, “Shadow transfer: Single image relighting for urban road scenes,” *arXiv:1909.10363*, 2019.

[6] W. Kim, M. Srinivasan Ramanagopal, C. Barto, M. Yu, K. Rosaen, N. Goumas, R. Vasudevan, and M. Johnson-Roberson, “Pedx: Benchmark dataset for metric 3d pose estimation of pedestrians in complex urban intersections,” *IEEE Robotics and Automation Letters*, pp. 1–8, 2019. [Online]. Available: http://pedx.io/

[7] M. S. Ramanagopal, C. Anderson, R. Vasudevan, and M. Johnson-Roberson, “Failing to learn: Autonomously identifying perception failures for self-driving cars,” *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3860–3867, 2018.

[8] J. Zhang, W. Chen, Y. Wang, R. Vasudevan, and M. Johnson-Roberson, “Point set voting for partial point cloud analysis,” *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 596–603, 2021.

[9] T. Zhao, Y. Xu, M. Monfort, W. Choi, C. Baker, Y. Zhao, Y. Wang, and Y. N. Wu, “Multi-agent tensor fusion for contextual trajectory prediction,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 12 126–12 134.

[10] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, “Trajectron++: Multi-agent generative trajectory forecasting with heterogeneous data for control,” *arXiv preprint arXiv:2001.03093*, 2020.

[11] J. Gao, C. Sun, H. Zhao, Y. Shen, D. Anguelov, C. Li, and C. Schmid, “Vectornet: Encoding hd maps and agent dynamics from vectorized representation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 11 525–11 533.

[12] X. Du, R. Vasudevan, and M. Johnson-Roberson, “Bio-lstm: A biomechanically inspired recurrent neural network for 3-d pedestrian pose and gait prediction,” *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1501–1508, 2019.

[13] E. Corona, A. Pumarola, G. Alenya, and F. Moreno-Noguer, "Context-aware human motion prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2020, pp. 6992–7001.

[14] M.-Y. Yu, R. Vasudevan, and M. Johnson-Roberson, "Occlusion-aware risk assessment for autonomous driving in urban environments," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 2235–2241, 2019.

[15] Y. Nager, A. Censi, and E. Frazzoli, "What lies in the shadows? safe and computation-aware motion planning for autonomous vehicles using intent-aware dynamic shadow regions," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 5800–5806.

[16] M.-Y. Yu, R. Vasudevan, and M. Johnson-Roberson, "Risk assessment and planning with bidirectional reachability for autonomous driving," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 5363–5369.

[17] S. Vaskov, H. Larson, S. Kousik, M. Johnson-Roberson, and R. Vasudevan, "Not-at-fault driving in traffic: A reachability-based approach," in *IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 2785–2790.

[18] S. Kousik, "Reachability-based trajectory design," Ph.D. dissertation, University of Michigan, 2020.

[19] M. Althoff and J. M. Dolan, "Online verification of automated road vehicles using reachability analysis," *IEEE Transactions on Robotics*, vol. 30, no. 4, pp. 903–918, 2014.

[20] S. L. Herbert, M. Chen, S. Han, S. Bansal, J. F. Fisac, and C. J. Tomlin, "Fastrack: A modular framework for fast and guaranteed safe motion planning," in *IEEE 56th Annual Conference on Decision and Control (CDC)*. IEEE, 2017, pp. 1517–1522.

[21] K. Leung, E. Schmerling, M. Zhang, M. Chen, J. Talbot, J. C. Gerdes, and M. Pavone, "On infusing reachability-based safety assurance within planning frameworks for human–robot vehicle interactions," *The International Journal of Robotics Research*, vol. 39, no. 10-11, pp. 1326–1345, 2020.

[22] P. Trautman and A. Krause, "Unfreezing the robot: Navigation in dense, interacting crowds," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2010, pp. 797–803.

[23] Y. Pan, Q. Lin, H. Shah, and J. M. Dolan, "Safe planning for self-driving via adaptive constrained ilqr," *arXiv preprint arXiv:2003.02757*, 2020.

[24] M. Koschi, C. Pek, M. Beikirch, and M. Althoff, "Set-based prediction of pedestrians in urban environments considering formalized traffic rules," in *21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 2704–2711.

[25] N. Jaipuria, G. Habibi, and J. P. How, "Learning in the curbside coordinate frame for a transferable pedestrian trajectory prediction model," in *21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 3125–3131.

[26] B. Liu, E. Adeli, Z. Cao, K.-H. Lee, A. Shenoi, A. Gaidon, and J. C. Niebles, "Spatiotemporal relationship reasoning for pedestrian intent prediction," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3485–3492, 2020.

[27] C. Blaiotta, "Learning generative socially aware models of pedestrian motion," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3433–3440, 2019.

[28] S. K. Jayaraman, D. Tilbury, J. Yang, A. Pradhan, and L. Robert, "Analysis and prediction of pedestrian crosswalk behavior during automated vehicle interactions," in *IEEE International Conference on Robotics and Automation (ICRA).* IEEE, 2020, pp. 6426–6432.

[29] M. S. Darms, P. E. Rybski, C. Baker, and C. Urmson, "Obstacle detection and tracking for the urban challenge," *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 3, pp. 475–485, 2009.

[30] D. Yang, L. Li, K. Redmill, and Ü. Özgüner, "Top-view trajectories: A pedestrian dataset of vehicle-crowd interaction from controlled experiments and crowded campus," in *IEEE Intelligent Vehicles Symposium (IV).* IEEE, 2019, pp. 899–904.

[31] K. M. Kitani, B. D. Ziebart, J. A. Bagnell, and M. Hebert, "Activity forecasting," in *Proceedings of the European Conference on Computer Vision (ECCV).* Springer, 2012, pp. 201–214.

[32] V. Karasev, A. Ayvaci, B. Heisele, and S. Soatto, "Intent-aware long-term prediction of pedestrian motion," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 2543–2549.

[33] D. Helbing and P. Molnar, "Social force model for pedestrian dynamics," *Physical review E*, vol. 51, no. 5, p. 4282, 1995.

[34] M. Treiber and A. Kesting, "Traffic flow dynamics," *Traffic Flow Dynamics: Data, Models and Simulation, Springer-Verlag Berlin Heidelberg*, 2013.

[35] A. D. Dragan, K. C. Lee, and S. S. Srinivasa, "Legibility and predictability of robot motion," in *8th ACM/IEEE International Conference on Human-Robot Interaction (HRI).* IEEE, 2013, pp. 301–308.

[36] K. Hornik, M. Stinchcombe, H. White, *et al.*, "Multilayer feedforward networks are universal approximators." *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.

[37] I. Steinwart, "Support vector machines are universally consistent," *Journal of Complexity*, vol. 18, no. 3, pp. 768–791, 2002.

[38] A. Palazzi, D. Abati, F. Solera, R. Cucchiara, *et al.*, "Predicting the driver's focus of attention: the dr (eye) ve project," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 7, pp. 1720–1733, 2018.

[39] M. Luber, J. A. Stork, G. D. Tipaldi, and K. O. Arras, "People tracking with human motion predictions from social forces," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2010, pp. 464–469.

[40] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social lstm: Human trajectory prediction in crowded spaces," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 961–971.

[41] Y. Luo, P. Cai, A. Bera, D. Hsu, W. S. Lee, and D. Manocha, "Porca: Modeling and planning for autonomous driving among many pedestrians," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3418–3425, 2018.

[42] S. Yi, H. Li, and X. Wang, "Pedestrian behavior understanding and prediction with deep neural networks," in *Proceedings of the European Conference on Computer Vision (ECCV).* Springer, 2016, pp. 263–279.

[43] Y. Xu, Z. Piao, and S. Gao, "Encoding crowd interaction with deep neural network for pedestrian trajectory prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 5275–5284.

[44] N. Nikhil and B. T. Morris, "Convolutional neural network for trajectory prediction," *arXiv preprint arXiv:1809.00696*, 2018.

[45] S. Pellegrini, A. Ess, and L. Van Gool, "Improving data association by joint modeling of pedestrian trajectories and groupings," in *Proceedings of the European Conference on Computer Vision (ECCV).* Springer, 2010, pp. 452–465.

[46] A. Robicquet, A. Sadeghian, A. Alahi, and S. Savarese, "Learning social etiquette: Human trajectory understanding in crowded scenes," in *Proceedings of the European Conference on Computer Vision (ECCV).* Springer, 2016, pp. 549–565.

[47] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu, "Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 7, pp. 1325–1339, 2014.

[48] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Adv. Neural Info. Process. Syst.*, 2012, pp. 1097–1105.

[49] M. Johnson-Roberson, C. Barto, R. Mehta, S. N. Sridhar, K. Rosaen, and R. Vasudevan, "Driving in the matrix: Can virtual worlds replace human-generated annotations for real world tasks?" in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 746–753.

[50] G. Rogez and C. Schmid, "Mocap-guided data augmentation for 3d pose estimation in the wild," in *Advances in Neural Information Processing Systems*, 2016, pp. 3108–3116.

[51] S. Pellegrini, A. Ess, K. Schindler, and L. Van Gool, "You'll never walk alone: Modeling social behavior for multi-target tracking," in *IEEE 12th International Conference on Computer Vision*, 2009, pp. 261–268. [Online]. Available: http://www.vision.ee.ethz.ch/en/datasets/

[52] A. Lerner, Y. Chrysanthou, and D. Lischinski, "Crowds by example," in *Computer Graphics Forum*, vol. 26, no. 3. Wiley Online Library, 2007, pp. 655–664. [Online]. Available: https://graphics.cs.ucy.ac.cy/research/downloads/crowd-data

[53] A. Vemula, K. Muelling, and J. Oh, "Social attention: Modeling attention in human crowds," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 1–7.

[54] T. Fernando, S. Denman, S. Sridharan, and C. Fookes, "Soft+ hardwired attention: An lstm framework for human trajectory prediction and abnormal event detection," *Neural networks*, vol. 108, pp. 466–478, 2018.

[55] H. Xue, D. Q. Huynh, and M. Reynolds, "Ss-lstm: A hierarchical lstm model for pedestrian trajectory prediction," in *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2018, pp. 1186–1194.

[56] A. Sadeghian, V. Kosaraju, A. Sadeghian, N. Hirose, and S. Savarese, "Sophie: An attentive gan for predicting paths compliant to social and physical constraints," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 1349–1358.

[57] H. Manh and G. Alaghband, "Scene-lstm: A model for human trajectory prediction," *arXiv preprint arXiv:1808.04018*, 2018.

[58] S. R. Richter, V. Vineet, S. Roth, and V. Koltun, "Playing for data: Ground truth from computer games," in *Proceedings of the European Conference on Computer Vision (ECCV).* Springer, 2016, pp. 102–118.

[59] D. Kappler, J. Bohg, and S. Schaal, "Leveraging big data for grasp planning," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 4304–4311.

[60] R. Diankov, "Automated construction of robotic manipulation programs," PhD dissertation, The Robotics Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, 2010.

[61] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 23–30.

[62] A. Prakash, S. Boochoon, M. Brophy, D. Acuna, E. Cameracci, G. State, O. Shapira, and S. Birchfield, "Structured domain randomization: Bridging the reality gap by context-aware synthetic data," *arXiv preprint arXiv:1810.10093*, 2018.

[63] J. Tremblay, A. Prakash, D. Acuna, M. Brophy, V. Jampani, C. Anil, T. To, E. Cameracci, S. Boochoon, and S. Birchfield, "Training deep networks with synthetic data: Bridging the reality gap by domain randomization," *arXiv preprint arXiv:1804.06516*, 2018.

[64] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 779–788.

[65] X. Cui, V. Goel, and B. Kingsbury, "Data augmentation for deep neural network acoustic modeling," *IEEE/ACM Transactions on Audio Speech Language Processing*, vol. 23, no. 9, pp. 1469–1477, 2015.

[66] Y. Xu, R. Jia, L. Mou, G. Li, Y. Chen, Y. Lu, and Z. Jin, "Improved relation classification by deep recurrent neural networks with data augmentation," *arXiv preprint arXiv:1601.03651*, 2016.

[67] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, "Learning from simulated and unsupervised images through adversarial training," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2107–2116.

[68] L. Sixt, B. Wild, and T. Landgraf, "Rendergan: Generating realistic labeled data," *Frontiers in Robotics and AI*, vol. 5, p. 66, 2018.

[69] J. Li, K. A. Skinner, R. M. Eustice, and M. Johnson-Roberson, "Watergan: Unsupervised generative network to enable real-time color correction of monocular underwater images," *IEEE Robotics and Automation Letters*, vol. 3, no. 1, pp. 387–394, 2018.

[70] C. F. V. Loan, *Introduction to scientific computing: a matrix-vector approach using MATLAB*. Upper Saddle River, New Jersey: Prentice-Hall, 1999.

[71] A. Lawitzky, D. Althoff, C. F. Passenberg, G. Tanzmeister, D. Wollherr, and M. Buss, "Interactive scene prediction for automotive applications," in *IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2013, pp. 1028–1033.

[72] S. Lefèvre, D. Vasquez, and C. Laugier, "A survey on motion prediction and risk assessment for intelligent vehicles," *ROBOMECH Journal*, vol. 1, no. 1, pp. 1–14, 2014.

[73] B. Kim, C. M. Kang, J. Kim, S. H. Lee, C. C. Chung, and J. W. Choi, "Probabilistic vehicle trajectory prediction over occupancy grid map via recurrent neural network," in *IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2017, pp. 399–404.

[74] L. Xin, P. Wang, C.-Y. Chan, J. Chen, S. E. Li, and B. Cheng, "Intention-aware long horizon trajectory prediction of surrounding vehicles using dual lstm networks," in *IEEE 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 1441–1446.

[75] N. Deo and M. M. Trivedi, "Convolutional social pooling for vehicle trajectory prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 1468–1476.

[76] Y. Hu, W. Zhan, and M. Tomizuka, "Probabilistic prediction of vehicle semantic intention and motion," in *IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 307–313.

[77] R. Chandra, U. Bhattacharya, A. Bera, and D. Manocha, "Traphic: Trajectory prediction in dense and heterogeneous traffic using weighted interactions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 8483–8492.

[78] Y. Ma, X. Zhu, S. Zhang, R. Yang, W. Wang, and D. Manocha, "Trafficpredict: Trajectory prediction for heterogeneous traffic-agents," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 6120–6127.

[79] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. Torr, and M. Chandraker, "Desire: Distant future prediction in dynamic scenes with interacting agents," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 336–345.

[80] US Department of Transportation. (2008) Ngsim - next generation simulation. Accessed on: 2019-06-30. Available: http://www.ngsim.fhwa.dot.gov/. [Online]. Available: http://www.ngsim.fhwa.dot.gov/

[81] J. Wei, J. M. Dolan, and B. Litkouhi, "Autonomous vehicle social behavior for highway entrance ramp management," in *IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2013, pp. 201–207.

[82] A. Berthelot, A. Tamke, T. Dang, and G. Breuel, "Handling uncertainties in criticality assessment," in *IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2011, pp. 571–576.

[83] J. Wiest, M. Höffken, U. Kreßel, and K. Dietmayer, "Probabilistic trajectory prediction with gaussian mixture models," in *IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2012, pp. 141–146.

[84] N. Kaempchen, K. Weiss, M. Schaefer, and K. C. Dietmayer, "Imm object tracking for high dynamic driving maneuvers," in *IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2004, pp. 825–830.

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

[86] M. Sarvi and M. Kuwahara, "Microsimulation of freeway ramp merging processes under congested traffic conditions," *IEEE Transactions on Intelligent Transportation Systems*, vol. 8, no. 3, pp. 470–479, 2007.

[87] A. Kondyli and L. Elefteriadou, "Modeling driver behavior at freeway–ramp merges," *Transportation Research Record*, vol. 2249, no. 1, pp. 29–37, 2011.

[88] M. Bahram, C. Hubmann, A. Lawitzky, M. Aeberhard, and D. Wollherr, "A combined model- and learning-based framework for interaction-aware maneuver prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 6, pp. 1538–1550, 2016.

[89] N. Deo, A. Rangesh, and M. M. Trivedi, "How would surround vehicles move? a unified framework for maneuver classification and motion prediction," *IEEE Transactions on Intelligent Vehicles*, vol. 3, no. 2, pp. 129–140, 2018.

[90] C. Dong, J. M. Dolan, and B. Litkouhi, "Smooth behavioral estimation for ramp merging control in autonomous driving," in *IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 1692–1697.

[91] C. Hubmann, J. Schulz, G. Xu, D. Althoff, and C. Stiller, "A belief state planner for interactive merge maneuvers in congested traffic," in *IEEE 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 1617–1624.

[92] M. Treiber, A. Hennecke, and D. Helbing, "Congested traffic states in empirical observations and microscopic simulations," *Physical Review E*, vol. 62, no. 2, pp. 1805–1824, 2000.

[93] M. S. Andersen, J. Dahl, and L. Vandenberghe. Cvxopt. Accessed on: 2019-08-14. [Online]. Available: https://cvxopt.org/

[94] C. Schöller, V. Aravantinos, F. Lay, and A. Knoll, "What the constant velocity model can teach us about pedestrian motion prediction," *IEEE Robotics and Automation Letters*, 2020.

[95] V. Kuleshov, N. Fenner, and S. Ermon, "Accurate uncertainties for deep learning using calibrated regression," in *International Conference on Machine Learning (ICML)*, 2018, pp. 2796–2804.

[96] B. Anvari, M. G. Bell, A. Sivakumar, and W. Y. Ochieng, "Modelling shared space users via rule-based social force model," *Transportation Research Part C: Emerging Technologies*, vol. 51, pp. 83–103, 2015.

[97] J. Bock, R. Krajewski, T. Moers, S. Runde, L. Vater, and L. Eckstein, "The ind dataset: A drone dataset of naturalistic road user trajectories at german intersections," *arXiv preprint arXiv:1911.07602*, 2019.

[98] Y. Hashimoto, G. Yanlei, L.-T. Hsu, and K. Shunsuke, "A probabilistic model for the estimation of pedestrian crossing behavior at signalized intersections," in *IEEE 18th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2015, pp. 1520–1526.

[99] J. F. Kooij, F. Flohr, E. A. Pool, and D. M. Gavrila, "Context-based path prediction for targets with switching dynamics," *International Journal of Computer Vision*, vol. 127, no. 3, pp. 239–262, 2019.

[100] W. Zeng, P. Chen, H. Nakamura, and M. Iryo-Asano, "Application of social force model to pedestrian behavior analysis at signalized crosswalk," *Transportation research part C: emerging technologies*, vol. 40, pp. 143–159, 2014.

[101] V. Kosaraju, A. Sadeghian, R. Martín-Martín, I. Reid, H. Rezatofighi, and S. Savarese, "Social-bigat: Multimodal trajectory forecasting using bicycle-gan and graph attention networks," in *Advances in Neural Information Processing Systems*, 2019, pp. 137–146.

[102] C. Gourieroux, A. Monfort, and E. Renault, "Consistent pseudo-maximum likelihood estimators," *Annals of Economics and Statistics*, no. 125/126, pp. 187–218, 2017.

[103] A. Gelman, X.-L. Meng, and H. Stern, "Posterior predictive assessment of model fitness via realized discrepancies," *Statistica sinica*, pp. 733–760, 1996.

[104] J. Fan and L.-S. Huang, "Goodness-of-fit tests for parametric regression models," *Journal of the American Statistical Association*, vol. 96, no. 454, pp. 640–652, 2001.

[105] R. Krajewski, J. Bock, L. Kloeker, and L. Eckstein, "The highd dataset: A drone dataset of naturalistic vehicle trajectories on german highways for validation of highly automated driving systems," in *21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 2118–2125.

[106] J. Sörstedt, L. Svensson, F. Sandblom, and L. Hammarstrand, "A new vehicle motion model for improved predictions and situation assessment," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 4, pp. 1209–1219, 2011.

[107] M. Schreier, V. Willert, and J. Adamy, "An integrated approach to maneuver-based trajectory prediction and criticality assessment in arbitrary road environments," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 10, pp. 2751–2766, 2016.

[108] G. Xie, H. Gao, L. Qian, B. Huang, K. Li, and J. Wang, "Vehicle trajectory prediction by integrating physics-and maneuver-based approaches using interactive multiple models," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 7, pp. 5999–6008, 2017.

[109] S. Hoermann, M. Bach, and K. Dietmayer, "Dynamic occupancy grid prediction for urban autonomous driving: A deep learning approach with fully automatic labeling," in *IEEE International Conference on Robotics and Automation.* IEEE, 2018, pp. 2056–2063.

[110] C. Anderson, R. Vasudevan, and M. Johnson-Roberson, "Low latency trajectory predictions for interaction aware highway driving," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5456–5463, 2020.

[111] M. Liang, B. Yang, R. Hu, Y. Chen, R. Liao, S. Feng, and R. Urtasun, "Learning lane graph representations for motion forecasting," in *Proceedings of the European Conference on Computer Vision (ECCV).* Springer, 2020, pp. 541–556.

[112] Y. Chai, B. Sapp, M. Bansal, and D. Anguelov, "Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction," in *Conference on Robot Learning*, 2020, pp. 86–99.

[113] J. Mercat, T. Gilles, N. Zoghby, G. Sandou, D. Beauvois, and G. Gil, "Multi-head attention for joint multi-modal vehicle motion forecasting," in *IEEE International Conference on Robotics and Automation*, 2020.

[114] A. Jain, S. Casas, R. Liao, Y. Xiong, S. Feng, S. Segal, and R. Urtasun, "Discrete residual flow for probabilistic pedestrian behavior prediction," in *Conference on Robot Learning*, 2020, pp. 407–419.

[115] S. H. Park, B. Kim, C. M. Kang, C. C. Chung, and J. W. Choi, "Sequence-to-sequence prediction of vehicle trajectory via lstm encoder-decoder architecture," in *2018 IEEE Intelligent Vehicles Symposium (IV).* IEEE, 2018, pp. 1672–1678.

[116] C. Choi, "Shared cross-modal trajectory prediction for autonomous driving," *arXiv preprint arXiv:2004.00202*, 2020.

[117] J. Li, H. Ma, Z. Zhang, and M. Tomizuka, "Social-wagdat: Interaction-aware trajectory prediction via wasserstein graph double-attention network," *arXiv preprint arXiv:2002.06241*, 2020.

[118] X. Huang, S. G. McGill, J. A. DeCastro, L. Fletcher, J. J. Leonard, B. C. Williams, and G. Rosman, "Diversitygan: Diversity-aware vehicle motion prediction via latent semantic sampling," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5089–5096, 2020.

[119] J. Kim, K. Jo, W. Lim, M. Lee, and M. Sunwoo, "Curvilinear-coordinate-based object and situation assessment for highly automated vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 3, pp. 1559–1575, 2015.

[120] C. Thiemann, M. Treiber, and A. Kesting, "Estimating acceleration and lane-changing dynamics from next generation simulation trajectory data," *Transportation Research Record*, vol. 2088, no. 1, pp. 90–101, 2008.

[121] Y. Xia, S. Wang, and R.-L. Sheu, "S-lemma with equality and its applications," *Mathematical Programming*, vol. 156, no. 1-2, pp. 513–547, 2016.