Adaptive Prediction and Planning for Safe and Effective Autonomous Vehicles

Vijay Govindarajan

Electrical Engineering and Computer Sciences
University of California, Berkeley

Technical Report No. UCB/EECS-2021-263
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2021/EECS-2021-263.html

December 17, 2021

Adaptive Prediction and Planning for Safe and Effective Autonomous Vehicles

by

Vijay Govindarajan

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Engineering - Electrical Engineering and Computer Sciences

in the

Graduate Division

of the

University of California, Berkeley

Fall 2021

Adaptive Prediction and Planning for Safe and Effective Autonomous Vehicles

Abstract

Adaptive Prediction and Planning for Safe and Effective Autonomous Vehicles

by

Vijay Govindarajan

Doctor of Philosophy in Engineering - Electrical Engineering and Computer Sciences

University of California, Berkeley

Professor Francesco Borrelli, Co-chair

Professor Trevor Darrell, Co-chair

Despite recent advances in and many potential benefits of autonomous vehicles (AVs), there still remain several challenges until wide-scale adoption. The largest gap is in trust, where both passengers and proximal road users need to feel comfortable using and sharing the road with AVs. To bridge this gap, both prediction and planning algorithms need to include high fidelity driver models. This will help AVs to integrate better in mixed roadways and behave in a manner that conforms to the expectations of human stakeholders, helping to improve trust and acceptance.

We investigate algorithmic frameworks that adapt to observed behaviors and balance route efficiency with safety. After demonstrating the benefits of context-aware, data-driven, multimodal predictions both for nominal and set-based trajectory prediction, we tackle the issue of handling prediction errors made online with a confidence-aware framework. In particular, a confidence threshold is adaptively updated online based on the consistency of behaviors with the corresponding predictions. This enables the AV to make efficient route progress when faced with a consistent target vehicle, but prioritize safety faced with an erratic target vehicle. In addition to the confidence-aware approach, we present an alternative approach that incorporates feedback policies in a stochastic model predictive control framework. Benefits in metrics of mobility, comfort, and efficiency demonstrate the advantages of adaptive feedback policies for multimodal predictions.

Contents

List of Figures

List of Tables

Acknowledgments

I would like to thank my advisors, Prof. Francesco Borrelli and Prof. Trevor Darrell, for their guidance and support during my PhD. They provided me with the intellectual freedom to explore various topics in learning and control, as well as insightful advice to help me make progress and succeed in my research efforts. I could not have completed the work in this dissertation without their optimistic, patient, and sage advising. I would also like to thank Prof. Alexandre Bayen, Prof. Anca Dragan, and Prof. Sanjit Seshia for serving on my qualifying exam and dissertation committees and providing valuable feedback on my work. Prof. Katherine Driggs-Campbell mentored me at the start of my PhD journey. In addition to helping me learn the ropes of conducting research, she exposed me to many of the challenges in driver modeling and control that culminated in this dissertation.

Working in the MPC Lab has been a pleasure thanks to the close-knit group of colleagues – Xu, Siddharth, Monimoy, Charlott, Ed, Eunhyek, Hotae, Hansung, Roya, Thomas, Tony, Yeojun, Eric, Greg, Jon, George, Grace, Nitin, Ugo, Jacopo, Jongsang – whom I've had the opportunity to interact with and learn from over the years. I'd like to thank Xu Shen, Siddharth Nair, and Ivo Batkovic for being amazing collaborators and coauthors. The ambitious projects in this dissertation bore fruit thanks to their contributions and complementary insights. In addition, Monimoy Bujarbaruah has been a great sounding board and source of support.

Finally, I owe my successful journey to my family. My parents, Shrinivas and Shobhana, and sister, Nithya, made numerous sacrifices to ensure that I had a great education. Without their encouragement and unconditional support, I would have not made it this far. I dedicate this dissertation to them.

Chapter 1

Introduction

1.1 Motivation

Autonomous vehicles (AVs) offer several potential benefits, including bringing down the number of traffic accidents, reducing the amount of time spent in traffic, and enhancing mobility for those who cannot drive. There has been significant research and development in the past decade, with vehicle fleets tested in cities like Phoenix, San Francisco, and Pittsburgh [28]. In some limited situations and geographic locations, AVs are able to drive without a human backup driver at the wheel [48].

Despite these advances, a widescale adoption of AV technology has not yet materialized. A primary reason for this, aside from technological gaps, is due to the lack of trust of AV technology. A survey conducted in 2020 by the Partnership for Automated Vehicle Education (PAVE) revealed 48% of Americans would not ride “in a taxi or ride-share vehicle that was being driven autonomously.” In addition, only 58% of participants believed that “safe AVs will be available in ten years”, showing the current lack of trust in AV technology [5]. This survey indicates that, even if AVs have significant societal benefits, these cannot be realized without acceptance and trust of the stakeholders of the technology.

These stakeholders are not limited to the passengers sitting inside the AVs, but also other road users like proximal pedestrians, cyclists, and human drivers. In these roadways, it is imperative that the AVs can integrate seamlessly into mixed traffic, balancing objectives of safety and efficiency and aligning with human expectations of normative driver behavior.

This is not an easy problem, however, due to the different strategies and characteristics of human-driven vehicles as compared to AVs. In particular, AVs act as agents that strictly follow traffic rules and seek to reach a destination in a way that optimizes an objective consisting of factors like travel time, passenger comfort, and fuel efficiency. This is in contrast to human agents who are flexible in following traffic rules and are boundedly rational, choosing actions that achieve the driving goal but may not be optimal. For example, humans exhibit driving behaviors like speeding, driving with high jerk, and tailgating that are not optimal in terms of objectives like travel time and fuel efficiency. This misalignment in driving behavior leads to cases where neither agent is able to correctly predict what the other will do, which can result in inefficiencies in traffic flow and potential accidents [61].

To bridge the gap, research has focused on human driver modeling. This involves making autonomous vehicles drive in a similar manner to human-driven vehicles, but also improving prediction of human-driven vehicles for more effective interactions.

With respect to the former point, recent work has sought to learn human behavior from demonstration thanks to the availability of large scale driving datasets. In [95], a driving model was applied to predict discrete actions at intersections (straight, stop, left turn, right turn) and also to predict angular velocity inputs for lane following based on the Berkeley DeepDrive Video Dataset. Similarly, in [6], driving demonstrations were used to train a driving model based on a processed scene representations from an onboard perception stack. It was determined that simply training on a large dataset (30 million instances) was not enough to ensure reliability in unseen scenarios. The generalization error was reduced by augmenting the training loss for regularization and synthesizing edge case examples (e. g., perturbations in lateral offset). Yet this demonstrates a key limitation of data-driven models – in particular, enumerating the edge cases that may arise in driving to reduce generalization error is extremely difficult.

Datasets have also been essential in prediction of other human agents on the road. Earlier studies modeled individual drivers over short time horizons, by observing driving behavior under modes like attention and distraction and constructing corresponding models [83]. For example, [71] developed a driver model based on convex Markov chains to capture the stochasticity of human drivers and enable probabilistic queries to be made about safety. Recently, large-scale prediction datasets, including [85, 13, 42], have focused on longer term motion prediction and provide annotated scene context paired with trajectories taken by actors in a variety of traffic scenarios. The advantage of these datasets is that complex, data-driven driver models can be learned without prespecified feature selection and/or manual tuning, allowing for more nuanced interpretation of semantic context. For example, datasets can be used to learn human driver reward functions via inverse reinforcement learning, which can be incorporated in interaction-aware planning (e. g., [73, 72]) and analyzed for robustness to reward misspecification using formal methods [70]. In contrast, classical methods often require specialized knowledge of feature selection and tuning. For example, traffic flow models like the Intelligent Driver Model [87] require knowledge of model parameters like maximum longitudinal acceleration, minimum spacing and time headway between vehicles, etc. Similarly, methods like Kalman Filters and reachable sets require detailed modeling of vehicle dynamics, as well as disturbance covariance or bound identification [7, 33].

Having a single model capture a variety of heterogenous human drivers and traffic contexts without imposing large uncertainty bounds is challenging. While data-driven methods can help by reducing prediction errors and better modeling of underlying probability distributions, they still are prone to make mistakes in new situations and cannot be blindly trusted in all cases. Therefore, addressing the tradeoff between precise and accurate predictions, as detailed in [26], is a key design parameter in the successful deployment of any prediction framework.

This tradeoff is especially relevant when coupled with control design of an AV operating in a mixed environment. There exists principled methods of incorporating uncertainties and disturbances, through approaches like robust and stochastic control [14]. As the uncertainty bounds grow, the feasible set of control behaviors for the AV starts to shrink and often the optimal solution is simply to slow down until the uncertainty is resolved. This can lead to overly timid, risk-averse AVs that do not conform to expectations of human agents, which exacerbates the aforementioned human-AV misalignment problem. So determining how to adjust uncertainty or incorporate adaptive policies based on observed behavior is vital to finding the sweet spot of safe yet effective AV behaviors.

1.2 Outline and Contributions

We look at how to address some of these issues in this dissertation. We offer the following contributions:

Chapter 2 looks at the problem of providing nominal multimodal predictions given candidate goals. The specific domain chosen is a parking lot, which features numerous interactions in a compact driving region. We describe the generation of a parking behavior dataset and then detail a two-stage prediction architecture to estimate intent (i. e., parking spot) and trajectory execution. This shows the benefit of representing human driver behavior with data-driven, multimodal predictions, as compared to traditional model-based approaches.[1]

Chapter 3 extends the results in Chapter 2 to set-based, multimodal predictions for which a continuous probability distribution over trajectories is generated. Using the nuScenes and Lyft Level 5 Prediction datasets, we demonstrate the benefits of context-aware, data-driven, multimodal predictions for predicting driver behaviors with improved log likelihood and improved set precision than traditional methods. We then explore how to incorporate such predictions in a confidence-aware framework, which can adapt uncertainty online based on prediction error. We demonstrate the benefits of this adaptive confidence approach for collision avoidance through simulated interactions with a target vehicle at a traffic intersection in the CARLA simulator.

Finally, Chapter 4 considers an alternative framework to handling uncertain predictions. Rather than adjusting uncertainty through adaptive confidence levels, a feedback policy approach is proposed to provide flexible behaviors that are conditioned on future measurements of target vehicle behavior. This approach reduces conservatism as compared to traditional methods, for which a single control input sequence must be chosen to satisfy all likely target vehicle behaviors. The benefits of the feedback policy approach are evaluated at a traffic intersection in the CARLA simulator. The results indicate that our approach can improve mobility, comfort, and efficiency metrics as compared to open-loop baseline methods.[2]

[1]This work was done jointly with Xu Shen and Ivo Batkovic and appeared in [82].
[2]This work was done jointly with Siddharth Nair and appeared in [59].

Chapter 2

Nominal Multimodal Predictions for Human Parking Behaviors

This chapter starts with the problem of how to generate nominal, multimodal predictions in the case where approximate goal knowledge is available. Here, by nominal, we mean that the prediction format is a collection of trajectories, each with a mode probability but no uncertainty parameters (e. g., covariance matrix) in continuous state space. The specific domain selected is a parking lot, due to the limited availability of datasets and the numerous interactions in close proximity. Based on a collected dataset of parking behaviors, we show that incorporating parking spot intent in predictions, even if estimated, can reduce metrics of trajectory prediction error. In addition, including semantic context can help in long-term prediction performance.[1]

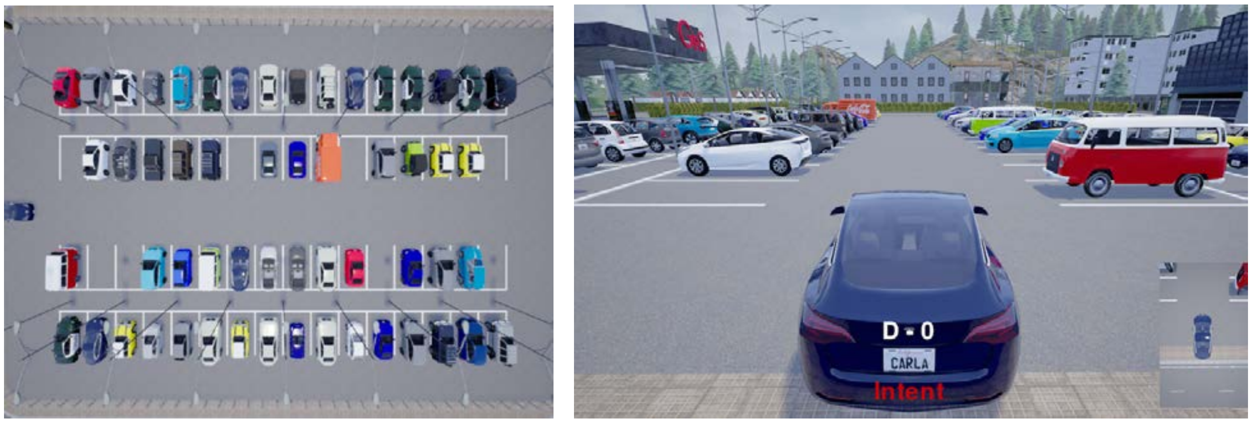
2.1 Introduction

While autonomous driving technologies have advanced by leaps and bounds, autonomous vehicles (AVs) still face great challenges. Robust perception [44], prediction [50], and interaction in real traffic scenarios with other participants [8, 27], especially human-driven vehicles, are difficult. Depending on the estimated behavior of the others, an AV's behavior may be too conservative, aggressive, or in the worst case, unsafe.

This problem is especially difficult in compact and unstructured domains like parking lots, which feature numerous interactions with human agents in close proximity [52] and may lead to congestion with suboptimal policies [81]. The potential to equip sensing and communicating infrastructure in these environments may help enable algorithmic solutions for connected AVs [36].

Extensive research has been conducted to investigate the problem of vehicle motion prediction, and algorithms are developed upon various level of accessible information.

[1]This work was done jointly with Xu Shen and Ivo Batkovic and appeared in [82].



(a) Bird's-eye view. (b) Driver view.

Figure 2.1: Parking lot scenario.

Physics-based methods use pose history to provide intuitive short-term predictions [41], where Kalman Filters [77] are used to propagate the vehicle dynamics forward in time and predict a trajectory or reachable set [4]. On the other side, data-driven methods like Recurrent Neural Networks (RNNs), especially Long Short-Term Memory (LSTM) networks, can learn a motion model without explicit knowledge of the vehicle parameters or input profile. The encoder of the LSTM processes a pose history and produces a summary of this sequence, which is then passed to a decoder for prediction [62, 46].

Since the vehicle motion is directly influenced by the intent, such as turning, lane keeping, and lane changing, embedding the intent information with the motion information can improve prediction performance [43]. The LSTM encoder learns a representation of the motion data to predict intent and generate interpretable multi-modal predictions [22]. For highway or urban driving, intent is usually classified based on lane graphs [84, 78, 31].

Not requiring explicit processing of trajectory or environment structure, Convolutional Neural Networks (CNNs) have been used in conjunction with LSTMs to synthesize information and make predictions from image sequences [23]. In recent work, these architectures have been applied to bird's-eye view (BEV) semantic images [16], including vehicle geometry, dynamics [20], and collision avoidance constraints for multimodal motion prediction [21].

Most existing studies discussed above have focused on structured environments, e.g., there exists a well defined road network [88] consisting of intersections, lanes, and traffic lights. However, for environments such as parking lots, the following challenges arise:

1. only limited public datasets are available of human-driven vehicles inside parking lots [91];

2. the parking maneuver is typically more complex [97] and challenging than highway driving;

3. the compact space and proximity among surrounding objects increases the risk of collision and congestion.

In this work, we focus on the problem of predicting a human driver's intended parking spot and future trajectory, given a set of features and levels of model abstraction. This chapter offers the following contributions:

1. we develop a parking lot simulation environment using the CARLA simulator and a racing wheel interface;
2. we collect an annotated dataset of human parking behaviors including both forward and reverse parking maneuvers, as well as various parking spot selections;
3. we propose a hierarchical LSTM model structure which can provide both intent and multi-modal trajectory predictions;
4. we propose a nested CNN-LSTM model structure with a visual encoder applied to semantic BEV images capturing parking lot geometry;
5. we compare a physics-based Kalman Filter baseline against the higher complexity LSTM and CNN-LSTM models to investigate the impact of model complexity, feature complexity, and amount of accessible information on prediction performance.

This chapter is organized as follows. Section 2.2 discusses the experimental design and dataset generation. Section 2.3 elaborates on the algorithms designed for intent classification and motion prediction. Section 2.4 discusses the results of the prediction algorithms. Finally, Section 2.5 summarizes our key findings and ideas for future work.

2.2 Experimental Design and Dataset

This section provides an overview of the parking lot simulation environment and experimental setup. We generated a dataset from human driving demonstrations consisting of trajectories, final parked spot location, and signaled intent. This dataset was then used for intent and motion prediction.

2.2.1 Parking Lot Scenario

In order to collect parking demonstrations in a controlled fashion, we used the CARLA simulator and CARLA ROS bridge [24] with a custom parking lot map modified from Town04. The parking lot consists of 4 rows with 16 spots each. In each trial, static vehicles were spawned into parking spots such that only 8 free spot options, located in the middle two rows, were available. The specific locations of these free spots were varied across trials to gather a diverse range of parking demonstrations from the subject. A free spot configuration example can be seen in Fig. 2.1a.

Given this setting, the subject was instructed to park into a free spot of their choosing, following a specified forward or reverse parking maneuver. When the subject selected the parking spot, he or she was instructed to press a button to signal a determined intent. The subject used a Logitech G27 racing wheel to control brake, throttle, and steering of the ego vehicle. In this experiment, only the ego vehicle was moving; all other vehicles in the scene remained static and parked. The driver view is visualized in in Fig. 2.1b.

A total of 10 subjects performed 30 forward parking and 30 reverse parking demonstrations, resulting in 600 total demonstrations. In each demonstration, the kinematic motion state history of the ego vehicle was recorded, as well as intent signals to know when a parking spot had been selected. In addition, the locations of each parking spot was collected. All demonstrations containing collisions or without intent signaling were filtered out. Furthermore, the configuration of the parking lot was recorded together with the bounding boxes of the ego vehicle and all other parked vehicles.

2.2.2 Dataset Generation

We first introduce some notation used for the data included in each demonstration.

- The vehicle pose at time t is denoted by $\vec{z}(t) = \begin{bmatrix} x(t) & y(t) & \theta(t) \end{bmatrix}^\top$. We assume the vehicle lies on the xy-plane and ignore variation in altitude, pitch, and roll.
- We denote the parking spot occupancy at time t as

$$\mathcal{O}(t) = \begin{bmatrix} x_{s,1}(t) & y_{s,1}(t) & f_{s,1}(t) \\ \vdots & \vdots & \vdots \\ x_{s,G}(t) & y_{s,G}(t) & f_{s,G}(t) \end{bmatrix},$$

  which is a matrix of size G by 3, where G is the number of parking spots. In each row j, the first two entries, $x_{s,j}(t), y_{s,j}(t)$ are the spot location and the last entry, $f_{s,j}(t)$, is a binary variable set to 1 if the spot is free.
- The ground truth distribution of the driver intent is denoted by $g(t)$, which is a one-hot vector with size $G + 1$. The $(G + 1)$-th element corresponds to undetermined intent. After the intent is signaled, $g(t)$ corresponds to the spot where the subject finally parks in.
- The parking bird's-eye view $I(t)$ is a semantic image of shape H by W by 3 representing the parking environment in Fig. 2.1a. The three channels correspond to the parking spot markings, static vehicle bounding boxes, and ego vehicle bounding box respectively.

For each demonstration, as shown in Fig. 2.2, the time intervals before the subject started driving and after the vehicle was parked were removed. Given a timestep $\Delta t = 0.1$s, the remaining portion of demonstrations were processed into short snippets with a history horizon $N_{\text{hist}} = 5$, and a prediction horizon $N_{\text{pred}} = 20$[2]. Each snippet was further processed to

[2]More details, as well as the processed dataset, can be found at `https://bit.ly/parkpredict`.

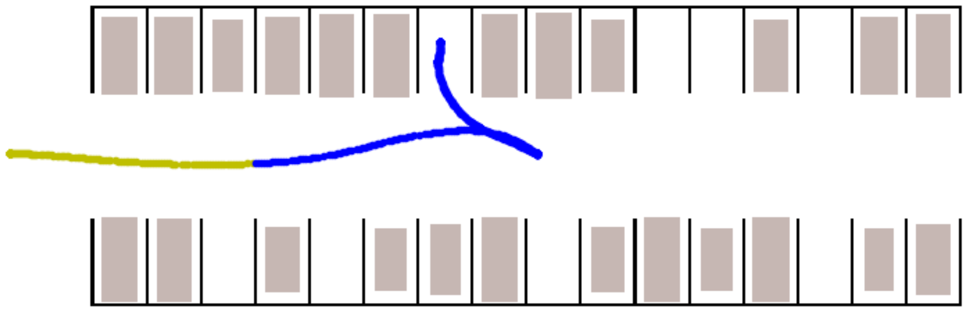


Figure 2.2: Sample parking demonstration focused on the middle two rows. The shaded bounding boxes represent parked vehicles. At first, the human driver has an undetermined intent (yellow section). Then the driver decides the spot intent and trajectory is shown in blue.

generate the following features:

1. motion history of $\vec{z}(t)$: $\mathcal{Z}_{\text{hist}}(t) \in \mathbb{R}^{N_{\text{hist}} \times 3}$;
2. parking spot and occupancy $\mathcal{O}(t) \in \mathbb{R}^{G \times 3}$;
3. image history of $I(t)$: $\mathcal{I}_{\text{hist}}(t) \in \mathbb{R}^{N_{\text{hist}} \times H \times W \times 3}$;

and labels:

1. future motion of $\vec{z}(t)$: $\mathcal{Z}_{\text{future}}(t) \in \mathbb{R}^{N_{\text{pred}} \times 3}$
2. ground truth driver intent: $g(t)$.

Note that all history features are sampled backward in time from t with horizon N_{pred} and step size Δt. Similarly, the prediction label is sampled forward in time from t with horizon N_{future} and step size Δt.

This processing resulted in a dataset $\mathcal{D} = \left\{ \left(\mathcal{Z}_{\text{hist}}^{(i)}, \mathcal{O}^{(i)}, \mathcal{I}_{\text{hist}}^{(i)} \right), \left(\mathcal{Z}_{\text{future}}^{(i)}, g^{(i)} \right) \right\}_{i=1}^{M}$. Here, the superscript (i) corresponds to the i-th dataset instance, and the total number of instances is $M = 20850$. The first tuple corresponds to a feature and the second tuple corresponds to a label, where we drop the time dependence of each entry going forward.

2.3 Methodology

Given the dataset $\mathcal{D}$, we break the prediction problem into intent and trajectory estimation subproblems. Using input history $\mathcal{X}_{\text{hist}}^{(i)}$ and occupancy $\mathcal{O}^{(i)}$, we generate the distribution of predicted driver intent $\hat{g}^{(i)}$ and the predicted future N_{pred}-step trajectory $\hat{\mathcal{Z}}_{\text{future}}^{(i)}$. We evaluate three models that vary in both model and feature complexity: an EKF baseline, an LSTM network, and a CNN-LSTM network. The following sections describe the model structures and input history features $\mathcal{X}_{\text{hist}}^{(i)}$ provided to each model.

2.3.1 Extended Kalman Filter (EKF) with Constant Velocity

As a baseline, we use an EKF with a constant velocity assumption for which $\mathcal{X}^{(i)}_{\text{hist}} = \left(\mathcal{Z}^{(i)}_{\text{hist}}\right)$. The following state dynamics and measurement model are used.

State Dynamics

$$\begin{bmatrix} x_{k+1} \\ y_{k+1} \\ \theta_{k+1} \\ v_{k+1} \\ \omega_{k+1} \end{bmatrix} = \begin{bmatrix} x_k + v_k \cos(\theta_k)\Delta t \\ y_k + v_k \sin(\theta_k)\Delta t \\ \theta_k + \omega_k \Delta t \\ v_k \\ \omega_k \end{bmatrix} + q_k, \;\; q_k \sim \mathcal{N}(0, Q)$$

Measurement Model

$$\hat{z}_k = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_k \\ y_k \\ \theta_k \\ v_k \\ \omega_k \end{bmatrix} + r_k, \;\; r_k \sim \mathcal{N}(0, R)$$

The disturbance covariance, Q, was estimated using the pose and velocity information provided in each demonstration. The noise covariance was chosen as $R = diag(1\text{e-}3, 1\text{e-}3, 1\text{e-}3)$, to reflect the ground truth pose measurement. We first run the time and measurement updates for the pose history $\mathcal{Z}_{\text{hist}}$ and extrapolate $\hat{\mathcal{Z}}^{(i)}_{future}$ by the time update. Then, $\hat{g}^{(i)}$ is estimated by assigning probability to free spots (i.e. using $\mathcal{O}^{(i)}$) based on the inverse of the Euclidean distance to the final predicted position. If a given spot exceeds a distance threshold (20 m), then the corresponding probability is added to the undetermined category, the $(G+1)$-th element of $\hat{g}^{(i)}$.

2.3.2 Long Short-Term Memory Network (LSTM)

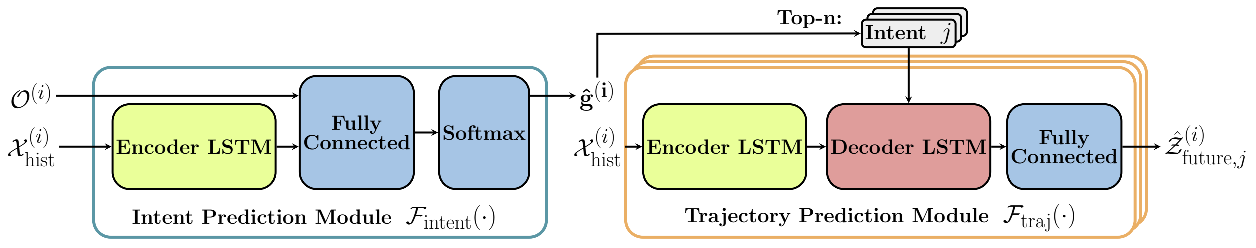

$\mathcal{O}^{(i)}$
$\mathcal{X}^{(i)}_{\text{hist}}$
Encoder LSTM
Fully Connected
Softmax
Intent Prediction Module $\mathcal{F}_{\text{intent}}(\cdot)$
$\hat{\mathbf{g}}^{(\mathbf{i})}$
Top-n:
Intent j
$\mathcal{X}^{(i)}_{\text{hist}}$
Encoder LSTM
Decoder LSTM
Fully Connected
Trajectory Prediction Module $\mathcal{F}_{\text{traj}}(\cdot)$
$\hat{\mathcal{Z}}^{(i)}_{\text{future},j}$

Figure 2.3: Multi-modal LSTM prediction model.

Our proposed LSTM model is shown in Fig. 2.3. As with the EKF, $\mathcal{X}^{(i)}_{\text{hist}} = \left(\mathcal{Z}^{(i)}_{\text{hist}}\right)$, but intent and trajectory estimation are addressed in the reversed order. In particular, unlike the EKF, the model predicts multimodal, intent-conditioned trajectories with the following hierarchical structure.

Intent Prediction Module

The intent prediction module takes $\mathcal{X}^{(i)}_{\text{hist}}$ and $\mathcal{O}^{(i)}$ as inputs and estimates the intent probability distribution:

$$\hat{g}^{(i)} = \mathcal{F}_{\text{intent}}(\mathcal{X}^{(i)}_{\text{hist}}, \mathcal{O}^{(i)})$$

The pose history and parking spot occupancy are first passed into the encoder LSTM stack and then processed by a fully connected layer. A softmax output layer produces the predicted intent distribution, $\hat{g}^{(i)}$.

The objective function we minimize for intent prediction has three components, where $j \in \{1, ..., G+1\}$ denotes the intent index.

- Cross entropy between prediction $\hat{g}^{(i)}$ and ground truth $g^{(i)}$ to drive the predicted distribution closer to the ground truth label:

$$J^{\text{intent}}_1 = -\sum_{j=1}^{G+1} g^{(i)}_j \log(\hat{g}^{(i)}_j).$$

- Negative entropy of the predicted distribution $\hat{g}^{(i)}$ to account for the stochasticity of the driver:

$$J^{\text{intent}}_2 = -\mathcal{H}(\hat{g}^{(i)}) = \sum_{j=1}^{G+1} \hat{g}^{(i)}_j \log(\hat{g}^{(i)}_j).$$

- Penalty of predicting already occupied parking spots:

$$J^{\text{intent}}_3 = \sum_{j=1}^{G} \max\{(\hat{g}^{(i)}_j - f^{(i)}_{s,j}), 0\}.$$

Therefore, the final objective function is

$$J^{\text{intent}} = J^{\text{intent}}_1 + J^{\text{intent}}_2 + J^{\text{intent}}_3.$$

Trajectory Prediction Module

The trajectory prediction module takes $\mathcal{X}^{(i)}_{\text{hist}}$ and the intent index $j \in \{1, ..., G+1\}$ as input and estimates the future trajectory for N_{pred} timesteps:

$$\hat{\mathcal{Z}}^{(i)}_{\text{future},j} = \mathcal{F}_{\text{traj}}(\mathcal{X}^{(i)}_{\text{hist}}, j).$$

This module works sequentially with the intent prediction module to generate multimodal predictions. The encoder first processes $\mathcal{X}^{(i)}_{\text{hist}}$. The encoder's final hidden state and cell state are used to initialize the decoder. Then, for each intent index j, the decoder and the subsequent fully connected layer return a predicted future trajectory $\hat{\mathcal{Z}}^{(i)}_{\text{future},j}$, which is associated with probability $\hat{g}^{(i)}_j$.

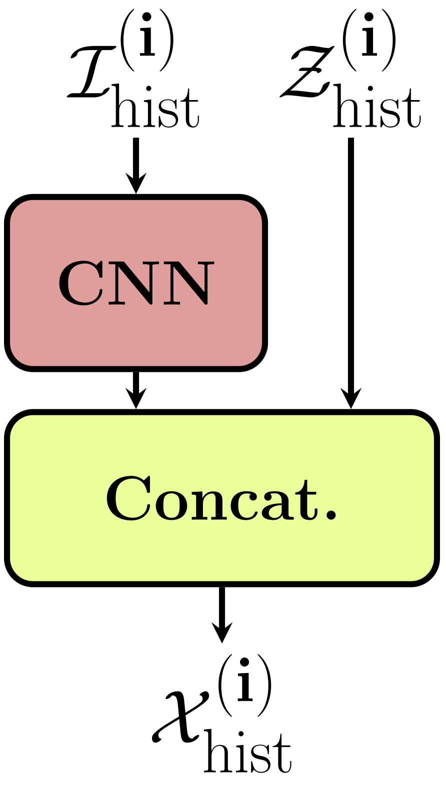
However, during training, we decouple this module with intent prediction and only use ground truth label $g^{(i)}$:

$$\hat{\mathcal{Z}}^{(i)}_{\text{future},gt} = \mathcal{F}_{\text{traj}}(\mathcal{X}^{(i)}_{\text{hist}}, \operatorname*{argmax}_j g^{(i)}_j).$$

The objective function J^{traj} is defined using mean squared error (MSE) on position. Concretely, let $\vec{z}^{(i)}_k$ and $\hat{\vec{z}}^{(i)}_k$ be the k-th row of $\mathcal{Z}^{(i)}_{\text{future}}$ and $\hat{\mathcal{Z}}^{(i)}_{\text{future}}$ respectively, then:

$$J^{\text{traj}} = \frac{1}{N_{\text{pred}}} \sum_{k=1}^{N_{\text{pred}}} \sqrt{(\vec{z}^{(i)}_k - \hat{\vec{z}}^{(i)}_k)^\top \text{diag}(1,1,0)(\vec{z}^{(i)}_k - \hat{\vec{z}}^{(i)}_k)}$$

2.3.3 Convolutional Neural Network LSTM (CNN-LSTM)


$\mathcal{I}^{(\mathbf{i})}_{\text{hist}}$
$\mathcal{Z}^{(\mathbf{i})}_{\text{hist}}$
CNN
Concat.
$\mathcal{X}^{(\mathbf{i})}_{\text{hist}}$

Type / Stride	Parameters
Conv / s1	$8 \times (3 \times 3)$
Max Pool / s2	Pool 2×2
Batch Norm	–
Conv / s2	$16 \times (3 \times 3)$
Max Pool / s2	Pool 2×2
Batch Norm	–
Flatten	–
Dropout	$p = 0.5$
Fully connect	128
Dropout	$p = 0.5$
Fully connect	16

Figure 2.4: CNN scene processing architecture. The table on the right describes the CNN preprocessing block in the left figure.

The CNN-LSTM model, like the LSTM model, uses the same structure in Fig. 2.3. However, as depicted in Fig. 2.4, a single CNN preprocesses each image in $\mathcal{I}^{(i)}_{\text{hist}}$. The generated image features are then concatenated with the motion features. Hence, for the CNN-LSTM model, $\mathcal{X}^{(i)}_{\text{hist}} = \left(\mathcal{F}_{\text{CNN}}(\mathcal{I}^{(i)}_{\text{hist}}), \mathcal{Z}^{(i)}_{\text{hist}}\right)$, where $\mathcal{F}_{\text{CNN}}(\cdot)$ represents the visual encoding operation performed by the CNN. This is inspired from the approaches used in [21, 96], which fuse motion and visual features with a LSTM temporal encoder.

2.3.4 Prediction Evaluation

To compare the performance of each prediction algorithm, we use 5-fold cross validation, where the LSTM and CNN-LSTM are trained for 200 epochs with batch size 32. The variable $\tilde{M}$ here corresponds to the cardinality of the corresponding hold-out set being evaluated.

1. Top-n Accuracy: Let the set $\mathcal{G}_n(\hat{g}^{(i)})$ include the n most likely intent categories in the predicted intent distribution, $\hat{g}^{(i)}$. Then, the top-n accuracy is:

$$\mathcal{A}_n = \tfrac{1}{\tilde{M}} \sum_{i=1}^{\tilde{M}} \sum_{j=1}^{G+1} \; g^{(i)}_j \; \mathbb{I}\left(g^{(i)}_j \in \mathcal{G}_n(\hat{g}^{(i)})\right)$$

   where $\mathbb{I}(\cdot)$ is the indicator function.

2. Mean Distance Error: Given a predicted $\hat{\mathcal{Z}}_{future}$ and actual trajectory $\mathcal{Z}_{future}$, we can look at the position error as a function of the prediction timestep. The mean distance error at timestep k, d_k, is:

$$d_k = \tfrac{1}{\tilde{M}} \sum_{i=1}^{\tilde{M}} \sqrt{(\vec{z}^{(i)}_k - \hat{\vec{z}}^{(i)}_k)^\top \; diag(1,1,0) \; (\vec{z}^{(i)}_k - \hat{\vec{z}}^{(i)}_k)}$$

2.4 Results

In this section, we compare the intent prediction capability and the trajectory prediction error of each algorithm using the selected evaluation metrics. We investigate the impact of information level and multimodality on the prediction performance. For brevity, CNN-LSTM is shortened to CNN in the subsequent figures.

2.4.1 Intent Classification

Fig. 2.5 shows the top-n accuracy $\mathcal{A}_n$ across all models evaluated. As expected, the LSTM and CNN-LSTM outperforms the EKF baseline at every n, achieving roughly 85% top-1 accuracy and nearly 100% top-3 or top-5 accuracy. This follows from the fact that the LSTM and CNN-LSTM are trained specifically on the intent prediction task, while a heuristic inverse distance approach is used after the trajectory prediction step of the EKF.

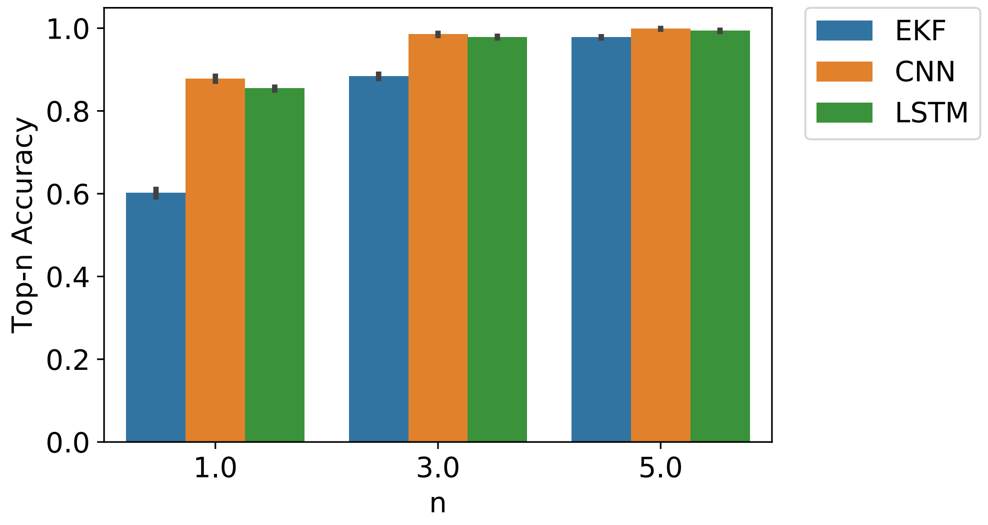


Figure 2.5: Top-n intent classification accuracy, $\mathcal{A}_n$.

2.4.2 Trajectory Prediction

In this section, we look at the following models and levels of information, where $\star \in \{\text{LSTM}, \text{CNN}\}$:

- $\star$**_no_intent**: intent-agnostic model where the trajectory module $\mathcal{F}_{\text{traj}}(\cdot)$ is re-trained and applied only zeroed intent input.
- $\star$**_gt_intent**: intent-conditioned model where $\mathcal{F}_{\text{traj}}(\cdot)$ is decoupled from intent module $\mathcal{F}_{\text{intent}}(\cdot)$, predicting $\hat{\mathcal{Z}}^{(i)}_{future,gt}$ only using ground truth intent $g^{(i)}$.
- $\star$**_multimodal**: intent-conditioned, multimodal model where $\mathcal{F}_{\text{traj}}(\cdot)$ is applied to the top-n entries of $\hat{g}^{(i)}$ from $\mathcal{F}_{\text{intent}}(\cdot)$. Here we select $n = 3$.

Fig. 2.6 captures the impact of intent knowledge and information level on the mean distance error, d_k, at each timestep k. For both the CNN-LSTM and LSTM models, we observe that they outperform the EKF for long-term predictions, as they learn a more nuanced motion model. For the LSTM model, the benefit of intent knowledge is minimal, as the no intent model is on par with the ground truth model. However, in the case of the CNN-LSTM, the additional knowledge of intent aids the prediction, as seen after time step 12. This suggests that the semantic BEV images can help the model to better understand the intent label for prediction.

Fig. 2.7 shows the benefit of multimodal predictions on the mean distance error, d_k, at each timestep k. Note that for the multimodal predictions, we take the top $n = 3$ intent labels and generate 3 separate rollouts. The results reported here are computed by finding the rollout nearest the ground truth trajectory (i.e. using mean squared error on position,

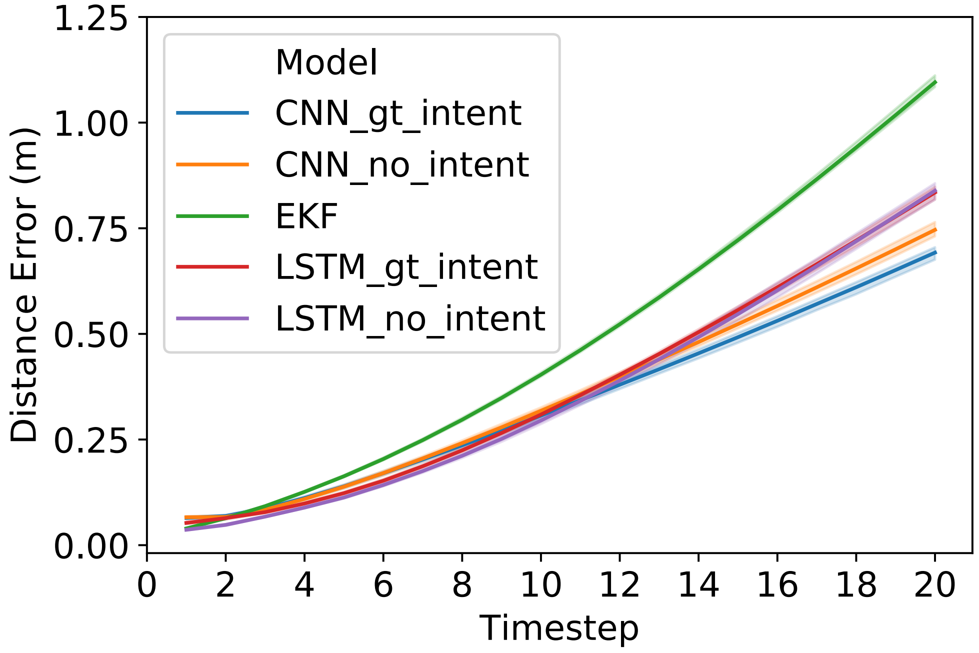


Figure 2.6: Mean distance error d_k vs. timestep k across varying models and levels of intent knowledge.

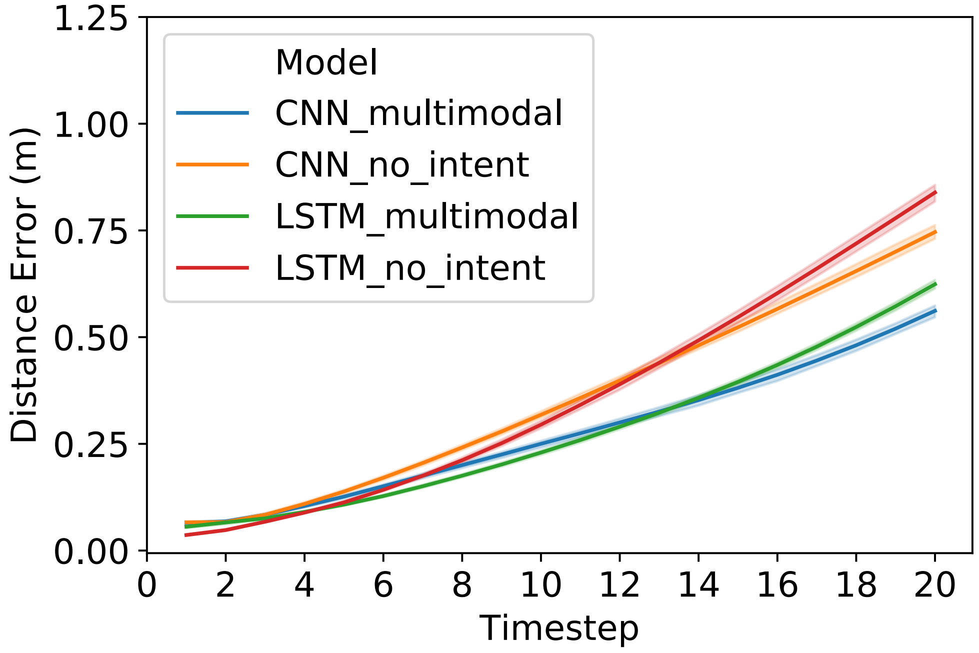


Figure 2.7: Mean distance error d_k vs. timestep k comparing intent-agnostic unimodal predictions to the best rollout among multimodal, intent-conditioned predictions.

J^{traj}) and then using this single rollout to evaluate d_k. We note that every intent-conditioned model does on par or outperforms the model without intent knowledge, over the prediction horizon. That shows that even if the intent is predicted and not known precisely, this additional information can still reduce trajectory prediction error compared to intent-agnostic predictions. This is likely due, in part, to the multimodal nature of the trajectory prediction, which can capture more evolution possibilities and driver model stochasticity relative to a unimodal trajectory prediction. We also observe that the CNN-LSTM models are well-suited for long-term predictions, for which the geometry of the parking lot is likely more informative.

2.4.3 Scenario Examples

We show how the prediction algorithms compare for a sample prediction instance in Fig. 2.8. The three sub-figures depict the 2-D layout of the parking lot, where the filled shaded boxes are occupied parking spots.

For intent prediction, the proximity-based heuristics of the EKF prioritizes the nearest spots in front of the vehicle, but misses the case that the vehicle may reverse into the ground truth spot backwards. Instead, the LSTM and CNN-LSTM capture the ground truth in top-3 candidates, meaning that they also learned from data that the driver might choose to back up in their maneuver.

For trajectory prediction, the EKF extrapolates the dynamics so it only offers single trajectory prediction with a significant delay. The LSTM and CNN-LSTM fit the ground truth better and offer multi-modal behaviors for other likely nearby spots. The top-3 candidates from the LSTM and CNN-LSTM are relatively more aware of the obstacles, as the LSTM better leverages the occupancy information and the CNN-LSTM learns a more detailed obstacle representation through semantic image inputs. Therefore, both models place more emphasis on the reverse maneuver.

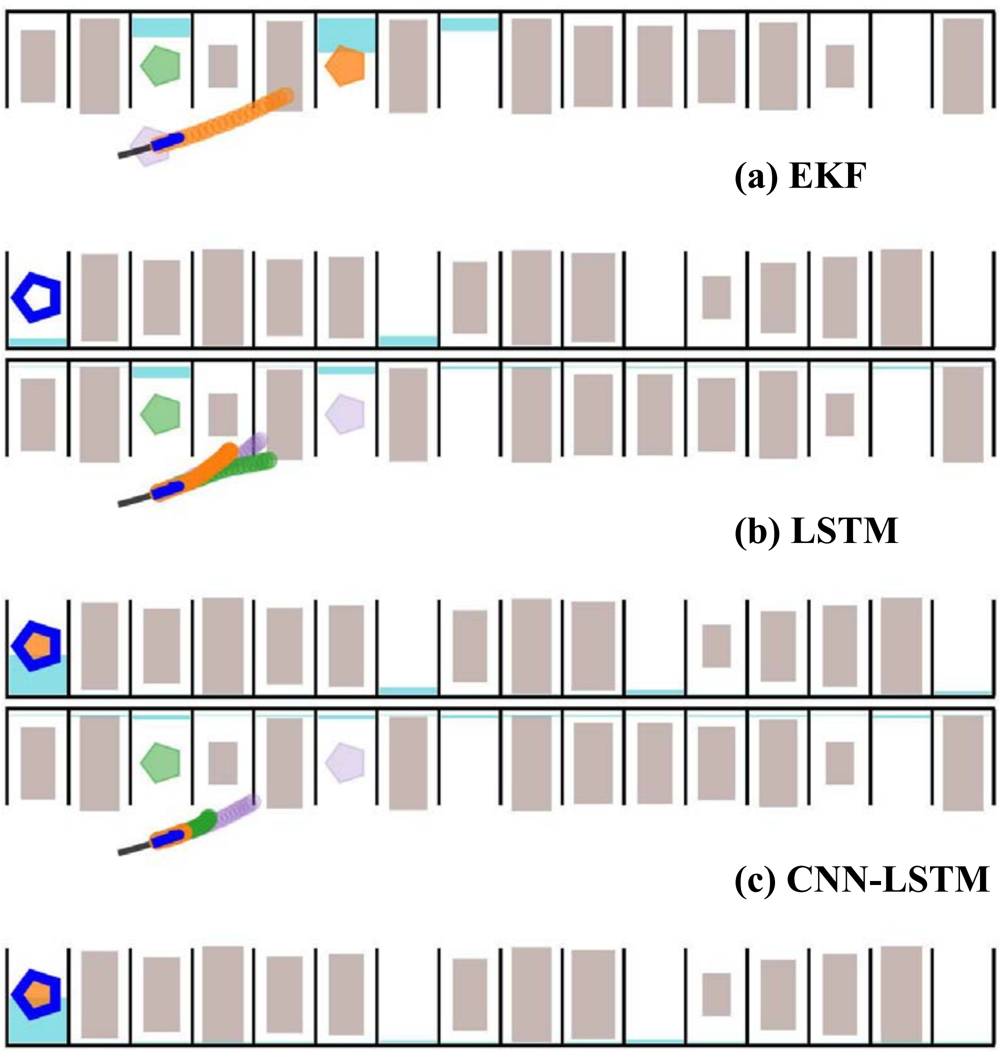


Figure 2.8: Prediction example across models. The black curve represents the pose history $\mathcal{Z}_{hist}^{(i)}$; the blue pentagon and curve stand for ground truth future intent $g^{(i)}$ and motion $\mathcal{Z}_{future}^{(i)}$ respectively; the orange, green, purple pentagons and curves correspond to the top-3 intent and trajectory predictions. Their order and transparency levels reflect the probability $\hat{g}_j^{(i)}$, which is visualized by the cyan bars in spots. When a pentagon is marked on the trajectory, it corresponds to the undetermined intent.

2.5 Discussion

This work investigated the problem of predicting a human driver's parking intent and maneuver with varying levels of feature and model complexity. This problem serves as a proxy for the general case of making nominal, multimodal predictions given well-defined, enumerated goals. A custom CARLA simulator parking lot environment was constructed and used to generate a dataset of human parking maneuvers. We compared an intent-conditioned LSTM and CNN-LSTM prediction model against an EKF baseline and noted the benefits of providing intent information for trajectory prediction, even if estimated. Additionally, by encoding obstacles and parking lot geometry, the semantic BEV images help improve prediction performance in the long term. The hierarchical framework is capable of offering multi-modal driver behavior prediction in a relatively unstructured environment like parking lots.

In the following chapter, we extend the prediction model for more general set-based multimodal predictions based on an underlying Gaussian mixture distribution. These architectures do not need explicit goals to be specified, allowing for a wider usage across traffic scenarios and context. We also consider the problem of how to incorporate these predictions in a collision avoidance framework and how to use the idea of model confidence to adapt conservatism given the wide variety of human drivers an AV may encounter.

Chapter 3

Confidence Aware Multimodal Predictions for Collision Avoidance

In the previous chapter, we demonstrated the advantages of intent-conditioned, multimodal predictions for predicting human parking behaviors. We now expand the setting of driving behaviors to more general traffic intersections, leveraging recent advances in large scale driving datasets. The key focus of this chapter is how to adjust conservatism of predictions based on observed behaviors, acting more conservatively around irrational drivers and less cautiously near rational drivers adhering to norms of driving behavior. This framework is applied to predictions represented as Gaussian mixture trajectory distributions, allowing for characterization of behavior consistency and continuous adjustment of conservatism based on confidence levels.

3.1 Introduction

Recent advances in motion prediction have been spurred on by the increased number of prediction datasets containing annotated scene context and trajectories of agents in a variety of traffic scenes [13, 42]. The state of the art models tend to be multimodal, data-driven (e. g., based on a trained neural network), and incorporate the scene context in making predictions. However, these models are not infallible. For example, the best performing models on the nuScenes prediction leaderboard (as of 2021) have a minimum average displacement error of over 1 m and a miss rate (which captures the event when all trajectory samples eventually deviate by 2 m or more from the ground truth) of approximately 40% [58]. This prediction error is computed over a test set that, in theory, is taken from the same distribution as the training set.

However, these prediction errors are exacerbated when trained models are deployed on scenarios that are not well-represented in the training set distribution. The environment may have changed – for example, an icy road can reduce the friction coefficient and thereby impact the vehicle's braking performance. Or norms may vary by city – for example, the acceptable set of behaviors for handling a vehicle-pedestrian interaction may not be the same in different regions [12]. There may be unmodeled causal factors not captured by the model – for example, one of the vehicles may be an ambulance with active sirens, leading nearby agents to pull over. Another issue is that the other agents themselves operate under stochastic and time-varying policies. Depending on factors like sobriety [3] and cognitive workload [53], even the same human driver can vary significantly in their behavior over time.

This leads to the central issue of how to consider these uncertain predictions for collision avoidance. Supposing that the prediction model generates a trajectory probability distribution, one approach is to opt for very low risk by choosing a very large confidence bound. Yet, this can lead to very conservative motion planning, where the ego vehicle slows down too often or is too timid to perform interactive maneuvers like turning in front of an oncoming vehicle. At the same time, fixing a low confidence bound results in planning that relies very heavily on the quality of the prediction made and may incur high risk when the target vehicle's behavior is inconsistent with the prediction.

This motivates the use of adaptive risk vs. conservatism selection in online deployment, where we can tune the confidence bounds based on how consistent the predictions are with the observed behavior. This idea of confidence-aware predictions was demonstrated in [30], where a boundedly rational human agent was modeled using a temperature parameter that could represent both rational and irrational behaviors. In this work, we extend the confidence-aware approach to more general trajectory distributions that can be expressed as using a Gaussian Mixture Model, including a subset of data-driven prediction architectures.

Our contributions are the following:

- we benchmark multiple prediction approaches on large scale traffic datasets to understand their ability to model normative behavior through both trajectory and set-based metrics;
- we extend the idea of confidence-aware predictions to Gaussian mixture trajectory distributions by maintaining an adaptive confidence threshold;
- we apply these confidence-aware, multimodal predictions to collision avoidance by generating occupancy sets parametrized by a confidence threshold;
- we demonstrate the benefits of the confidence-aware approach by evaluation in an simulated traffic intersection faced with a rational and irrational target vehicle.

The chapter is organized as follows. Section 3.2 provides an overview of motion prediction and confidence-aware prediction approaches. Sections 3.3 and 3.4 evaluate the benefits of data-driven, multimodal, and context-aware prediction models for normative behavior prediction on large-scale driving datasets. Section 3.5 then details how the base prediction model can be equipped with an adaptive confidence threshold and how to incorporate these predictions for collision avoidance. Sections 3.6 and 3.7 describe and provide the evaluation of the confidence-aware approach for motion planning versus fixed confidence baselines in a simulated CARLA traffic intersection. Finally, Section 3.8 summarizes the work and details avenues for future work.

3.2 Literature Review

We provide a brief overview of motion prediction methods and the use of confidence-aware predictions for adjustable conservatism in online deployment.

3.2.1 Motion Prediction

There exists a vast body of literature on human motion prediction, both in the context of driver models and general human-robot interaction. Broad coverage of these methods are provided in [51, 69]. The key axes of variation introduced by the surveys include (1) the specific modeling approach; (2) the level of scene information consumed by the model; and (3) the output format of the prediction.

The simplest models are physics-based and rely on an explicit dynamic model for extrapolation of observed motion (e. g., using a Kalman Filter). Pattern/maneuver-based models consider the higher level goals of an agent (e. g., turn left) before considering specific behavior execution (e. g., velocity and curvature profile) or rely on learned, data-driven models of behavior (e. g., by training a neural network on a trajectory prediction dataset). The most complex models are planning-based and consider joint interaction among agents in the scene, either given an explicit reward function or inferring the reward function through inverse reinforcement learning.

In terms of scene information, the lowest level of information is the prior pose trajectory of an agent. More contextual cues can be provided in terms of static context (e. g., lane centerlines, crosswalks) and dynamic context (e. g., traffic light states, bounding box information of nearby vehicles and pedestrians).

Finally, the prediction of a given agent can be a single trajectory, a collection of trajectory samples, a trajectory probability distribution, or a reachable set.

In this work, we limit our focus to physics and maneuver-based models and consider the impact of detailed scene context on prediction performance. We consider models that produce Gaussian trajectory distributions, such that multimodal models predict Gaussian mixture trajectory distributions.

3.2.2 Confidence-Aware Prediction

Despite the advances in motion prediction, there can be several scenarios in which the prediction is inconsistent with the future behavior of a target agent. Aside from potential bias of the prediction model, we may have incomplete information about the scene and not model some important causal factors (e.g. inclement weather and ensuing loss of visibility, animals crossing the road). Or perhaps the target agent could be behaving in a very uncommon way (e.g. drunk driving) that is in the heavy tail of behaviors. To handle such cases, a risk assessment based on model confidence can be used to determine whether the ego agent should behave more or less conservatively.

Risk can be thought of in terms of collision probability or in terms of unexpected behaviors relative to a prediction model [51]. The former approach seeks to provide some metric related to collision risk, either as a probability or through indicator metrics like time-to-collision. The latter approach, which is the focus here, seeks to detect unexpected behaviors relative to the prediction model. Observed behaviors that are inconsistent with predictions, i.e. that have low likelihood or violate modeling assumptions, reflect a low model confidence and increased risk. In contrast, behaviors that align well with the predictions reflect high model confidence and low risk.

Determining model confidence is intuitive for probabilistic models that provide likelihood distributions. In object tracking, for example, the innovation (or residual) likelihood reflects how consistent the measurement is with the prediction model. Online adaptation to a target agent can be performed by monitoring residual likelihood probabilities and applying a decision rule for scaling covariance matrices or switching between a set of process model hypotheses. In the context of driving, a target agent driving straight with constant velocity could be modeled with a simple motion model with low process noise, while a target agent turning with high lateral acceleration may involve a complex model with increased process noise [7].

Similarly, model confidence has been explored in the setting where humans are modeled as Boltzmann rational agents in a partially observable Markov decision process. The assumption is that the human is attempting to maximize utility. While the dynamics model is fully known, the reward function's structure is known but there is parametric uncertainty that must be estimated online. In particular, there may be multiple candidate goals or behavioral modes under which the human operates. In addition, even if the reward function were known exactly, the human agent may deviate from the corresponding optimal policy due to bounded rationality. By maintaining a Bayesian belief over a model confidence parameter, a trajectory probability distribution can be computed with adjustable conservatism online based on consistency of the demonstrated human behavior with the candidate set of reward functions. The use of model confidence, coupled with a safe motion planner, enables robotic motion planning that can adjust safety vs. efficiency by behaving more cautiously when faced with a less confident prediction model [30].

Rule-based methods also include a mechanism to tune conservatism based on model confidence. Rather than updating a continuous rationality parameter, a hybrid system formulation can be applied where each mode includes a varying number of motion constraints based on traffic rules and norms. For example, the most restrictive mode considers the full reachable set of the target agent, while less restrictive modes add on traffic rules like maintaining the speed limit or driving within lane boundaries. The conservativeness of the prediction can be adapted online simply by switching to the mode that best reflects the observed driving behavior, allowing both rule-following and rule-breaking agents to be appropriately modeled [49].

In this work, we extend the model confidence approach taken by [30] to general, Gaussian mixture trajectory distributions for which an explicit dynamics or reward model may not be available. We do this by considering the prediction residuals and adjusting a confidence threshold, as inspired by the classical methods in [7].

3.3 Normative Behavior Prediction Methodology

In this section, we first describe the elements required for normative behavior prediction, including the datasets, models, and metrics used.

3.3.1 Notation

We would like to predict the future $N_F \in \mathbb{Z}_{++}$ position states of target agents in a scene, given $N_H \in \mathbb{Z}_{++}$ timesteps of historical information and scene context information.

We define the following state and trajectory notation:

- $z_t^{(i)} = \begin{bmatrix} x_t^{(i)} & y_t^{(i)} & \theta_t^{(i)} \end{bmatrix}^\top \in \mathbb{R}^2 \times (-\pi, \pi]$ represents the pose of the i-th target agent at timestep t.

- $\mathcal{T}_{k|t}^{(i)} = \text{diag}(1, 1, 0) z_{k+t}^{(i)} \in \mathbb{R}^2$ represents the position of the i-th target agent in k timesteps after t.

- $\mathcal{T}_{t:t+N_F}^{(i)} = \{\mathcal{T}_{k|t}^{(i)}\}_{k=1}^{N_F} \in \mathbb{R}^{N_F \times 2}$ represents the N_F-step trajectory of the i-th agent starting at timestep $t+1$.

We express the prediction for the i-th target agent at timestep t as a Gaussian mixture:

$$\mathcal{G}_{t,M}^{(i)} = \left\{ \hat{p}_{t,j}^{(i)}, (\hat{\mu}_{k|t,j}^{(i)}, \hat{\Sigma}_{k|t,j}^{(i)})_{k=1}^{N_F} \right\}_{j=1}^{M}$$

where

- M is the number of modes.

- $\hat{p}_{t,j}^{(i)} \in [0, 1]$ is the probability of the j-th mode, assumed to be constant throughout the N_F-step prediction horizon.

- $\hat{\mu}_{k|t,j}^{(i)} \in \mathbb{R}^2$ and $\hat{\Sigma}_{k|t,j}^{(i)} \in \mathbb{S}_{++}^2$ are the mean and covariance, respectively, of a Gaussian distribution at timestep $k+t$ in mode j.

Analogous to the ground truth trajectory $\mathcal{T}_{t:t+N_F}^{(i)}$, we use the notation $\hat{\mu}_{t:t+N_F,j}^{(i)}$ to represent the predicted mean trajectory in mode j. As a shorthand, we use the notation $\hat{p}_t^{(i)}$ to represent the discrete probability distribution over the M modes.

Given a prediction $\mathcal{G}^{(i)}_{t,M}$ and the ground truth trajectory $\mathcal{T}^{(i)}_{t:t+N_F}$ for the i-th target agent, we can evaluate its probability as follows, where $\mathcal{N}$ represents the Gaussian PDF.

$$P(\mathcal{T}^{(i)}_{t:t+N_F}; \mathcal{G}^{(i)}_{t,M}) = \sum_{j=1}^{M} \hat{p}^{(i)}_{t,j} \prod_{k=1}^{N_F} \mathcal{N}(\mathcal{T}^{(i)}_{k|t}; \hat{\mu}^{(i)}_{k|t,j}, \hat{\Sigma}^{(i)}_{k|t,j})$$

For the purposes of collision avoidance, we may choose to prune unlikely modes and focus on a higher probability subset of modes. Let $I_m(\cdot)$ represent the operation that returns the index set corresponding to the top-m largest entries in a collection. Then we can generate a truncated Gaussian mixture, $\bar{\mathcal{G}}^{(i)}_{t,m}$, as follows:

$$\bar{\mathcal{G}}^{(i)}_{t,m} = \left\{ \bar{\hat{p}}^{(i)}_{t,j}, (\hat{\mu}^{(i)}_{k|t,j}, \hat{\Sigma}^{(i)}_{k|t,j})_{k=1}^{N_F} \right\}_{j \in I_m\left(\hat{p}^{(i)}_t\right)}$$

where $\bar{\hat{p}}^{(i)}_{t,j}$ are the normalized probabilities after truncation of the Gaussian mixture:

$$\bar{\hat{p}}^{(i)}_{t,j} = \frac{\hat{p}^{(i)}_{t,j}}{\sum\limits_{j \in I_m\left(\hat{p}^{(i)}_t\right)} \hat{p}^{(i)}_{t,j}}$$

Finally, we describe the uncertainty ellipsoid with confidence threshold $\beta \in \mathbb{R}_+$, mean $\mu \in \mathbb{R}^2$, and covariance $\Sigma \in \mathbb{S}^2_{++}$ as follows:

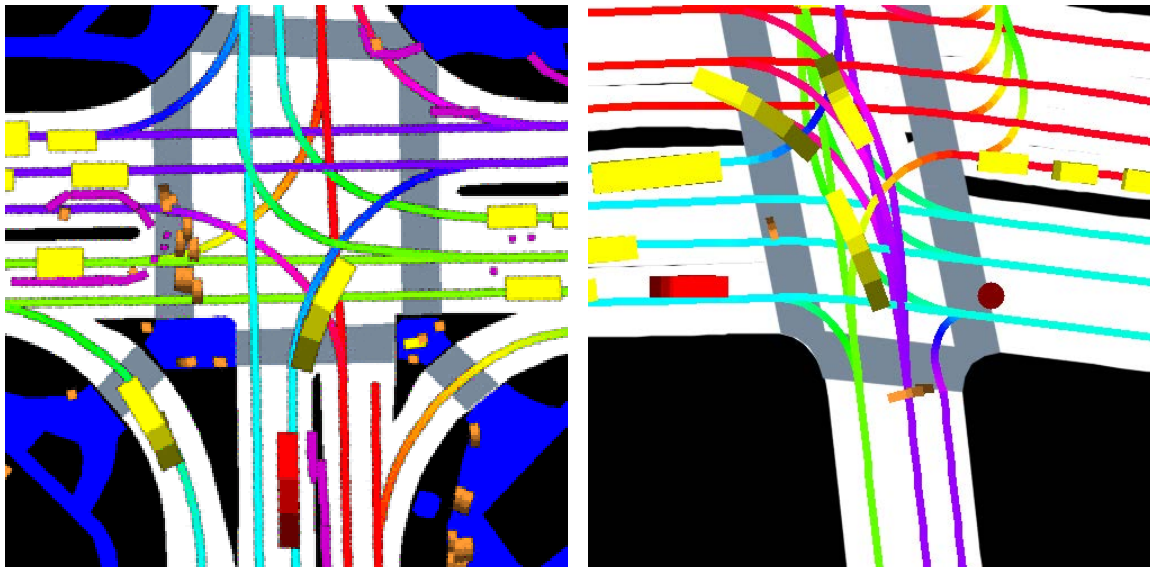
$$\mathcal{E}(\beta, \mu, \Sigma) = \{z \in \mathbb{R}^2 \colon (z - \mu)^\top \Sigma^{-1} (z - \mu) \leq \beta\}$$

3.3.2 Datasets

In this section, we describe two large-scale autonomous vehicle (AV) datasets used to train the prediction models.

nuScenes [13]

The nuScenes dataset includes sensor suite measurements (camera, lidar, GPS/IMU, and CAN) collected from AVs operating in Boston and Singapore. These measurements are paired with bounding box annotations for vehicles and pedestrians. We focus on the prediction challenge, which involves predicting the future 6-second motion of vehicle agents, given the past 1-second motion history and map context. The poses are annotated at approximately 2 Hz, so $N_H = 2$ and $N_F = 12$. The map context is represented as a rasterized image as depicted in Fig. 3.1a. Using the specified prediction challenge splits and removing any dataset instances with incomplete past/future motion information, this resulted in 29808, 7884, and 8379 dataset instances for training, validation, and test sets respectively.



(a) nuScenes [13] (b) Lyft Level 5 Prediction [42]

Figure 3.1: Scene context images. These include driveable regions, pedestrian walkways, lane centerlines, and bounding box motion history of nearby agents. The target vehicle is represented as a red rectangle.

Lyft Level 5 Prediction [42]

The Lyft Level 5 Prediction dataset is similar to the nuScenes dataset with some differences. First, the dataset is collected along a fixed 7-mile route in Palo Alto. Second, traffic lights (as observed by the AV perception stack) are annotated. Third, the dataset is larger with over 1000 hours of driving and annotations at 10 Hz. For prediction, the task was to predict 5 seconds of future motion given 1 second of past motion history and map context (Fig. 3.1b). Dataset instances were downsampled and chosen for inclusion according to the following procedure:

- Only the AV motion was considered for prediction, as object annotations for other agents were outputs of a perception stack that introduced label noise.
- Candidate instances were sampled at 1 Hz and removed if there was incomplete motion information.
- Instances with near-stationary future trajectories (i.e. under 1 meter in length) were pruned.
- Any instances remaining after the first three criteria were chosen via a stratified sampling procedure over future trajectories based on average velocity and lateral acceleration. For the training set only, the majority strata were undersampled to mitigate dataset imbalance.

The motion history was sampled at 5 Hz, so $N_H = 5$ and $N_F = 25$ for this dataset. This procedure resulted in 306186, 100032, and 100030 instances for training, validation, and test sets respectively.

3.3.3 Models

Given the prediction problem and normative behavior datasets, we turn to the models used to learn and predict normative behavior. We examine prediction performance as a function of the following factors:

1. **Model-based vs. data-driven:** Model-based predictions involve a explicit underlying dynamics model and/or control policy. In contrast, data-driven predictions (e. g., neural networks) need not have an explicit dynamics model or policy but can learn a more general function mapping from features to predicted trajectory.
2. **Unimodal vs. multimodal:** Unimodal predictions involve a single Gaussian distribution over trajectories ($\mathcal{G}^{(i)}_{t,M}$, where $M = 1$), whereas multimodal predictions are Gaussian mixture distributions over trajectories ($\mathcal{G}^{(i)}_{t,M}$, where $M > 1$).
3. **Context-agnostic vs. context-aware**: Context-agnostic predictions only use the target agent's observed motion, $\{z^{(i)}_{k|t}\}^{0}_{k=-N_H}$. Context-aware predictions, indicated with the suffix **-C**, additionally incorporate local scene context (i.e. nearby agents, lane information, traffic lights).

Model-Based Predictions

We choose to model the target agent with a set of discrete kinematic point-mass models with process noise. The base process model assumes constant acceleration and turn rate (CATR):

$$\begin{bmatrix} x_{k+1} \\ y_{k+1} \\ \theta_{k+1} \\ v_{k+1} \\ \omega_{k+1} \\ a_{k+1} \end{bmatrix} = \begin{bmatrix} x_k + v_k \cos\theta_k \Delta t \\ y_k + v_k \sin\theta_k \Delta t \\ \theta_k + \omega_k \Delta t \\ v_k + a_k \Delta t \\ \omega_k \\ a_k \end{bmatrix} + w_k$$

where, at timestep k, the pose is $(x_k,\, y_k,\, \theta_k)$; the speed is v_k; the angular velocity is ω_k, and the acceleration is a_k. $w_k \sim \mathcal{N}(0, Q)$ is zero-mean Gaussian process noise with covariance Q.

Other variants considered include constant acceleration and heading (CAH), constant velocity and turn rate (CVTR), and constant velocity and heading (CVH), which are reduced-order versions of the CATR model.

These motion models are each coupled with a simple pose measurement model with measurement noise:

$$o_k = \begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} + n_k$$

where, at timestep k, the measurement is o_k and $n_k \sim \mathcal{N}(0, R)$ is the associated zero-mean Gaussian measurement noise with covariance R.

1. **Extended Kalman Filter (EKF)**
   A single EKF (the specific motion model by dataset is described later) is used to filter and then extrapolate the observed target vehicle motion. The measurement noise covariance R is fixed, based on localization error distributions in [19], while the process noise covariance, Q, is identified by running Kalman smoothing and expectation maximization over the training dataset [68]. The prediction is a unimodal Gaussian trajectory.

2. **Static Multiple Model (SMM)**
   The SMM combines multiple, pretrained EKFs (based on CVH, CVTR, CAH, CATR motion models) in a filter bank to generate multimodal predictions. Each EKF motion model corresponds to a different mode, and the SMM assumes that the target agent is following only one of the motion models throughout the prediction horizon.

   Starting from a prior mode distribution, the residual likelihoods of each mode-matched filter is used to generate a posterior distribution over modes. This process is repeated recursively over the target agent's pose history to update the discrete mode distribution, where the initial distribution over modes is uniform [7]. The Gaussian trajectory distributions per mode are then generated by extrapolating each associated filter, resulting in a multimodal GMM prediction.

3. **Lane Follower with Context (LF-C)**
   Similar to the EKF model, the LF-C model filters the observed target vehicle motion. However, rather than extrapolating the motion via a constant input motion model, the LF-C incorporates the scene context and uses a lane-following control policy (with input noise) in an EKF framework to predict the future motion of the target agent. The assumption is that the target agent is trying to track one of a set of candidate lanes, subject to nearby agents in its vicinity. We restrict the observable scene context region around the target agent to 40 m in front, 10 m behind, and 25 m on the sides to mimic the target vehicle's limited sensing range.

   This implementation builds off the lane follower EKF formulation in [64, 63] to compute a lane prior distribution and generate corresponding trajectories. The control policies are implemented based on the Intelligent Driver Model (IDM) for acceleration [87] and a feedback/feedforward policy for curvature [45]. Finally, a lane posterior distribution is determined by computing a quadratic input cost, motivated by the reward function approach in [30]. The learnable parameters include covariance terms for the lane follower EKF and cost shaping terms that affect the lane prior and posterior distributions. Further details can be found in Appendix A.

The LF-C model returns a unimodal Gaussian trajectory corresponding to the lane with highest posterior probability.

4. **Multimodal Lane Follower with Context (LFM-C)**
The LFM-C model is essentially identical to the LF-C with the difference that it allows for multimodal predictions. In particular, we cap at a maximum of 16 possible lane trajectories to be considered as modes. The lane posterior distribution is paired with the lane trajectories to produce a multimodal GMM prediction.

Data-Driven Predictions

Given recent advances in neural-network based motion prediction, we opt to choose neural network predictors to generate data-driven predictions. In particular, we focus on the MultiPath architecture, which uses trajectory anchors to guide the network in predicting a GMM trajectory distribution and alleviate challenges with mode collapse during training [18].

First, we describe the feature representation used for all networks evaluated. A context image (e.g. in Fig. 3.1), which is a rasterized representation of the scene context in the same region considered by the LF-C and LFM-C models, is preprocessed by a ResNet50 network [38]. In parallel, the target vehicle's pose history is preprocessed by a Long Short-Term Memory (LSTM) network [40]. These image and motion features are then concatenated and passed through a fully connected network to provide prediction features for the subsequent trajectory prediction head networks. Further details on the network architecture and training procedure are provided in Appendix A.

The MultiPath architecture involves identification of anchor trajectories about which the multimodal predictions are generated. This was performed by clustering the training datasets using k-means clustering with a Euclidean distance metric over trajectories, as implemented in the tslearn library [86]. A fixed set of $M = 16$ anchors were determined and used for training the MultiPath models.

1. **Regression (REG)**
Given the aforementioned prediction features, the REG model applies a single linear mapping on the prediction features to generate $5 \times N_F$ parameters to produce a unimodal Gaussian trajectory distribution. In particular, for each prediction timestep, the model predicts the mean (2 XY coordinates) and covariance matrix eigendecomposition (i.e. 2 eigenvalues and 1 rotation angle). The REG model is context-agnostic so a blank image is used for the scene context. The model is trained using the log-likelihood of the actual trajectory given the predicted trajectory distribution.

2. **MultiPath (MP)**
The MP model extends the REG model to the multimodal case. Given a set of M

predetermined anchor trajectories, MP predicts $M \times (1 + 5 \times N_F)$ parameters. This is broken down as follows: M parameters correspond to the discrete mode probability distribution, while the remaining $M \times (5 \times N_F)$ parameters describe the trajectory distributions (similar to REG) for each of the M modes. The model is trained by considering the the log-likelihood of the actual trajectory and ground truth mode (determined based on L_2 norm distance with respect to the anchor trajectories), given the GMM predicted trajectory distribution. In this work, $M = 16$ modes were selected. Like the REG model, MP uses a blank image for the scene context.

3. **Regression with Context (REG-C)**
   Identical to the REG model but with the scene context image given without modification.

4. **MultiPath with Context (MP-C)**
   Identical to the MP model but with the scene context image given without modification.

3.3.4 Prediction Evaluation Metrics

To determine the performance of the prediction models, we evaluate them using trajectory-based and set-based metrics on the truncated Gaussian mixture $\bar{\mathcal{G}}^{(i)}_{t,m}$ for $m \in \{1, 3, 5\}$.

We compute the average displacement error as a building block for trajectory-based metrics. Given two trajectories, $\tau_{t:t+N_F}$ and $\tilde{\tau}_{t:t+N_F}$, the average displacement error is:

$$d(\tau_{t:t+N_F}, \tilde{\tau}_{t:t+N_F}) = \frac{1}{N_F} \sum_{k=1}^{N_F} \left\| \tau_{k|t} - \tilde{\tau}_{k|t} \right\|_2$$

A building block for evaluating mode classification is identification of the active mode index in the full Gaussian mixture, $\mathcal{G}^{(i)}_{t,M}$, which is determined using the average displacement error as follows:

$$j_{\text{active}}(\mathcal{G}^{(i)}_{t,M}, \mathcal{T}^{(i)}_{t:t+N_F}) = \underset{j \in \{1, \cdots, M\}}{\operatorname{argmin}} \; d(\mathcal{T}^{(i)}_{t:t+N_F}, \hat{\mu}^{(i)}_{t:t+N_F,j})$$

Finally, $\mathbb{1}_S$ is the indicator function with respect to set S:

$$\mathbb{1}_S(x) = \begin{cases} 1 & x \in S \\ 0 & x \notin S \end{cases}$$

The trajectory-based metrics include:

1. Log-likelihood:

$$\log \text{LL}(\bar{\mathcal{G}}^{(i)}_{t,m}, \mathcal{T}^{(i)}_{t:t+N_F}) = \log \left(\sum_{j \in I_m\left(\hat{p}^{(i)}_t\right)} \bar{\hat{p}}^{(i)}_{t,j} \prod_{k=1}^{N_F} \mathcal{N}(\mathcal{T}^{(i)}_{k|t}; \hat{\mu}^{(i)}_{k|t,j}, \hat{\Sigma}^{(i)}_{k|t,j}) \right)$$

2. Minimum Average Displacement Error:

$$\text{minADE}(\bar{\mathcal{G}}_{t,m}^{(i)}, \mathcal{T}_{t:t+N_F}^{(i)}) = \min_{j \in I_m\left(\hat{p}_t^{(i)}\right)} d(\mathcal{T}_{t:t+N_F}^{(i)}, \hat{\mu}_{t:t+N_F,j}^{(i)})$$

3. Minimum Final Displacement Error:

$$\text{minFDE}(\bar{\mathcal{G}}_{t,m}^{(i)}, \mathcal{T}_{t:t+N_F}^{(i)}) = \min_{j \in I_m\left(\hat{p}_t^{(i)}\right)} \left\| \mathcal{T}_{N_F|t}^{(i)} - \hat{\mu}_{N_F|t,j}^{(i)} \right\|_2$$

4. Top-m Accuracy:

$$A_m(\mathcal{G}_{t,M}^{(i)}, \mathcal{T}_{t:t+N_F}^{(i)}) = \mathbb{1}_{I_m\left(\hat{p}_t^{(i)}\right)} \left(j_{\text{active}}(\mathcal{G}_{t,M}^{(i)}, \mathcal{T}_{t:t+N_F}^{(i)}) \right)$$

Given the Gaussian mixture and a desired confidence threshold, we can also generate occupancy sets that approximate chance constraints of the actual Gaussian mixture probability distribution. Assuming we use a mode-invariant confidence threshold β, the predicted occupancy set of agent i at timestep $k+t$, $\Delta_{k|t}^{(i)}(\beta, \bar{\mathcal{G}}_{t,m}^{(i)})$, is the union of the corresponding uncertainty ellipsoids for each mode in $\bar{\mathcal{G}}_{t,m}^{(i)}$:

$$\Delta_{k|t}^{(i)}(\beta, \bar{\mathcal{G}}_{t,m}^{(i)}) = \bigcup_{j \in I_m\left(\hat{p}_t^{(i)}\right)} \mathcal{E}(\beta, \hat{\mu}_{k|t,j}^{(i)}, \hat{\Sigma}_{k|t,j}^{(i)})$$

The set evaluation metrics, parametrized by β, include:

1. Set Area:

$$\text{Area}(\bar{\mathcal{G}}_{t,m}^{(i)}, \mathcal{T}_{t:t+N_F}^{(i)}, \beta) = \lambda \left(\bigcup_{k=1}^{N_F} \Delta_{k|t}^{(i)}(\beta, \bar{\mathcal{G}}_{t,m}^{(i)}) \right)$$

2. Set Accuracy:

$$\mathcal{A}(\bar{\mathcal{G}}_{t,m}^{(i)}, \mathcal{T}_{t:t+N_F}^{(i)}, \beta) = \prod_{k=1}^{N_F} \mathbb{1}_{\Delta_{k|t}^{(i)}(\beta, \bar{\mathcal{G}}_{t,m}^{(i)})} (\mathcal{T}_{k|t}^{(i)})$$

Note that $\lambda(\cdot)$ is the Lebesgue measure, numerically computed over a 200 m by 200 m grid with resolution 10 cm. In addition, the definition of set accuracy means that all states of a trajectory must fall within the corresponding occupancy sets to be considered a correct prediction.

3.4 Normative Behavior Prediction Results

In this section, we present the evaluated trajectory and set metrics on the nuScenes and Lyft Level 5 Prediction datasets. We seek to understand how the aforementioned factors – model-based vs. data-driven, unimodal vs. multimodal, and context-agnostic vs. context-aware models – impact normative behavior prediction performance.

The trajectory metrics for the nuScenes and Lyft Level 5 Prediction datasets are summarized in Table 3.1 and Table 3.2, respectively. Similarly, the set metrics are visualized in Fig. 3.2 and Fig. 3.3 for both datasets.

3.4.1 Model-based vs. data-driven

The key benefits of the data-driven prediction models are best captured by the log likelihood trajectory metrics and the set area vs. accuracy plots. There is a noticeable increase in log-likelihood with the data-driven models, which is matched by the reduction in set size required for the same level of accuracy as model-based methods.

This indicates that the data-driven prediction models can more efficiently capture the trajectory probability distributions as compared to the model-based variants. In particular, the key advantages are the flexible ability to adjust means and covariances given the data instance, rather than in a predetermined manner based on the EKF model structure.

3.4.2 Unimodal vs. multimodal

In general, the unimodal prediction models outperform the multimodal models if the most likely mode is considered for evaluation (i.e. $m = 1$). The power of the multimodal models is apparent, however, as more modes are considered. In particular, when $m = 3$, there is a noticeable improvement in trajectory displacement metrics (minADE and minFDE).

In terms of log likelihood and set metrics, the story is a bit more nuanced and varies with the type of model and dataset considered. First, the benefits of considering more modes are less prominent when considering model-based approaches (especially the SMM) since the mode-level uncertainties are simply combined without reduction. In contrast, using the data-driven models, the mode-level uncertainties are learned jointly with the mode probabilities and mean trajectories, such that increasing the number of anchors/modes can help to reduce the mode-level uncertainties required for effective fitting. This is shown clearly in Fig. 3.2 and Fig. 3.3, where the black curve representing the unimodal set prediction is more efficient in set area in the model-based case but less efficient in the data-driven case.

3.4.3 Context-agnostic vs. context-aware

The context-aware methods are more efficient in terms of log likelihood and set area, indicating that the knowledge of scene context can help to reduce prediction uncertainty. For data-driven predictions, the context also helps to improve trajectory metrics like minADE and minFDE.

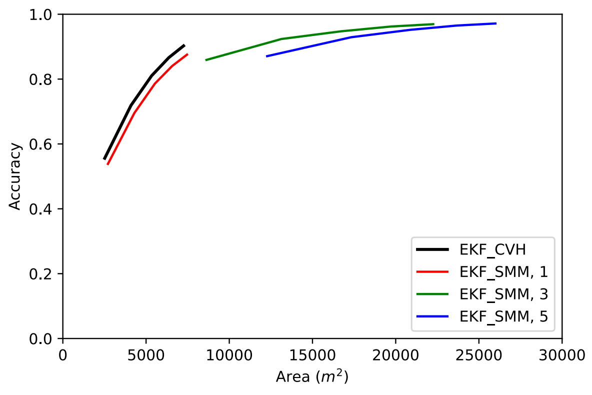
However, the model-based lane follower models (LFU-C and LFM-C) actually have higher trajectory prediction errors. This may be due to the simplistic modeling of joint interactions and use of rule-based methods (in particular, the Intelligent Driver Model). Without more expressive methods like inverse reinforcement learning, the model-based methods are too rigid to appropriately capture the driver policy over a long time horizon. In contrast, the data-driven methods are able to incorporate the scene context more effectively to improve the trajectory prediction as well.

Table 3.1: Trajectory metric evaluation on the nuScenes test set, where m is the number of modes considered. The constant velocity and heading (CVH) model was selected for the EKF result.

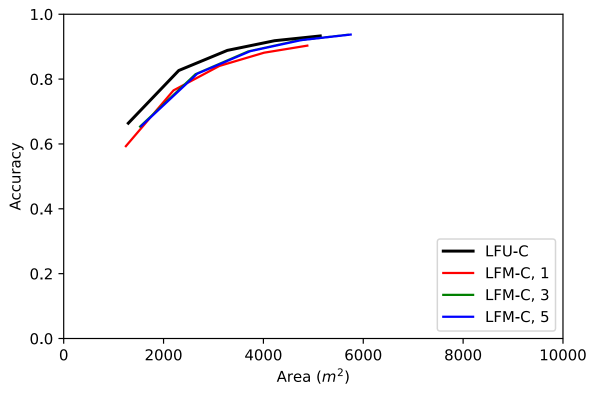
Model	**minADE (m)**			**minFDE (m)**			**Top-m Accuracy**			**log LL**		
	$m=1$	$m=3$	$m=5$	$m=1$	$m=3$	$m=5$	$m=1$	$m=3$	$m=5$	$m=1$	$m=3$	$m=5$
EKF	4.628	-	-	11.264	-	-	-	-	-	-71.02	-	-
SMM	5.896	3.798	3.717	14.323	9.188	8.979	0.348	0.832	1.000	-86.63	-66.82	-66.72
REG	4.715	-	-	11.579	-	-	-	-	-	-58.44	-	-
MP	4.571	2.645	2.026	10.717	6.126	4.452	0.378	0.758	0.918	-101.95	-58.15	-46.01
LFU-C	4.938	-	-	11.506	-	-	-	-	-	-59.34	-	-
LFM-C	5.016	4.324	4.304	11.725	9.558	9.503	0.731	0.993	0.999	-60.54	-56.77	-56.68
REG-C	4.090	-	-	9.656	-	-	-	-	-	-52.76	-	-
MP-C	4.497	2.454	1.872	10.165	5.431	3.879	0.371	0.782	0.942	-110.85	-55.71	-43.36

Table 3.2: Trajectory metric evaluation on the Lyft Level 5 Prediction test set, where m is the number of modes considered. The constant acceleration and heading (CAH) model was selected for the EKF result.

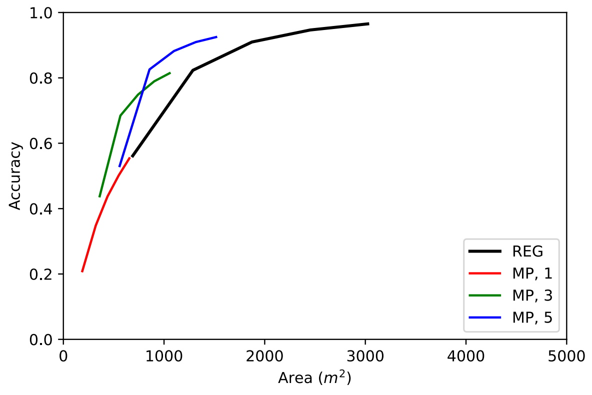
Model	**minADE (m)**			**minFDE (m)**			**Top-m Accuracy (%)**			**log LL**		
	$m=1$	$m=3$	$m=5$	$m=1$	$m=3$	$m=5$	$m=1$	$m=3$	$m=5$	$m=1$	$m=3$	$m=5$
EKF	1.753	-	-	5.059	-	-	-	-	-	-152.28	-	-
SMM	2.254	1.241	1.168	5.994	3.480	3.309	0.361	0.808	1.000	-143.77	-120.96	-120.84
REG	1.803	-	-	4.472	-	-	-	-	-	-83.79	-	-
MP	1.746	0.941	0.820	4.747	2.402	2.013	0.594	0.915	0.988	-103.94	-65.99	-62.16
LFU-C	3.038	-	-	7.351	-	-	-	-	-	-109.42	-	-
LFM-C	2.900	2.408	2.374	6.810	5.284	5.170	0.687	0.961	0.995	-118.96	-101.84	-100.05
REG-C	1.263	-	-	3.046	-	-	-	-	-	-68.62	-	-
MP-C	0.703	0.425	0.404	1.761	0.989	0.920	0.785	0.985	0.998	-69.35	-53.12	-51.74



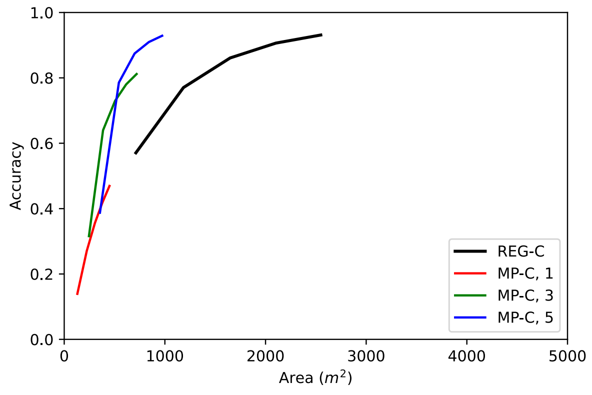
(a) Model-based, context-agnostic comparison.



(b) Model-based, context-aware comparison.

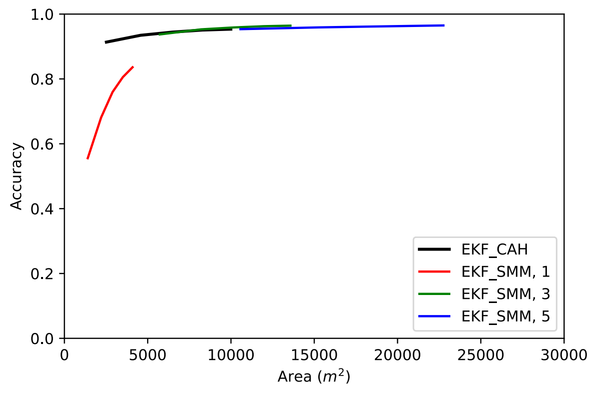


(c) Data-driven, context-agnostic comparison.

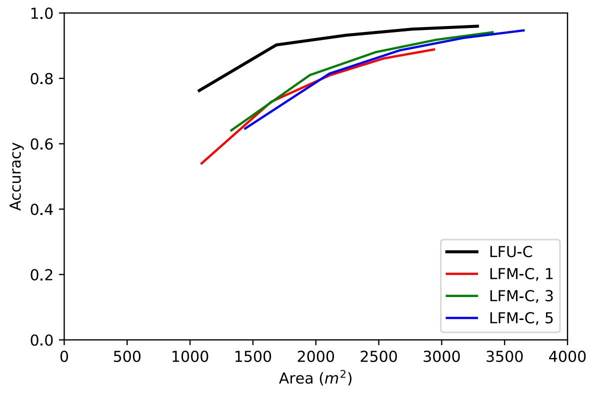


(d) Data-driven, context-aware comparison.

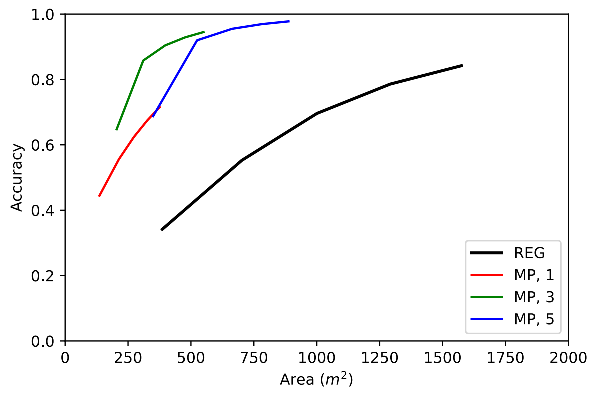
Figure 3.2: Set area vs. accuracy evaluated on the nuScenes test set for confidence threshold β varied from 2.0 to 8.0. The left column represents context-agnostic models, while the right column represents contains context-aware models. The black curves represent the unimodal model variant, while the red, green, and blue curves represent the multimodal model with the top $m = 1, 3, 5$ modes considered. The constant velocity and heading (CVH) motion model was selected for the EKF result based on its validation set performance.



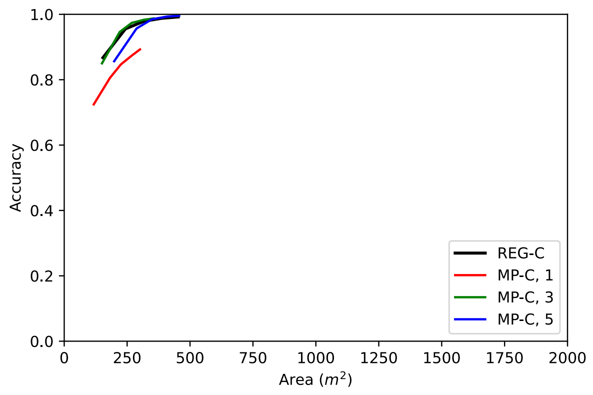
(a) Model-based, context-agnostic comparison.



(b) Model-based, context-aware comparison.



(c) Data-driven, context-agnostic comparison.



(d) Data-driven, context-aware comparison.

Figure 3.3: Set area vs. accuracy evaluated on the Lyft Level 5 Prediction test set for confidence threshold β varied from 2.0 to 8.0. The left column represents context-agnostic models, while the right column represents contains context-aware models. The black curves represent the unimodal model variant, while the red, green, and blue curves represent the multimodal model with the top $m = 1, 3, 5$ modes considered. The constant acceleration and heading (CAH) motion model was selected for the EKF result based on its validation set performance.

3.5 Confidence Aware Multimodal Predictions for Collision Avoidance

While we have shown the benefits of using context-aware, data-driven, multimodal prediction architectures like MultiPath (MP-C), there still remains nonzero prediction errors. Recent state-of-the-art prediction architectures have made further improvements by modeling joint vehicle interactions in social context ([74, 55]) or augmenting the training loss with geometric penalties for leaving the driveable region or opposing the lane direction of travel [11, 35]. However, even these improved methods are ultimately limited by the training dataset considered and may make larger errors in rare cases that are in the heavy tail of behaviors. Examples include drunk and drowsy drivers, who approximately follow their lane but with larger tracking errors and oscillatory control compared to alert drivers [3]. Such edge cases should be addressed by the prediction framework, but in a manner that doesn't lead to over-conservative driving (e.g. slamming on the brakes) in nominal cases where the other driver conforms with normative behavior.

In this section, we describe an confidence threshold adjustment approach that allows the prediction framework to adapt to drivers following normative behaviors as well as those who do not. This extends the notion of confidence aware predictions in [30] to Gaussian mixture predictions that may lack an explicit reward and dynamics model. We then describe how this framework can be used to generate occupancy sets for collision avoidance.

3.5.1 Adaptive Confidence Threshold Adjustment

When deploying the prediction architecture online, we would like to have an adaptive mechanism that allows us to adjust a risk tolerance (equivalently, a confidence threshold) based on how consistent the predictions are with the observed behavior. As demonstrated in [30], when using reward-based prediction models, modeling irrational and rational agents can be achieved by maintaining a model confidence parameter in a Bayesian framework.

However, in contrast to [30], we do not assume an underlying policy or dynamics model for the target agent. Therefore, in place of a Bayesian framework, we propose a method inspired by classical continuous adjustment methods used for target tracking [7] to adapt confidence thresholds online. The basic idea is to adaptively grow or shrink our confidence threshold so that previously experienced behavior can be captured efficiently by our set-based predictions.

As introduced in Section 3.3, we consider the truncated Gaussian mixture, $\bar{\mathcal{G}}_{t,m}$, representing the Gaussian mixture prediction with m modes made at timestep t. Given a confidence threshold β, we can generate a occupancy set at timestep t by considering the corresponding uncertainty ellipsoids $\mathcal{E}(\beta, \hat{\mu}_{k|t,j}, \hat{\Sigma}_{k|t,j})$ per mode j in $\bar{\mathcal{G}}_{t,m}$. Note that we are dropping the superscript (i) as we specialize our discussion to a single target vehicle for notational convenience; however, this framework is straightforward to generalize to several target vehicles.

We assume that the confidence threshold for the target vehicle is a parameter that varies slowly over time – the agent does not switch instantaneously between control policies. However, we must maintain a minimum confidence threshold for all timesteps. We also opt to have a mode-agnostic confidence threshold to mitigate potential issues with mode switching in the truncation process for the m modes.

Confidence Threshold Measurement

In order to update the confidence threshold online, we need a way to determine how consistent the predictions are with the observed behavior. At a given timestep, we consider a prediction made several timesteps prior and determine how well the subsequent target vehicle position measurements align with the older prediction in retrospect.

Concretely, let N_{sw} represent the number of timesteps we would like to consider in a sliding window manner. At timestep t, we want to see how well the prediction made at timestep $t - N_{\text{sw}}$ captured the behavior from $t + N_{\text{sw}} + 1, \cdots, t$ for which we have observed the target vehicle positions. This is done by computing the smallest confidence threshold, σ_t, required such that the prediction made at $t - N_{\text{sw}}$ fully captures all target vehicle positions in the corresponding occupancy sets for the sliding window horizon:

$$\begin{aligned} \sigma_t = \quad & \underset{s \in \mathbb{R}_{++}}{\text{argmin}} && s \\ & \text{subject to} && \mathcal{T}_{k|t-N_{\text{sw}}} \in \Delta_{k|t-N_{\text{sw}}}(s, \bar{\mathcal{G}}_{t-N_{\text{sw}},m}), \forall k \in \{1, \cdots, N_{\text{sw}}\} \end{aligned}$$

Confidence Threshold Update

Equipped with the confidence threshold measurement, we can now determine how an adaptive confidence threshold can be updated. Let $\beta_t \in \mathbb{R}_{++}$ represent the adaptive confidence threshold at timestep t. Our lower bound confidence threshold is $\beta_{\text{low}} \in \mathbb{R}_{++}$.

Starting from an initialization of β_0, the update is done using a low-pass filter with parameter $\alpha \in (0, 1)$:

$$\beta_{t+1} = \alpha \max(\sigma_t, \beta_{\text{low}}) + (1 - \alpha)\beta_t$$

In practice, the measurements σ_t are considered in a sliding window and the maximum of this window is used for the update. This adds some robustness by enforcing that the target vehicle must behave consistently with the predictions for a sustained period before the confidence threshold shrinks.

3.5.2 Occupancy Sets for Collision Avoidance

Given the choice of confidence threshold β_t and prediction $\bar{\mathcal{G}}_{t,m}$, we could generate the occupancy sets $\{\Delta_{k|t}(\beta_t, \bar{\mathcal{G}}_{t,m})\}_{k=1}^{N_F}$ as detailed in Section 3.3.4. However, while these sets represent the likely regions for the center of the target vehicle, they do not consider the dimensions of the target vehicle.

To address this issue, we estimate $\hat{\theta}_{k|t,j}$, the yaw angle of the target vehicle at timestep $t+k$ in a given mode j, by finite differences of the corresponding mean trajectory $\{\hat{\mu}_{k|t,j}\}_{k=1}^{N_F}$. The simplifying assumption being made is that the actual target vehicle pose varies only in translation with respect to the mean trajectory and not in orientation. With this assumption, we can define a collision avoidance set at timestep $t+k$ that incorporates the target vehicle's geometry as:

$$\mathcal{O}_{k|t}(\beta, \bar{\mathcal{G}}_{t,m}) = \bigcup_{j \in I_m(\hat{p}_t)} R(\hat{\theta}_{k|t,j}) B_{\text{TV}} + \mathcal{E}(\beta, \hat{\mu}_{k|t,j}, \hat{\Sigma}_{k|t,j})$$

where:

- $R(\theta) \in \mathbb{R}^{2\times 2}$ is a matrix that performs a rotation counterclockwise by the angle θ.
- B_{TV} is the polytope enclosing the target vehicle's footprint. In this case, it is an axis-aligned rectangle centered about the origin with side lengths equal to the vehicle's length and width.
- The operator $+$ here refers to the Minkowski sum and adds the uncertainty ellipsoid $\mathcal{E}(\beta, \cdot, \cdot)$ with the oriented target vehicle footprint.

In implementation, we choose to overapproximate the uncertainty ellipsoid with a tight bounding rectangle for faster computation of the Minkowski sum at the cost of slightly increased conservatism. The collision avoidance set $\mathcal{O}_{k|t}$ is thus a collection of polytopes that is amenable to a variety of motion planning approaches. We discuss our specific design to incorporate these collision avoidance constraints in motion planning in the following section.

3.6 Closed-Loop CARLA Simulation Design

In this section, we describe closed-loop simulations in CARLA [25] that evaluate the adaptive confidence threshold approach introduced in Section 3.5.1 for motion planning.

3.6.1 Scenarios

The ego vehicle travels through the intersection shown in Fig. 3.4, which is an unsignalized 3-way intersection with yield signs. At the same time, there is a target vehicle whose driving behavior the ego vehicle must predict and avoid. We consider two scenarios:

1. Both the ego vehicle and target vehicle must turn left at the intersection.
2. The ego vehicle proceeds straight but must yield to a left-turning target vehicle.

To ensure improved repeatability of our experiments, we use the synchronous mode of CARLA. This ensures that all processing (prediction, planning, and control) is complete

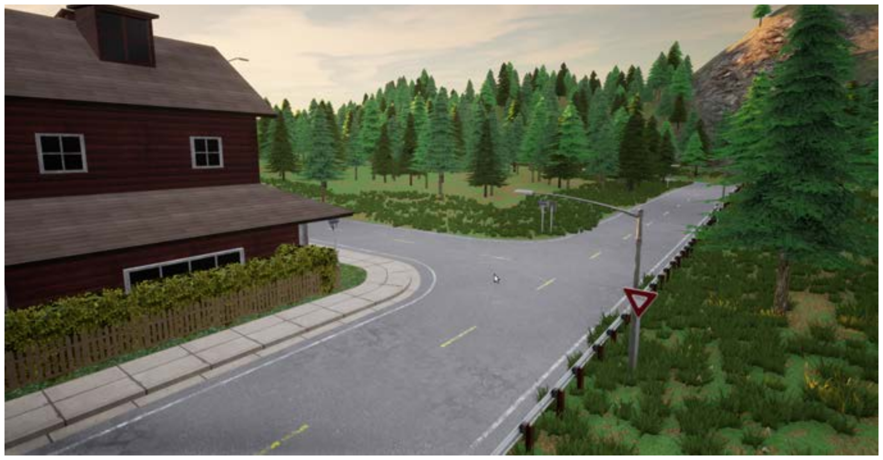


Figure 3.4: Unsignalized 3-way intersection in Town07 of CARLA used for evaluation.

before advancing the simulation. Thus, our results only consider the impact of the control inputs selected and not the time delays incurred in computing them[2].

3.6.2 Target Vehicle Policy

The key purpose of the adaptive confidence threshold adjustment introduced in Section 3.5.1 is to adjust conservatism as a function of the target vehicle's behavior. So we consider two behaviors – one that confirms to normative behaviors and is rational and another that has degraded lane tracking performance with large oscillations and is irrational.

The base model used for the target vehicle's control policy is a simple PI controller for longitudinal acceleration and a feedback/feedforward approach for the steering angle [45]. Low level actuation delays for throttle, brake, and steering subsystems are also configured.

The rational model involves setting low actuation delays and choosing speed, acceleration, and steering angle setpoints based on a preview of the upcoming trajectory. In particular, this model will slow down before turns to adhere to a comfortable lateral acceleration level and receives noise-free measurements of its tracking error performance.

In contrast, the irrational model behaves with degraded tracking performance. The design for this model was inspired by the NHTSA directives on detecting impaired, drunk drivers [3]. In particular, impaired drivers tend to have problems with maintaining proper lane position, weaving in the lane and deviating from lane boundaries. The longitudinal control is also impacted with varying speed tracking and sudden acceleration/deceleration. These factors are modeled by adding a sinusoidal lateral error and speed setpoint noise component to induce swerving and varying speed. In addition, the low level control delays are increased and the steering feedback is more myopic relative to the rational model. Details on this implementation can be found in Appendix B.

[2]All experiments were run on a computer with a Intel i9-9900K CPU, 32 GB RAM, and a RTX 2080 Ti GPU.

3.6.3 Ego Vehicle Policy

The ego vehicle is tasked with following its specified route while avoiding collision with the target vehicle. For simplicity, a lane-aligned Frenet frame (Fig. 3.5) is selected for the ego vehicle's control design. The coordinate s indicates the arclength (m) along the route, e_y is the lateral error (m) from the lane centerline, and e_ψ is the heading error (rad) with respect to the corresponding lane heading at s. Speed and curvature profiles for the lane centerline are computed offline and incorporate lateral acceleration constraints to limit speed while turning.

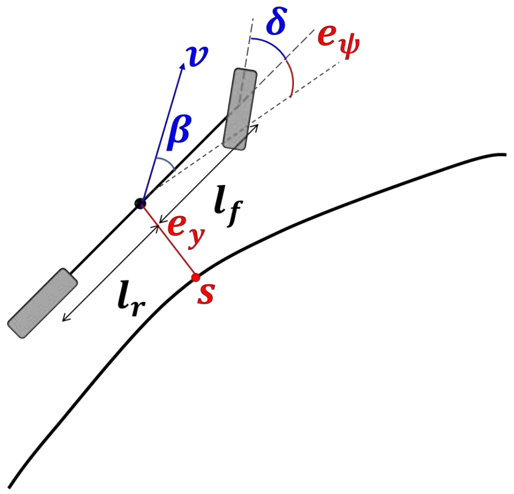


Figure 3.5: Lane-aligned Frenet frame and kinematic bicycle model used for ego vehicle control design.

Using the Frenet frame coordinates, the goal of lane tracking is to balance keeping lane (e_y and e_ψ) errors small while making route progress and considering comfort. Additionally, the ego vehicle should avoid collisions with the target vehicle by remaining outside of occupied s intervals. We pose the lane tracking problem in the context of model predictive control next.

Kinematic Bicycle Model in Frenet Frame

Since the scenarios involve low speed and low lateral acceleration driving, a kinematic bicycle model is chosen to model the ego vehicle system evolution [65]. Transformation from an inertial to Frenet frame results in the following system of differential equations.

$$
\begin{aligned}
\dot{s} &= \frac{v\cos(e_\psi + \beta)}{1 - \kappa(s)e_y} \\
\dot{e}_y &= v\sin(e_\psi + \beta) \\
\dot{e}_\psi &= \frac{v\sin(\beta)}{l_r} - \kappa(s)\dot{s} \\
\dot{v} &= a \\
\beta &= \arctan\left(\frac{l_r}{l_f + l_r}\tan(\delta)\right)
\end{aligned}
\tag{3.1}
$$

where:

- l_f and l_r are geometric wheelbase parameters of the vehicle,
- v is the vehicle's speed,
- a is the acceleration input,
- β in this context refers to the vehicle sideslip angle,
- δ is the front steering angle input,
- $\kappa(s)$ is the instantaneous curvature of the lane at s.

This model is discretized via forward Euler integration, resulting in the following discrete-time state, input, and model:

$$z_t^{\text{EV}} = \begin{bmatrix} s_t & e_{y,t} & e_{\psi,t} & v_t \end{bmatrix}^\top \tag{3.2}$$

$$u_t^{\text{EV}} = \begin{bmatrix} a_t & \delta_t \end{bmatrix}^\top \tag{3.3}$$

$$z_{t+1}^{\text{EV}} = f(z_t^{\text{EV}}, u_t^{\text{EV}}) \tag{3.4}$$

As in our prior notation, we consider the subscript $\star_{k|t}$ to indicate the value of $\star$ at timestep $t + k$. So $z_{0|t}^{\text{EV}} = z_t^{\text{EV}}$ is the current state of the ego vehicle at timestep t.

State and Input Constraints

There are actuation and actuation rate constraints on acceleration and steering angle. The corresponding feasible set of actuation inputs at timestep $t + k$ is denoted $\mathcal{U}_{k|t}$.

In terms of state constraints, there are lane and speed constraints, as well as the interval constraints. The lane constraints bound the lateral error and heading error to keep the vehicle within the lane and roughly aligned with the direction of travel. The speed constraints are set to avoid the vehicle from moving in reverse and from excessive speeding.

The interval constraints are where the collision avoidance sets $\mathcal{O}_{k|t}$ are considered. Let the length of the lane centerline be $s_{\max}$. Then the entire lane can be represented as an interval

$\mathcal{I}_{\text{lane}} = [0, s_{\max}]$, and the occupied portions of this lane interval are the regions where $\mathcal{O}_{k|t}$ intrudes within the lane width. The problem of collision avoidance is then cast as attempting to remain within the unoccupied portions of $\mathcal{I}_{\text{lane}}$ at each timestep in the planning horizon, similar to the method in [75]. A more detailed description of identifying these free lane interval regions is provided in Appendix B.

The feasible state region considering all lane, speed, and interval constraints is $\mathcal{Z}_{k|t}$.

Objective

The objective is to minimize the following tracking cost over a N-step horizon:

$$J(z_t^{\text{EV}}, u_{0|t}^{\text{EV}}, \cdots, u_{N-1|t}^{\text{EV}}) = \sum_{k=1}^{N} \left(\left\| z_{k|t}^{\text{EV}} - z_{k|t}^{\text{REF}} \right\|_Q^2 \right) + \sum_{k=0}^{N-1} \left(\left\| u_{k+1|t}^{\text{EV}} - u_{k|t}^{\text{EV}} \right\|_R^2 \right) \tag{3.5}$$

where the former term penalizes state tracking deviation and the latter term penalizes input rate (jerk, steering rate) for smooth actuation. $Q \in \mathbb{S}_+^2$ and $R \in \mathbb{S}_+^2$ are cost matrices. $z_{k|t}^{\text{REF}}$ is a reference state to track, where all state entries are zero except for the speed reference, which is generated by interpolating the precomputed speed profile for the lane centerline.

Lane Tracking MPC Problem

In summary, the lane tracking MPC problem is the following:

$$\begin{aligned}
\underset{u_{0|t}^{\text{EV}}, \cdots, u_{N-1|t}^{\text{EV}}}{\text{minimize}} \quad & J(z_t^{\text{EV}}, u_{0|t}^{\text{EV}}, \cdots, u_{N-1|t}^{\text{EV}}) && (3.6)\\
\text{subject to} \quad & z_{k+1|t}^{\text{EV}} = f(z_{k+1|t}^{\text{EV}}, u_{k+1|t}^{\text{EV}}) && (3.7)\\
& z_{k|t}^{\text{EV}} \in \mathcal{Z}_{k|t} && (3.8)\\
& u_{k|t}^{\text{EV}} \in \mathcal{U}_{k|t} && (3.9)\\
& z_{0|t}^{\text{EV}} = z_t^{\text{EV}} && (3.10)
\end{aligned}$$

where, at timestep t, the first input $u_{0|t}^{\text{EV}}$ is applied and the problem is resolved in a receding horizon manner. This is a nonlinear MPC problem, which is solved using IPOPT [92].

3.6.4 Manipulated Factors

For the two scenarios, we consider multiple joint rollouts of the ego vehicle and target vehicle to determine the performance of the adaptive confidence threshold approach for motion planning.

For the ego vehicle, the main factor is the choice of confidence threshold used to generate collision avoidance sets. We select three configurations:

1. **Low:** The confidence threshold β_t is set to a constant value of β_{low}. This corresponds to being more confident and optimistic in the prediction model's performance.

2. **Adaptive:** The confidence threshold β_t is time-varying, applying the adaptive approach in Section 3.5.1.

3. **High:** The confidence threshold β_t is set to a constant value of β_{high}. This corresponds to being more conservative and pessimistic in the prediction model's performance.

For the target vehicles, the two policies correspond to the two lane tracking models detailed in Section 3.6.2:

1. **Rational:** The target vehicle uses the well-tuned control policy to follow the lane with relatively low tracking error.

2. **Irrational:** The target vehicle uses the degraded control policy, mimicking an impaired driver, with oscillatory steering and speed control.

3.7 Closed-Loop CARLA Simulation Results

In this section, we demonstrate the benefits of the adaptive confidence threshold through scenario evaluation in CARLA. We show the impact of fixed vs adaptive confidence thresholds by visualizing the resultant trajectories followed by the ego vehicle. In the figures, the fixed confidence thresholds are $\beta_{\text{low}} = 0.211$ and $\beta_{\text{high}} = 9.21$, corresponding to the 10% and 99% confidence levels of the chi-square distribution with 2 degrees of freedom. The time-varying confidence threshold, β_t, uses the adaptive update detailed in Section 3.5.1 with initial value of $\beta_0 = 3.22$ corresponding to the 80% confidence level.

The scenario where the ego vehicle and target vehicle both make left turns is visualized in Fig. 3.6 and Fig. 3.7 for the rational and irrational target vehicle policies, respectively. The benefit of the adaptive approach is that ego vehicle is able to efficiently proceed through the left turn, similar to the β_{low} case, when the target vehicle is acting rationally and sticking to the left turn. However, when the target vehicle's behavior starts to degrade and oscillate both in speed and lateral position, the adaptive approach is able to approach the β_{high} confidence threshold and avoid a collision. The yielding scenario, as depicted in Fig. 3.8 and Fig. 3.9, also demonstrates a similar result. Using the adaptive policy helps to avoid a collision with the irrational target vehicle, but make faster route progress given the rational target vehicle behavior.

From this standpoint, the benefit of the adaptive confidence threshold approach is the ability to tune conservatism as a function of the observed target vehicle behavior, allowing for the safety vs. route efficiency tradeoff to be effectively managed as compared to fixed confidence threshold methods.

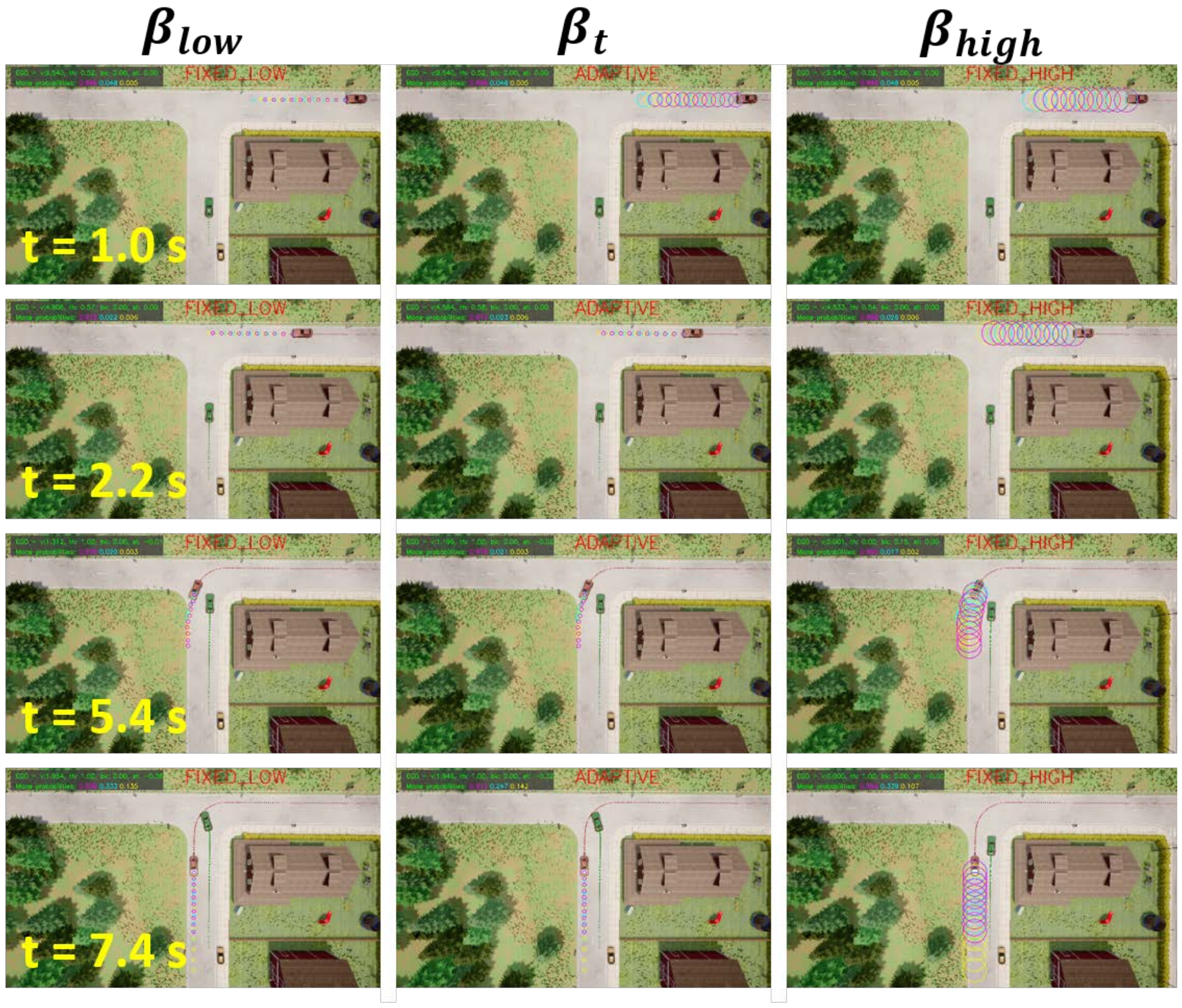


Figure 3.6: Joint left turn scenario with rational target vehicle (orange). The left and right columns show the actions taken by the ego vehicle (green) under fixed β values of β_{low} and β_{high}. In contrast, the middle column shows the time-varying, adaptive β_t that uses the approach in Section 3.5.1.
In this example, the ego vehicle with adaptive β_t is able to identify that the target vehicle's behavior is consistent with the prediction model, allowing the initial confidence threshold to be reduced by $t = 2.2\,\text{s}$ and approach β_{low}. This, in turn, allows the ego vehicle to stop braking sooner and proceed with the left turn earlier than the β_{high} case at $t = 7.4\,\text{s}$.

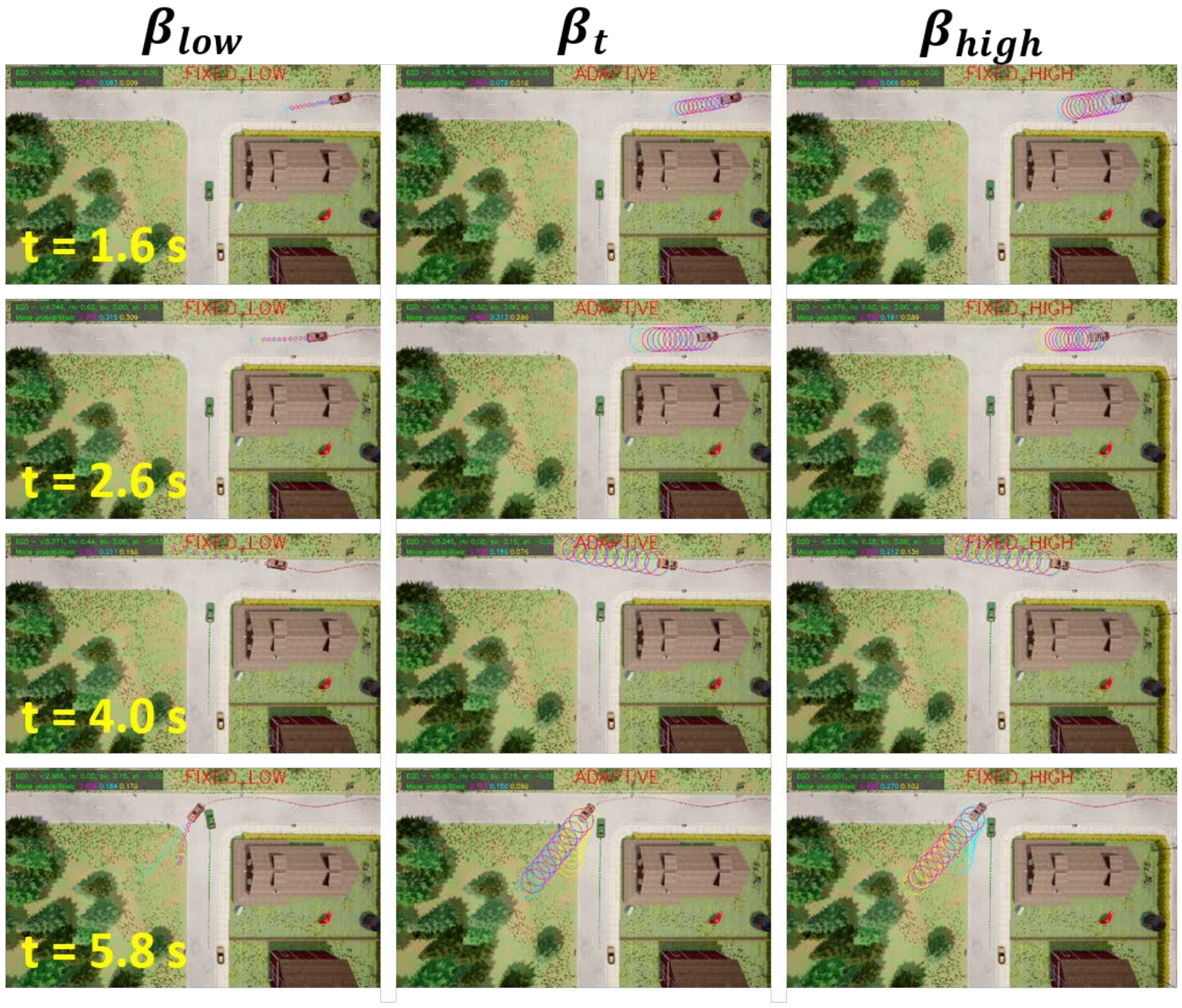


Figure 3.7: Joint left turn scenario with irrational target vehicle (orange). The left and right columns show the actions taken by the ego vehicle (green) under fixed β values of β_{low} and β_{high}. In contrast, the middle column shows the time-varying, adaptive β_t that uses the approach in Section 3.5.1.
In this example, the ego vehicle with adaptive β_t is able to identify that the target vehicle's behavior is not consistent with the prediction model. Therefore, the initial confidence threshold is increased by $t = 2.6\,\text{s}$, approaching β_{high}. This, in turn, allows the ego vehicle to slow down earlier and avoid a collision, in contrast to the β_{low} case at $t = 5.8\,\text{s}$.

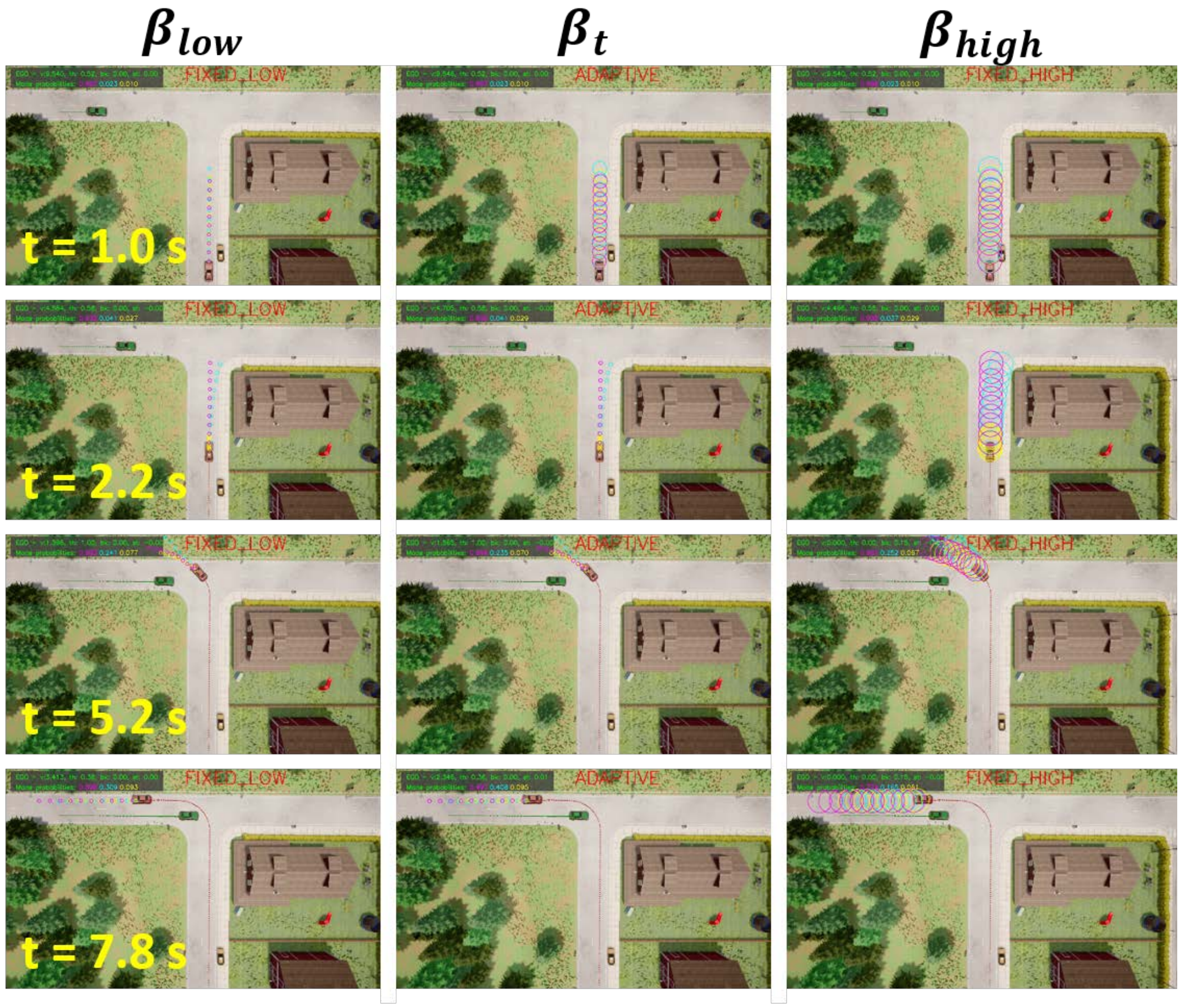


Figure 3.8: Yielding scenario with rational target vehicle (orange). The left and right columns show the actions taken by the ego vehicle (green) under fixed β values of β_{low} and β_{high}. In contrast, the middle column shows the time-varying, adaptive β_t that uses the approach in Section 3.5.1.
In this example, the ego vehicle with adaptive β_t is able to identify that the target vehicle's behavior is consistent with the prediction model. Therefore, the initial confidence threshold decreases by $t = 2.2\,\text{s}$, approaching β_{low}. The ego vehicle is able to slow down but avoid coming to a full stop at $t = 5.2\,\text{s}$, in comparison to the β_{high} case. In addition, under the adaptive approach, the ego vehicle can continue driving straight while the target vehicle is turning and thus makes further progress than the β_{high} case at $t = 7.8\,\text{s}$.

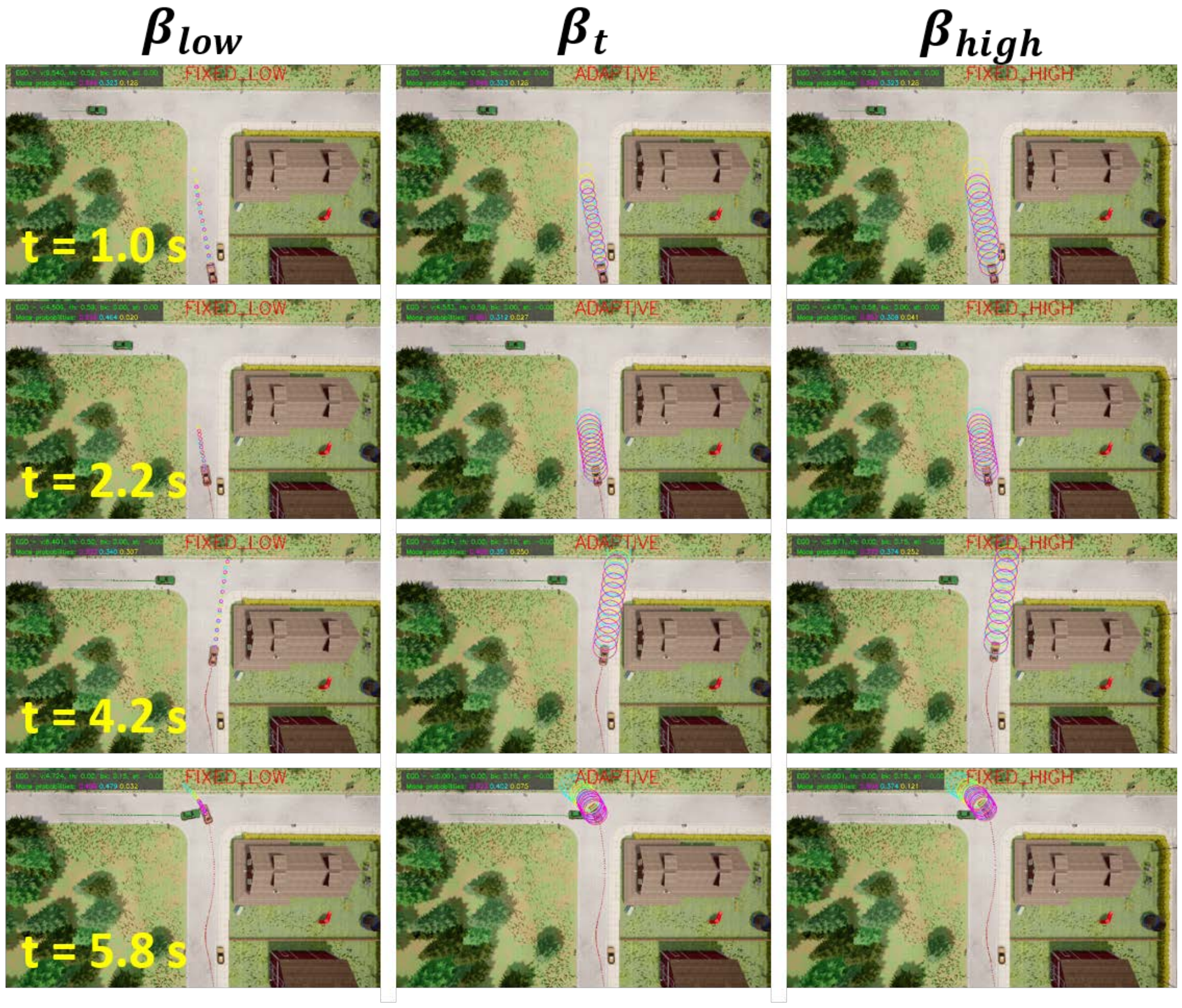


Figure 3.9: Yielding scenario with irrational target vehicle (orange). The left and right columns show the actions taken by the ego vehicle (green) under fixed β values of β_{low} and β_{high}. In contrast, the middle column shows the time-varying, adaptive β_t that uses the approach in Section 3.5.1.
In this example, the ego vehicle with adaptive β_t is able to identify that the target vehicle's behavior is not consistent with the prediction model. Therefore, the initial confidence threshold increases by $t = 2.2\,\text{s}$, approaching β_{high}. As a result, the adaptive approach enables the ego vehicle to escape a near collision that the β_{low} approach cannot at $t = 5.8\,\text{s}$.

3.8 Discussion

This chapter looked at the problem of extending confidence-aware prediction approaches to general Gaussian mixture trajectory distributions, including those predicted by recent data-driven, multimodal architectures. This combines the benefit of reduced prediction error and improved log likelihood of the data-driven predictions with the flexibility of confidence-aware predictions to handle both normative and anomalous driver behaviors.

First, a variety of prediction models varying in model structure (data-driven or model-based, unimodal or multimodal) and context awareness (context-aware or context-agnostic) were evaluated for normative behavior prediction performance on the nuScenes and Lyft Level 5 Prediction datasets. The main benefits of context-aware, data-driven, and multimodal predictions were in improved log likelihood and set area vs. accuracy, suggesting improved ability to capture actual behavior efficiently with reduced mode-level uncertainty.

Second, the problem of handling prediction errors online was addressed by introducing an adaptive confidence threshold update to adjust conservatism online as a function of past observed prediction performance. This confidence-aware approach was then applied in motion planning for collision avoidance at an intersection in CARLA. The key observation was the ability of the adaptive confidence threshold approach to prioritize efficiency when faced with a typical, rational driver but increase conservatism and avoid collision when faced with an atypical, irrational driver.

There are many areas of future improvement in this work. The use of a confidence threshold update is heuristic, with tuning parameters based on how large of a sliding window (N_{sw}) is considered for determining prediction consistency and how quickly to update the threshold (α). One improvement could be to combine this heuristic method with reachability analysis [33] to provide short-horizon safety guarantees while using the confidence threshold for longer-term collision avoidance. Recent advances in computing trajectory-parametrized reachable sets [89] and warm-start reachability analysis [39] may help enable online reachable set computation with varying initial conditions (e. g., vehicle speed and yawrate) and disturbances (e. g., actuation limits for a bus versus a racecar) under nonlinear vehicle dynamics.

In addition, scaling the confidence aware approach for joint vehicle predictions is another interesting extension. In data-driven, interaction-aware prediction models (e. g., [74]), the behavior of a single vehicle can impact other nearby vehicles and pedestrians, meaning that maintaining a separate, independent confidence threshold per agent may not reflect the interactive problem nature well. Determining how exactly anomalous agents impact the joint prediction will be vital for using such prediction approaches in online motion planning.

In terms of the control approach, we presented a simple lane tracking MPC formulation with collision avoidance incorporated as interval constraints. However, this approach can be overconservative since a single input sequence must robustly satisfy the interval constraints, no matter which actual mode or behavior the target vehicle actually follows. In the following chapter, we investigate an alternative approach that leverages mode-varying feedback policies to enable more flexible responses to target vehicle behavior observed online.

Chapter 4

Stochastic Model Predictive Control with Multimodal Predictions

In the previous chapter, we explored the benefits of context-aware, data-driven, multimodal predictions for trajectory and set predictions. We also proposed a heuristic collision avoidance method that flexibly adjusts a confidence threshold parameter online to improve closed-loop motion planning given the wide variety of drivers encountered on the road. In this chapter, we propose a more principled approach to the problem that incorporates the predictions directly in a stochastic model predictive control framework. In particular, by using feedback policies over the target vehicle's observed state, the resultant stochastic model predictive control approach can more flexibly and efficiently adapt to target vehicle behaviors compared to prior methods.[1]

4.1 Introduction

Motivation

Autonomous vehicle technologies have seen a surge in popularity over the last decade, with the potential to improve flow of traffic, safety and fuel efficiency [1]. While existing technology is being gradually introduced into scenarios such as highway driving and low-speed parking, the traffic intersection scenario presents a complex challenge, especially in the absence of vehicle-to-vehicle communication. In any traffic scenario, an autonomous vehicle agent must plan to follow its desired route while accounting for surrounding agents to maintain safety. The difficulty arises from the variability in the possible behaviors of the surrounding agents [2], [94]. To address this difficulty, significant research has been devoted to modelling these agent predictions as multi-modal distributions [10, 18, 74]. Such models capture uncertainty in both high-level decisions (desired route) and low-level executions (agent position, heading, speed).

[1]This work was done jointly with Siddharth Nair and appeared in [59].

The focus of this work is to investigate how to incorporate these multi-modal distributions for the surrounding agents (target vehicles) into a planning framework for the autonomous agent (ego vehicle). The main challenge in designing such a framework is to find a good balance between performance, safety, and computation time. We investigate this planning problem in the context of constrained optimal control and use Model Predictive Control (MPC), the state-of-the-art for real-time optimal control [56, 66]. We assume that predictions of the other agents are specified as Gaussian Mixture Models (GMMs) and propose a Stochastic Model Predictive Control (SMPC) framework that incorporates probabilistic collision avoidance and actuation constraints.

Related work

There is a large body of work focusing on the application of SMPC to autonomous driving [79, 67]. A typical SMPC algorithm involves solving a chance-constrained finite horizon optimal control problem in a receding horizon fashion [54]. The chance-constrained SMPC optimization problem, offers a less conservative modelling framework for handling constraints in uncertain environments compared to Robust MPC frameworks. In the context of autonomous driving, SMPC has been used for imposing chance constraints accounting for uncertainty in both the ego vehicle and target vehicle predictions in applications such as autonomous lane change [15, 34], cruise control [57] and platooning [17]. A common feature of works in these applications is that the predictions are either uni-modal or the underlying mode of the multi-modal prediction can be inferred accurately.

In order to handle multi-modal predictions (specifically GMMs), the work in [98] and [93] proposes nonlinear SMPC algorithms that suitably reformulate the collision avoidance chance constraint for all possible modes and demonstrate their algorithms at traffic intersection scenarios. However, a non-convex optimization problem is formulated to find a single open-loop input sequence that satisfies the collision avoidance chance constraints for all modes and possible evolutions of the target vehicles over the prediction horizon given by the GMM. This approach can be conservative and a feasible solution to the optimization problem may not exist. To remedy this issue, we optimize over a class of feedback policies [32] which adds flexibility due to the ability to react to realizations of the ego and target vehicle trajectories along the prediction horizon.

The work in [76] and [9] is the closest to our approach. The former considers uni-modal predictions and fixes a policy (computed offline) for which the optimization problem can still be conservative. The latter proposes a Robust MPC scheme that optimizes over policies with known, bounded polytopic supports for multi-modal distributions of the obstacle. However, this is done by enumerating over all possible sequences of vertices of the support of the obstacle distribution, resulting in a formulation where the number of constraints is exponential in the prediction horizon.

Contributions

Our main contributions are summarised as follows:

- We propose a SMPC framework that optimizes over a novel class of feedback policies designed to exploit additional structure in the GMM predictions. These policies assume feedback over the ego vehicle state and the target vehicle positions, thus reducing conservatism. Moreover, we show that our SMPC optimization problem can be posed as a Second-Order Cone Program (SOCP).
- We present a systematic evaluation of our framework along axes of (i) mobility, (ii) comfort, (iii) conservatism and (iv) computational efficiency at a simulated traffic intersection. To demonstrate the impact of optimizing over feedback policies, we include two baselines: (i) a chance-constrained Nonlinear MPC optimizing over open-loop sequences with the multi-modal collision avoidance chance-constraints as in [93] and (ii) an ablation of our SMPC formulation, which optimizes over open-loop sequences instead of feedback policies.

4.2 Problem Formulation

In this section we formally cast the problem of designing SMPC in the context of autonomous driving at intersections.

4.2.1 Preliminaries

Notation

- The index set $\{k_1, k_1+1, \ldots, k_2\}$ is denoted by $\mathcal{I}_{k_1}^{k_2}$.
- Given random variables X and Y, we write $X|Y$ as a shorthand for $X|Y = y$ (X conditioned on $Y = y$).
- Given a normal distribution $\mathcal{N}(\mu, \Sigma)$ and $\beta \geq 0$, define the $\beta-$confidence ellipsoid $\mathcal{E}(\mu, \Sigma, \beta) = \{x : (x-\mu)^\top \Sigma^{-1}(x-\mu) \leq \beta\}$.
- For any two positive semi-definite matrices $M_1, M_2 \in \mathbb{S}_+^n$, we have $M_1 \prec M_2 \Leftrightarrow M_2 - M_1 \in \mathbb{S}_+^n$.
- The binary operator $\otimes$ denotes the Kronecker product.
- The partial derivative of function $f(x,u)$ with respect to x at $(x,u) = (x_0, u_0)$ is denoted by $\partial_x f(x_0, u_0)$.

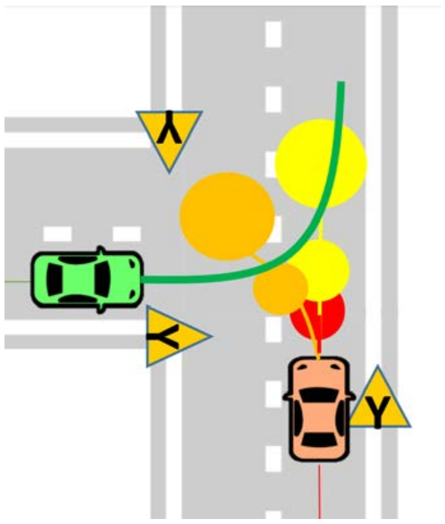


Figure 4.1: Depiction of an unsignalized intersection with green Ego Vehicle (EV) and orange Target Vehicle (TV). The EV arrives first and intends to turn left (green path). The EV is provided 3 possible predictions of TV behavior in decreasing order of probability (red: give EV right-of-way, orange: turn left and yellow: go straight).

Intersection setup

Consider the intersection shown in Figure 4.1. The Ego Vehicle (EV), depicted in green, is the vehicle to be controlled. All other vehicles at the intersection are called Target Vehicles (TVs).

EV modeling

Let $x_t = [X_t\ Y_t\ \theta_t\ v_t]^\top$ be the state of the EV at time t with (X_t, Y_t) and θ_t being its position and heading respectively in global coordinates, and v_t being its speed. The discrete-time dynamics of the EV are given by the Euler discretization of the kinematic bicycle model [47], denoted as $x_{t+1} = f^{EV}(x_t, u_t)$. The control inputs $u_t = [a_t\ \delta_t]^\top$ are the acceleration a_t and front steering angle δ_t. The system state and input constraints are given by polytopic sets $\mathcal{X}$,$\mathcal{U}$ respectively.

As depicted by the green path in Figure 4.1, we assume a kinematically feasible reference trajectory,

$$\{(x_t^{ref}, u_t^{ref})\}_{t=0}^{T} \tag{4.1}$$

is provided for the EV. This serves as the EV's desired trajectory which can be computed offline (or online at low frequency) by solving a nonlinear trajectory optimization problem, while accounting for the EV's route, actuation limits, and static environment constraints (like lane boundaries, traffic rules). However, this reference does not consider the dynamically evolving TVs for real-time obstacle avoidance.

TV predictions

We focus our presentation on a single TV although the proposed framework can be generalised for multiple TVs. Let $o_t = [X^o_t \ Y^o_t]^\top$ be the position (and state) of the TV at time t. For the purpose of dynamic obstacle avoidance, we assume that we are provided predictions of the TV's position for N time steps in the future, given by the random variables $\{o_{k|t}\}_{k=1}^N$, where each random variable $o_{k|t}$ describes the position of the TV at time $t+k$ conditioned on the current position o_t. The predictions for the TV at time t is encoded as a Gaussian Mixture Model (GMM):

$$\mathcal{G}_t = \left\{p_{t,j}, \{\mu_{k|t,j}, \Sigma_{k|t,j}\}_{k=1}^N\right\}_{j=1}^J \tag{4.2}$$

where

- J is the number of possible modes.
- $\mathbb{P}(\sigma_t = j|o_t) = p_{t,j} \in [0,1]$ is the probability that the mode at time t, denoted by σ_t, is j for some $j \in \mathcal{I}_1^J$.
- $o_{k|t,j} \sim \mathcal{N}(\mu_{k|t,j}, \Sigma_{k|t,j})$, i.e., the position of the TV at time $t+k$ conditioned on the current position o_t and mode $\sigma_t = j$, denoted by $o_{k|t,j}$, is given by the Gaussian distribution $\mathcal{N}(\mu_{k|t,j}, \Sigma_{k|t,j})$.

The multi-modal distribution of $o_{k|t}$ is thus given by

$$o_{k|t} \sim \sum_{j=1}^J p_{t,j}\mathcal{N}(\mu_{k|t,j}, \Sigma_{k|t,j}) \ \ \forall k \in \mathcal{I}_1^N. \tag{4.3}$$

4.2.2 Qualitative Control Objectives

- Design a feedback control $u_t = \pi_t(x_t, o_t)$ for the EV to track the reference trajectory (4.1) while avoiding collisions with the TV, whose predictions are given by the GMM (4.2).
- The algorithm to compute u_t must be computationally tractable and not overly conservative.

4.2.3 Overview of Proposed Approach

We propose a novel SMPC formulation to compute the feedback control u_t. The optimization problem of our SMPC takes the form,

$$
\begin{aligned}
\min_{\mathbf{u}_t,\mathbf{x}_t,\mathbf{o}_t} \quad & J_t(\mathbf{x}_t,\mathbf{u}_t) && \text{(4.4a)}\\
\text{s.t.} \quad & x_{k+1|t} = f_k^{EV}(x_{k|t},u_{k|t}), && \text{(4.4b)}\\
& o_{k+1|t}|o_{k|t} \sim f_k^{TV}(o_{k|t}), && \text{(4.4c)}\\
& \mathbb{P}(g_k(o_{k+1|t},x_{k+1|t}) \leq 0) \leq \epsilon, && \text{(4.4d)}\\
& (x_{k+1|t},u_{k|t}) \in \mathcal{X}\times\mathcal{U}, && \text{(4.4e)}\\
& \mathbf{u}_t \in \Pi(\mathbf{x}_t,\mathbf{o}_t), && \text{(4.4f)}\\
& x_{0|t} = x_t,\ o_{0|t} = o_t, && \text{(4.4g)}\\
& \forall k \in \mathcal{I}_0^{N-1}
\end{aligned}
$$

where $\mathbf{u}_t = [u_{0|t},\dots,u_{N-1|t}]$, $\mathbf{x}_t = [x_{0|t},\dots,x_{N|t}]$ and $\mathbf{o}_t = [o_{0|t},\dots,o_{N|t}]$. The SMPC feedback control action is given by the optimal solution of (4.4) as

$$u_t = \pi_{\text{SMPC}}(x_t,o_t) = u^\star_{0|t} \tag{4.5}$$

where the EV and TV state feedback enters as (4.4g).

The objective (4.4a) penalizes deviation of the EV trajectory from the reference (4.1) and the collision avoidance constraints are imposed as chance constraints (4.4d) along with polytopic state and input constraints $\mathcal{X}$,$\mathcal{U}$ for the EV in (4.4e). The TV predictions in (4.4c) are obtained from the GMM (4.2) by assuming additional dynamical structure (discussed in Sec. 4.3.2). Using this, our SMPC optimizes over a novel policy class (4.4f) that depends on predictions of the EV's and TV's states as opposed to optimization over open-loop sequences, which can often be conservative. This is our main theoretical contribution.

Moreover, we show that (4.4) can be posed as a convex optimization problem, making it computationally tractable and amenable to state-of-the-art solvers. This is achieved by (i) a convex parameterisation of policy class $\Pi(\cdot)$, (ii) using affine time varying models to generate EV and TV state predictions in (4.4b) and (4.4c), (iii) using a convex cost function for (4.4a), and (iv) constructing convex inner approximations of (4.4d).

4.3 Stochastic MPC with Multimodal Predictions

In this section, we detail our SMPC formulation for the EV to track the reference (4.1) while incorporating multi-modal predictions from (4.2) of the TV for obstacle avoidance.

4.3.1 EV prediction model

The EV prediction model (4.4b) is a linear time-varying model, obtained by linearizing $f^{EV}(\cdot)$ about the reference trajectory (4.1). Defining $x_{k|t} - x^{ref}_{t+k} = \Delta x_{k|t}$ and $u_{k|t} - u^{ref}_{t+k} = \Delta u_{k|t}$, we have $\forall k \in \mathcal{I}_0^{N-1}$,

$$\begin{aligned} \Delta x_{k+1|t} &= A_{k|t}\Delta x_{k|t} + B_{k|t}\Delta u_{k|t} + w_{k|t} \\ A_{k|t} &= \partial_x f^{EV}(x_{t+k}^{ref}, u_{t+k}^{ref}),\ B_{k|t} = \partial_u f^{EV}(x_{t+k}^{ref}, u_{t+k}^{ref}) \end{aligned} \tag{4.6}$$

where the additive process noise $w_{k|t} \sim \mathcal{N}(0, \Sigma_w)$ (i.i.d with respect to k) models linearization error and other stochastic noise sources. Consequently, the polytopic state and input constraints (4.4e) are then written as chance constraints $\forall k \in \mathcal{I}_0^{N-1}$,

$$\begin{aligned} &\mathbb{P}((\Delta x_{k+1|t}, \Delta u_{k|t}) \in \Delta\mathcal{X}_k \times \Delta\mathcal{U}_k) \geq 1 - \epsilon \\ &\Delta\mathcal{X}_k = \{\Delta x : F_k^x \Delta x \leq f_k^x\}, \Delta\mathcal{U}_k = \{\Delta u : F_k^u \Delta u \leq f_k^u\}. \end{aligned} \tag{4.7}$$

4.3.2 TV prediction model

Notice that the multi-modal distribution of the random variable $o_{k+1|t}$ (as given by (4.3)) is independent of the random variable $o_{k|t}$, i.e., $o_{k+1|t}|o_{k|t} = o_{k+1|t}$. However, since the TV positions are governed by the laws of motion, it is reasonable to assume dynamical structure to link positions at consecutive time steps such that $o_{k+1|t}|o_{k|t} \neq o_{k+1|t}$. Thus, a measurement of $o_{k|t}$ reduces the uncertainty in $o_{k+1|t}$, which can be exploited by the policy class (4.4f) in the SMPC problem. We model this structure using affine time-varying models for each mode, constructed using the parameters of the GMM (4.2) as follows.

We assume $\forall k \in \mathcal{I}_0^{N-1}, \forall j \in \mathcal{I}_1^J$ that $o_{k+1|t,j}$ conditioned on $o_{k|t,j}$ is given by the distribution

$$o_{k+1|t,j}|o_{k|t,j} \sim \mathcal{N}(T_{k|t,j}o_{k|t,j} + c_{k|t,j}, \tilde{\Sigma}_{k+1|t,j}), \tag{4.8}$$

where $\forall k \in \mathcal{I}_1^{N-1}$,

$$\begin{aligned} T_{k|t,j} &= \sqrt{\Sigma_{k+1|t,j}}\sqrt{\Sigma_{k|t,j}^{-1}},\ c_{k|t,j} = \mu_{k+1|t,j} - T_{k|t,j}\mu_{k|t,j}, \\ \tilde{\Sigma}_{k+1|t,j} &= \Sigma_n \quad (\text{for some } \Sigma_n \prec\prec \Sigma_{k+1|t,j}, \Sigma_n \succ 0) \end{aligned} \tag{4.9}$$

and $T_{0|t,j} = I,\ c_{0|t,j} = \mu_{1|t,j} - o_t,\ \tilde{\Sigma}_{1|t,j} = \Sigma_{1|t,j}$. For $k \geq 1$, this model captures the notion that if $o_{k|t,j}$ is S standard deviations away from $\mu_{k|t,j}$, then $o_{k+1|t,j}$ will also be about S standard deviations away from $\mu_{k+1|t,j}$.

The model in (4.8) describes inference over the state of the TV by conditioning the observations of the TV states on a given mode $\sigma_t = j$. To complete the description of the TV's multi-modal predictions, we now discuss inference over the mode σ_t using observations $o_{k|t}$. There are sophisticated methods for computing the posterior probability distribution $\mathbb{P}(\sigma_t = j|o_{k|t}) = p_{k|t,j}$ (e.g. using a Bayesian framework or expectation maximization for inference on state and mode) but these would complicate our SMPC optimization problem (4.4). Instead, we assume $p_{k|t,j} = p_{t,j}\ \forall k < \bar{k}$ and at some $\bar{k}$, the mode can be exactly inferred from the TV's state. We assume that there is a specified confidence level β fixed for each mode and timestep. Motivated by [9], our model for determining $\bar{k}$ is given by

$$\begin{aligned} \min\{k \in \mathcal{I}_1^N :\ &\mathcal{E}(\mu_{l|t,j_1}, \Sigma_{l|t,j_1}, \beta) \cap \mathcal{E}(\mu_{l|t,j_2}, \Sigma_{l|t,j_2}, \beta) = \emptyset, \\ &\forall l \geq k, \forall j_1, j_2 \in \mathcal{I}_1^J\} \end{aligned} \tag{4.10}$$

Thus $\bar{k}$ is the minimum time step along the prediction horizon such that all $\beta-$confidence ellipsoids for subsequent time steps are pairwise disjoint for all modes. Given $\bar{k}$, we assume that the mode of the ith TV can be determined using state $o_{\bar{k}|t}$ as given by

$$p_{\bar{k}|t,j} = \begin{cases} 1 & \text{if } o_{\bar{k}|t} \in \mathcal{E}(\mu_{k|t,j}, \Sigma_{k|t,j}, \beta) \\ 0 & \text{otherwise} \end{cases} \tag{4.11}$$

and $p_{k|t,j} = p_{\bar{k}|t,j}\ \forall k \in \mathcal{I}_{\bar{k}}^{N-1}$. In summary, we use (4.8) and (4.11) to obtain the TV prediction model $f_k^{TV}(\cdot)$ in (4.4c) as the multi-modal distribution of $o_{k+1|t}$ conditioned on $o_{k|t}$,

$$o_{k+1|t}|o_{k|t} \sim \sum_{j=1}^{J} p_{k|t,j}\mathcal{N}(T_{k|t,j}o_{k|t} + c_{k|t,j}, \tilde{\Sigma}_{k+1|t,j}). \tag{4.12}$$

Defining $n_{k|t,j} \sim \mathcal{N}(0, \tilde{\Sigma}_{k+1|t,j})\ \forall k \in \mathcal{I}_0^{N-1}, \forall j \in \mathcal{I}_1^J$, we can use properties of Gaussian random variables (closure under linear combinations and the orthogonality principle) to rewrite (4.12) as

$$o_{k+1|t} = T_{k|t,\sigma_t}o_{k|t} + c_{k|t,\sigma_t} + n_{k,\sigma_t} \quad \forall k \in \mathcal{I}_0^{N-1} \tag{4.13}$$

where the posterior distributions of the mode σ_t are determined using

$$\sigma_t|o_{k|t} = \begin{cases} \sigma_t & \forall k < \bar{k} \\ \sigma_t|o_{\bar{k}|t} \text{ given by (4.11)} & \forall k \geq \bar{k} \end{cases}. \tag{4.14}$$

4.3.3 Multi-modal Collision Avoidance Constraints

We assume that we are given or can infer (e. g., using finite differences) the TV rotation matrices for each mode along the prediction horizon as $\{\{R_{k|t,j}\}_{k=1}^N\}_{j=1}^J$.For collision avoidance between the EV and the TV, we impose the following chance constraint

$$\mathbb{P}(g_{k|t}(P_{k|t}, o_{k|t}) \geq 1\ ) \geq 1 - \epsilon \quad \forall k \in \mathcal{I}_1^N \tag{4.15}$$

where $P_{k|t} = [\Delta X_{k|t}\ \Delta Y_{k|t}]^\top + P_{k|t}^{ref}$ with $P_{k|t}^{ref} = [X_{t+k}^{ref}\ Y_{t+k}^{ref}]^\top$ given by (4.1), $[\Delta X_{k|t}\ \Delta Y_{k|t}]^\top$ given by (4.6), and

$$g_{k|t}(P, o) = (P - o)^\top R_{k|t,\sigma_t}^\top \begin{bmatrix} \frac{1}{a_{ca}^2} & 0 \\ 0 & \frac{1}{b_{ca}^2} \end{bmatrix} R_{k|t,\sigma_t}(P - o). \tag{4.16}$$

$a_{ca} = a_{TV} + d_{EV}, b_{ca} = b_{TV} + d_{EV}$ are semi-axes of the ellipse containing the TV's extent with a buffer of d_{EV}. $g_{k|t}(P, o) \geq 1$ implies that the EV's extent (modelled as a disc of radius d_{EV} and center P) does not intersect the TV's extent (modelled as an ellipse with semi-axes a_{TV}, b_{TV} and center o, oriented as given by $R_{k|t,\sigma_t}$). This constraint is non-convex because it involves the integral of the nonlinear function $g_{k|t}(\cdot)$ over the joint distribution of

$(P_{k|t}, o_{k|t}, \sigma_t)$. To address the multi-modality, we conservatively impose the chance constraint conditioned on every mode with the same risk level $1-\epsilon$ because of the following implication

$$\begin{aligned}&\mathbb{P}(g_{k|t,j}(P_{k|t}, o_{k|t,j}) \geq 1) \geq 1-\epsilon \;\; \forall j \in \mathcal{I}_1^J\\ \Rightarrow&\mathbb{P}(g_{k|t}(P_{k|t}, o_{k|t}) \geq 1) \geq 1-\epsilon\end{aligned}$$

where $g_{k|t,j}(P,o)$ is defined by conditioning (4.16) on mode j. To address the nonlinearity, we use the convexity of $g_{k|t,j}(\cdot)$ to construct its linear under-approximation in the following proposition.

Proposition 1. *For collision avoidance, define EV positions $\forall k \in \mathcal{I}_1^N, \forall j \in \mathcal{I}_1^J$ as $P^{ca}_{k|t,j} = \mu_{k|t,j} + \frac{1}{\sqrt{g_{k|t,j}(P^{ref}_{k|t}, \mu_{k|t,j})}}(P^{ref}_{k|t} - \mu_{k|t,j})$.*

The linearization is given by:

$$g^L_{k|t,j}(P,o) = \partial_P g_{k|t}(P^{ca}_{k|t,j}, \mu_{k|t,j})(P - P^{ca}_{k|t,j}) + \partial_o g_{k|t}(P^{ca}_{k|t,j}, \mu_{k|t,j})(o - \mu_{k|t,j}) \tag{4.17}$$

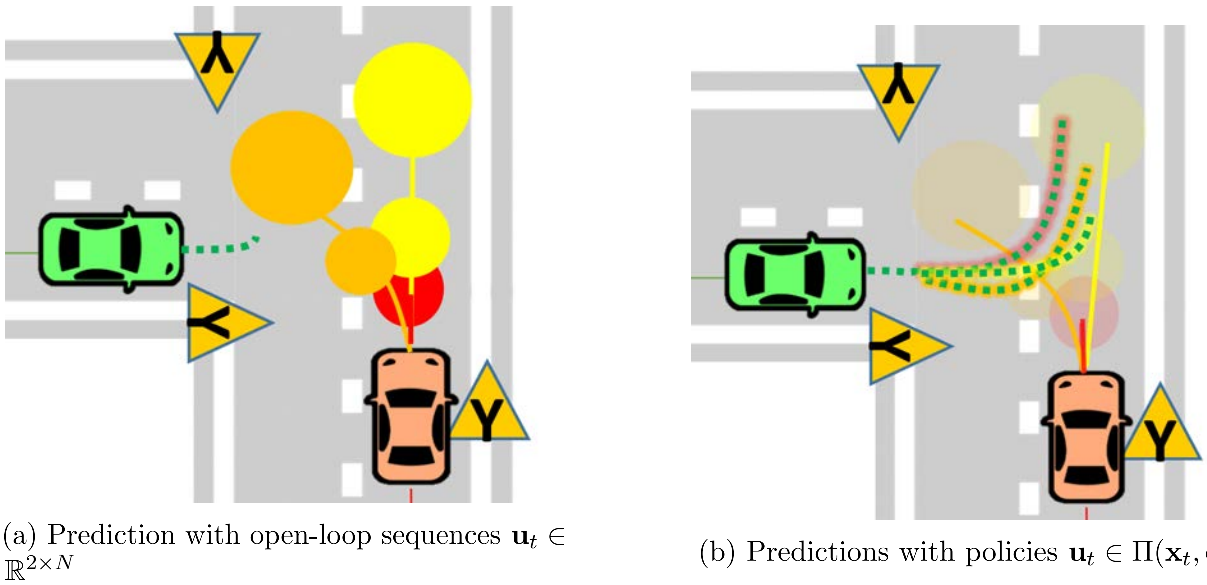
We have $\forall k \in \mathcal{I}_0^{N-1}$

$$\begin{aligned}&\mathbb{P}(g^L_{k+1|t,j}(P_{k+1|t}, o_{k+1|t,j}) \geq 0\ ) \geq 1-\epsilon \; \forall j \in \mathcal{I}_1^J\\ \Rightarrow&\mathbb{P}(g_{k|t}(P_{k+1|t}, o_{k+1|t}) \geq 1) \geq 1-\epsilon\end{aligned} \tag{4.18}$$

Proof. Appendix C.1 □

We build on these approximations to complete the reformulation of (4.15) for EV and TV trajectories given by (4.6), (4.13) in closed-loop with a feedback policy.

4.3.4 Predictions using EV and TV state feedback policies


(a) Prediction with open-loop sequences $\mathbf{u}_t \in \mathbb{R}^{2\times N}$

(b) Predictions with policies $\mathbf{u}_t \in \Pi(\mathbf{x}_t, \mathbf{o}_t)$

Figure 4.2: Open-loop input sequences versus policies. In (a), solving (4.4) over open-loop sequences can be conservative because EV prediction (green-dashed) from a single sequence of control inputs must satisfy all the obstacle avoidance constraints. In (b), optimizing over policies (4.4f) allows for different EV predictions depending on the TV trajectory realizations (green-dashed with highlights corresponding to different TV trajectories).

We present a policy class, $\Pi(\mathbf{x}_t, \mathbf{o}_t)$, for the EV that accounts for state feedback from both the EV and TVs. In general, optimizing over state feedback policies subsumes optimizing over open-loop sequences in (4.4) because the former has a strictly larger feasible set. For the assumed EV model (4.6) and mode-dependent TV model (4.13), we propose the following feedback policy for the EV:

$$\Delta u_{k|t} = \pi_{k|t}(x_{k|t}, o_{k|t}) = \\ \begin{cases} h_{k|t} + \sum\limits_{l=0}^{k-1} M_{l,k|t} w_{l|t} + K_{k|t} o_{k|t} & \text{if } k < \bar{k} \\ h^1_{k|t} + \sum\limits_{l=0}^{k-1} M^1_{l,k|t} w_{l|t} + K^1_{k|t} o_{k|t} & \text{if } k \geq \bar{k},\ \sigma_t = 1 \\ \vdots & \vdots \\ h^J_{k|t} + \sum\limits_{l=0}^{k-1} M^J_{l,k|t} w_{l|t} + K^J_{k|t} o_{k|t} & \text{if } k \geq \bar{k},\ \sigma_t = J \end{cases}. \tag{4.19}$$

This policy uses affine disturbance feedback for feedback over EV states (c.f. [32] for proof of equivalence) and linear feedback over TV states. Moreover since the mode of the TV can be determined as given by (4.14), we use separate mode-dependent policies along the prediction horizon for $k \geq \bar{k}$. For each $j \in \mathcal{I}_1^J$, denote the policy parameterization in (4.19) by $\mathbf{h}_t^j \in \mathbb{R}^{2N}, \mathbf{M}_t^j \in \mathbb{R}^{2N\times 4N}, \mathbf{K}_t^j \in \mathbb{R}^{2N\times 2N}$ (see Appendix C.2 for definitions). Given x_t and o_t, define the following set of policy parameters of (4.19) that satisfy the EV state and input chance constraints (4.7) and the multi-modal collision avoidance chance constraint (4.18):

$$\Pi_t(x_t, o_t) = \left\{ \{\mathbf{h}_t^j, \mathbf{M}_t^j, \mathbf{K}_t^j\}_{j=1}^J \middle| \begin{array}{c} (4.7), (4.18) \text{ hold } \forall k \in \mathcal{I}_0^{N-1}, \\ \Delta x_{0|t} = x_t - x_t^{ref}, o_{0|t} = o_t \end{array} \right\} \tag{4.20}$$

Proposition 2. *Let $\epsilon < \frac{1}{2}$. Then the set of policy parameters $\Pi_t(x_t, o_t)$ is a Second-Order Cone (SOC).*

Proof. Appendix C.3 □

4.3.5 SMPC Optimization Problem

We define the cost (4.4a) of the SMPC optimization problem (4.4) to penalise deviations of the EV state and input trajectories from the reference trajectory (4.1) as

$$J_t(\mathbf{x}_t, \mathbf{u}_t) = \mathbb{E}\left(\sum_{k=0}^{N-1} \Delta x_{k+1|t}^\top Q \Delta x_{k+1|t} + \Delta u_{k|t}^\top R \Delta u_{k|t} \right) \tag{4.21}$$

where $Q \succ 0, R \succ 0$. With this cost definition, we obtain the EV control (4.5) from our SMPC formulation as:

$$\min_{\{\mathbf{h}_t^j, \mathbf{K}_t^j, \mathbf{M}_t^j\}_{j=1}^J \in \Pi_t(x_t, o_t)} J_t(\mathbf{x}_t, \mathbf{u}_t).$$

The next proposition characterises this optimization problem.

Proposition 3. *The SMPC control action* (4.5) *is given by solving the following Second-Order Cone Program (SOCP):*

$$\begin{aligned} \min_{s, \{\mathbf{h}_t^j, \mathbf{K}_t^j, \mathbf{M}_t^j\}_{j=1}^J} \quad & s \\ s.t. \quad & \{\mathbf{h}_t^j, \mathbf{K}_t^j, \mathbf{M}_t^j\}_{j=1}^J \in \Pi_t(x_t, o_t), \\ & N_t(\{\mathbf{h}_t^j, \mathbf{K}_t^j, \mathbf{M}_t^j\}_{j=1}^J) \leq s + r_t \end{aligned} \tag{4.22}$$

where $N_t(\cdot)$ is convex and quadratic in $\{\mathbf{h}_t^j, \mathbf{K}_t^j, \mathbf{M}_t^j\}_{j=1}^J$ and r_t is a constant determined by x_t, the EV reference trajectory (4.1) *and the process noise covariance Σ_w.*

Proof. Appendix C.4 □

4.4 Experimental Design

In order to assess the benefits of our proposed stochastic MPC formulation, we use the CARLA [25] simulator to run closed-loop simulations[1]. In this section, we describe the simulation environment, evaluated scenarios, and prediction framework used to generate multimodal predictions. We then show how the stochastic MPC formulation is integrated into a planning framework to address these multimodal predictions. Finally, we provide a set of metrics and baseline policies used to evaluate the performance of our approach.

4.4.1 CARLA Simulation Environment

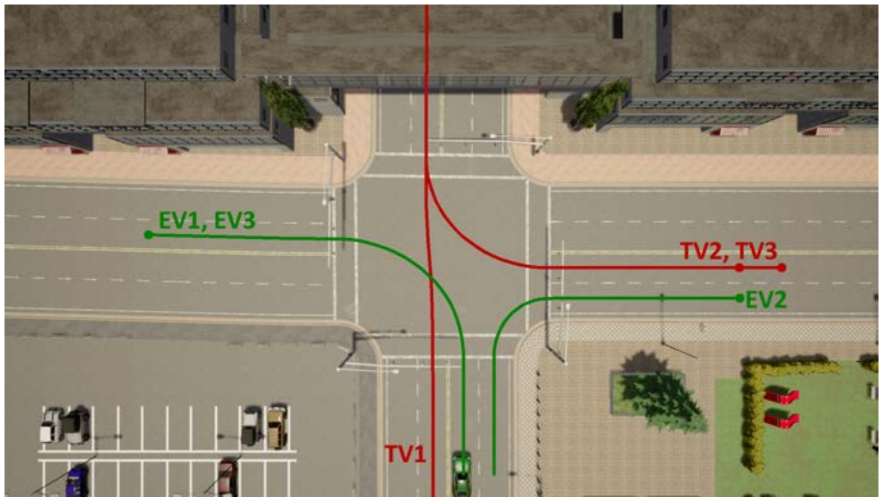


Figure 4.3: Routes used for EV and TV vehicles. The scenario is indicated by the numeric label. The EV follows a green path, while the TV follows a red path with the same scenario index.

The EV is tasked with passing through the Town05 intersection depicted in Figure 4.3, while avoiding any TVs simultaneously driving through the same intersection. The scenarios are:

1. EV turns left while TV drives straight.
2. EV turns right while TV turns left into the same road segment.
3. EV turns left while TV turns left into the opposite road segment.

As in the previous chapter, we use the synchronous mode of CARLA for improved repeatability of experiments.

[1]Supplementary material, simulation videos, and code are available at the following link: https://sites.google.com/view/siddharthnair/research/ICRA2022

The control for the TV is described by a simple nonlinear MPC (NMPC) scheme to track a pre-specified reference. For realism, the TV's NMPC also uses short predictions of the EV for simple distance-based obstacle avoidance constraints. Additionally, we note that while the intersection has traffic lights, these are ignored in our experiments. Effectively, each scenario is treated as an interaction at an unsignalized intersection.

4.4.2 Multimodal Prediction Architecture

A key assumption in our stochastic MPC approach is that the target vehicle's future motion is described by a GMM (4.2). To generate this probability distribution, we implement a context-aware variant of the MultiPath prediction model proposed in [18], identical to the model introduced in Section 3.3.3. The input to this neural network architecture is a semantic image and pose history for the target vehicle in the scene. These inputs are used to generate a GMM over future trajectories by regression with respect to a set of predetermined anchors and classification over the anchor probabilities. We note, however, that our SMPC approach is appropriate for any prediction model that produces a GMM trajectory distribution.

To train the model, we selected a subset of the Lyft Level 5 prediction dataset [42] with 16 anchor trajectories identified using k-means clustering. At runtime, we opted to truncate the GMM produced by MultiPath. In particular, we pick the top $J = 3$ modes and renormalize the mode probabilities correspondingly.

4.4.3 Motion Planning Architecture

Given the predicted multimodal distribution for the target vehicle, we use the framework introduced in Section 4.3 to generate feedback control policies for deployment.

In addition to predictions of the target vehicles, a dynamically feasible ego reference trajectory (4.1) must be provided. In our work, a NMPC problem is solved using IPOPT [92] which tracks a a high-level route provided by the CARLA waypoint API, while incorporating dynamic and actuation constraints.

Given the EV reference (4.1) and TV predictions (4.2), the SMPC optimization problem (4.22) is solved using Gurobi [37] to compute feedback control (4.5). When an infeasible problem is encountered, a braking control is commanded. The corresponding control action $u^*_{0|t}$ is given as a reference to a low-level control module that sets the vehicle's steering, throttle, and brake inputs. These loops are repeated until all vehicles in the scenario reach their destination.

4.4.4 Policies

We evaluate the following set of policies:

- **SMPC**: Our proposed framework, given by solving (4.22) which optimizes over our policy class (4.20).

- **SMPC_OL**: This model is an ablation of our approach, where $\mathbf{u}_t \in \mathbb{R}^{2\times N}$ replaces (4.4f) and the GMM (4.2) is used directly in (4.4c) instead of (4.12).
- **SMPC_BL**: Based off [93], a nonlinear SMPC algorithm is used with the collision avoidance chance constraints (4.4d) reformulated using VP concentration bounds. The optimization is performed over open-loop sequences, i.e., $\mathbf{u}_t \in \mathbb{R}^{2\times N}$ instead of (4.4f), and no additional dynamical structure assumed as in (4.4c), i.e., $o_{k+1|t} \sim f_k^{TV}(o_t)$.

4.4.5 Evaluation Metrics

To evaluate the performance of our approach with respect to the baseline policies, we introduce a set of closed-loop behavior metrics. A desirable planning framework enables high mobility without being overly conservative, allowing the timely completion of the driving task, while maintaining passenger comfort. The computation time should also not be exorbitant to allow for real-time processing of updated scene information.

The following metrics are used to assess these factors:

- **Mobility**: We record the time the EV takes to reach its goal: $\tilde{\mathcal{T}}_{episode}$, normalised by the time taken to reach the same goal without target vehicles.
- **Comfort**: We record (1) the peak lateral acceleration: $\tilde{\mathcal{A}}_{lat}$, normalised by the peak lateral acceleration in the absence of TVs, (2) the average longitudinal jerk: $\bar{\mathcal{J}}_{long}$ and (3) the average lateral jerk: $\bar{\mathcal{J}}_{lat}$. High values are undesirable, linked to sudden braking or steering.
- **Conservatism**: We record (1) the deviation of the closed-loop trajectory from the trajectory obtained without TVs: $\Delta\tau$, (2) the smallest Euclidean distance observed between the EV and TV: $\bar{d}_{min}$ and (3) the feasibility % of the SMPC optimization problem: $\mathcal{F}$. While an increase in (1) and (2) corresponds to higher safety, a large value could indicate an over-conservative planner and resulting low values of (3).
- **Computational Efficiency**: This is the average time taken by the solver: $\bar{\mathcal{T}}_{solve}$; lower is better.

4.5 Results

In this section, we present the results of the various SMPC policies (Section 4.4.4). For each scenario, we deploy each policy for 4 different initial conditions by varying: (1) the starting distance from the intersection in $\{10\,\mathrm{m}, 20\,\mathrm{m}\}$ and (2) the initial speed in $\{8\,\mathrm{m\,s^{-1}}, 10\,\mathrm{m\,s^{-1}}\}$. For all the policies, we use a prediction horizon of $N = 10$, discretization time-step for the EV dynamics as $dt = 0.2\,\mathrm{s}$, and a risk level of $\epsilon = 0.05$ for the chance constraints.

The closed-loop performance metrics, averaged across the initial conditions, are shown in Table 4.1. **SMPC** is generally able to improve or maintain mobility compared to the baselines. In Scenario 1, the mobility is a little lower for the **SMPC** approach, since it stays close to the tight left turn reference trajectory while the baselines deviate from the original lane. There is a noticeable improvement in comfort and conservatism metrics, as the **SMPC** can stay close to the TV-free reference trajectory without incurring high acceleration/jerk or getting too close to the TVs. While the **SMPC_OL** ablation is the fastest in solve time, the **SMPC** can be used real-time at approximately 15 - 20 Hz.

Table 4.1: Closed-loop performance comparison across all scenarios.

Scenario	**Policy**	**Mobility**	**Comfort**			**Conservatism**			**Efficiency**
		$\tilde{\mathcal{T}}_{episode}$	$\tilde{\mathcal{A}}_{lat}$	$\bar{\mathcal{J}}_{long}$ $(\frac{m}{s^3})$	$\bar{\mathcal{J}}_{lat}$ $(\frac{m}{s^3})$	$\Delta\tau$ (m)	$\bar{d}_{min}$ (m)	$\mathcal{F}$ (%)	$\bar{\mathcal{T}}_{solve}$ (ms)
1	**SMPC_BL**	1.099	1.830	7.336	12.824	5.396	5.270	98.66	33.063
	SMPC_OL	1.096	2.113	2.072	10.746	3.790	4.848	100.00	1.600
	SMPC	1.232	1.217	2.098	6.554	1.242	3.578	100.00	62.399
2	**SMPC_BL**	1.366	1.712	12.334	6.016	1.130	10.174	78.59	44.280
	SMPC_OL	1.571	1.553	7.929	5.557	3.900	9.288	78.01	1.676
	SMPC	1.294	1.012	2.531	5.691	1.027	7.884	100.00	63.949
3	**SMPC_BL**	1.370	1.620	10.971	4.378	2.707	11.582	83.09	43.372
	SMPC_OL	1.211	1.878	3.944	8.247	1.603	11.962	93.07	1.641
	SMPC	1.152	1.159	2.143	8.595	0.734	7.917	100.00	61.400

The merits of the **SMPC** approach can also be observed in Figures 4.4, 4.5, and 4.6. In general, the **SMPC** approach demonstrates less deviation from the reference route and more efficient completion of the maneuver. The **SMPC** approach exhibits a capability to get closer to the TV and incur a bit more risk while avoiding collisions. In Scenario 2 and Scenario 3, we observe a prediction mode that indicates the TV will intrude in the EV's lane that corresponds to an illegal traffic behavior. While the baseline methods **SMPC_OL** and **SMPC_BL** both become infeasible faced with the misspecified prediction mode, the **SMPC** is able to remain feasible throughout the scenario due to the flexible feedback policy structure. These results highlight the benefits of our approach in closed-loop execution.

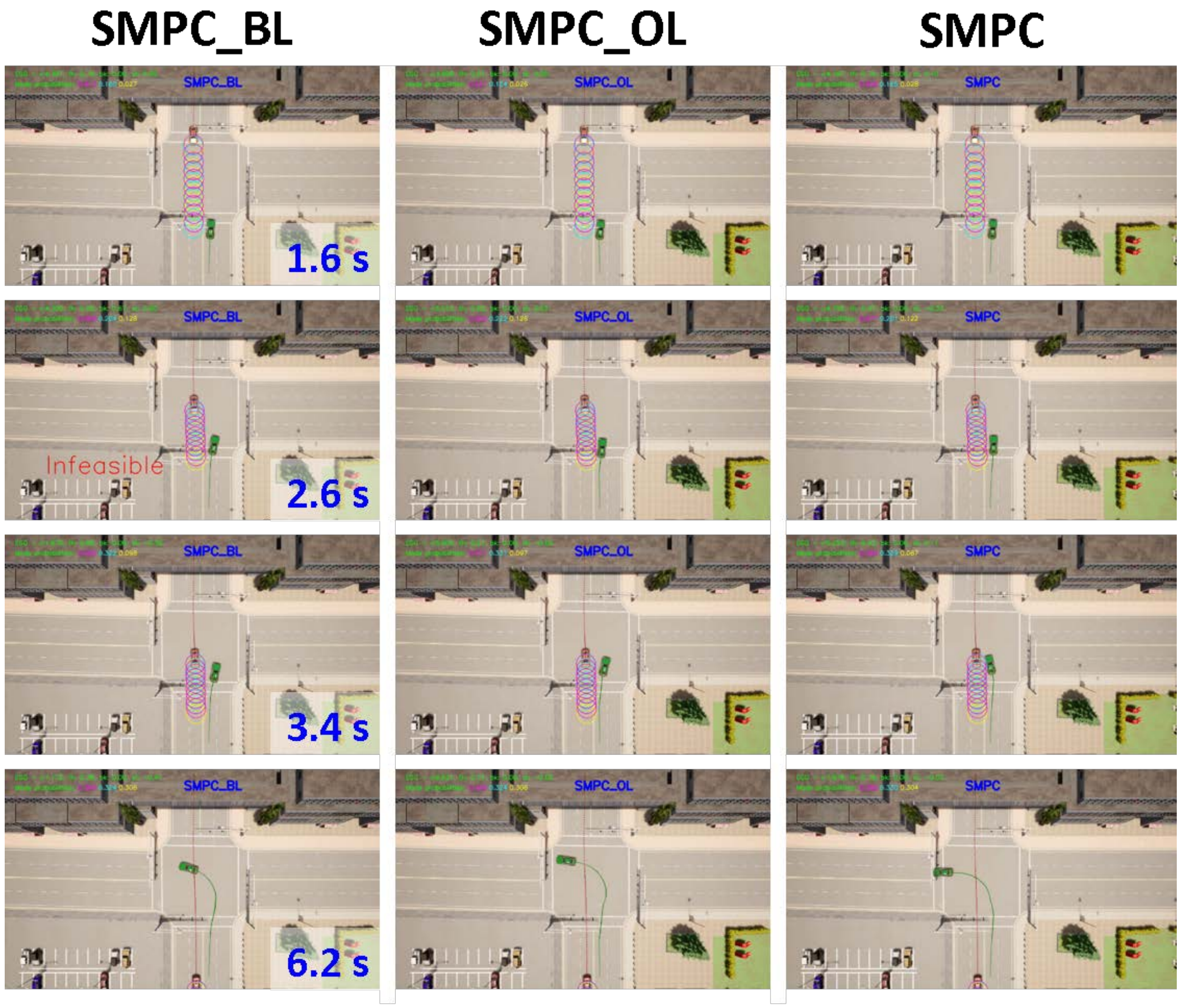


Figure 4.4: In Scenario 1, the EV (green) turns left while avoiding an oncoming TV (green) driving straight. The first two columns depict the actions taken by the EV under the baseline SMPC policies that rely on open-loop input sequences. The last column shows our proposed SMPC framework with mode-dependent feedback policies.

In this example, the SMPC baseline policies (**SMPC_BL** and **SMPC_OL**) lead to over-conservative behavior, with the EV deviating to the right between $t = 2.6\,\text{s}$ and $t = 3.4\,\text{s}$ to avoid a low probability lane intrusion event by the TV. In fact, the problem is infeasible for the **SMPC_BL** at $t = 2.6\,\text{s}$. In contrast, our **SMPC** approach is able to stay close to the original reference trajectory and remain in the same turning lane, in contrast to the baseline approaches, as observed around $t = 6.2\,\text{s}$.

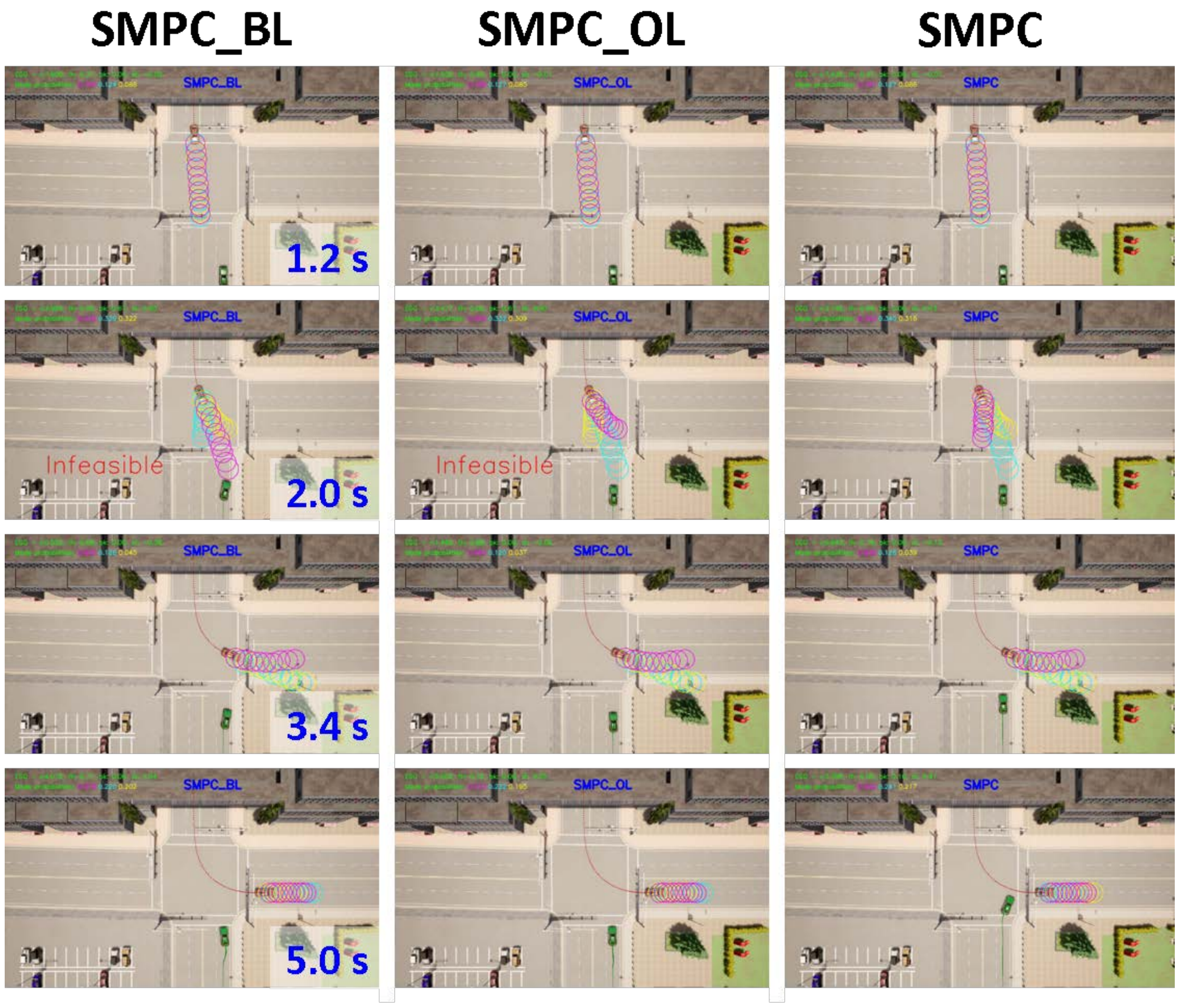


Figure 4.5: In Scenario 2, the EV (green) turns right while avoiding an oncoming TV (green) turning right into the same road segment. The first two columns depict the actions taken by the EV under the baseline SMPC policies that rely on open-loop input sequences. The last column shows our proposed SMPC framework with mode-dependent feedback policies.

In this example, we observe the flexibility of our **SMPC** approach to address potentially misspecified predictions. Around $t = 2.0\,\text{s}$, we observe the prediction model capturing the actual left turn behavior of the TV but also predicting an illegal lane intrusion into the EV's path (i. e., a behavior that a TV following traffic rules would not take). This results in both SMPC baselines (**SMPC_BL** and **SMPC_OL**) becoming infeasible at $t = 2.0\,\text{s}$. In contrast, our **SMPC** approach is able to remain feasible and make more efficient route progress as observed at $t = 3.4\,\text{s}$ and $t = 5.0\,\text{s}$.

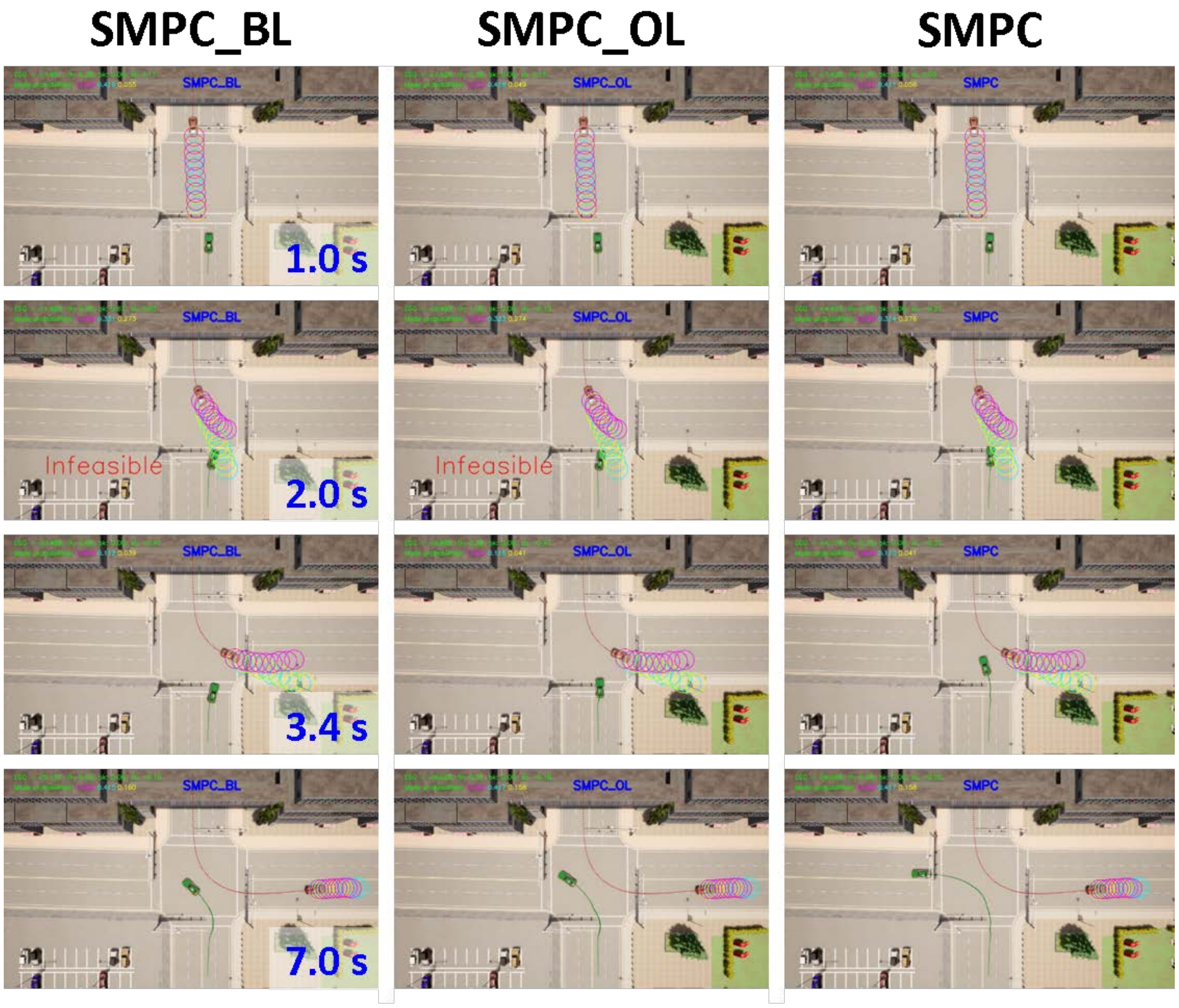


Figure 4.6: In Scenario 3, the EV (green) turns left while avoiding an oncoming TV (green) turning left into the opposite road segment. The first two columns depict the actions taken by the EV under the baseline SMPC policies that rely on open-loop input sequences. The last column shows our proposed SMPC framework with mode-dependent feedback policies. Similar to Scenario 2, we observe the flexibility of our **SMPC** approach to address potentially misspecified predictions. The TV has the same reference trajectory as in Scenario 2, and we observe the same challenges with infeasibility for **SMPC_BL** and **SMPC_OL** policies at $t = 2.0\,\mathrm{s}$ due to predictions of lane intrusion by the TV. The **SMPC** approach again demonstrates improved route efficiency and less deviation from the reference trajectory compared to the baseline approaches, as observed at $t = 7.0\,\mathrm{s}$.

4.6 Discussion

In this chapter, we presented a SMPC planning framework leveraging feedback policies for EV trajectory tracking control with collision avoidance chance constraints. In particular, by exploiting additional structure in GMM predictions, we proposed a policy class that uses affine disturbance feedback over EV states and linear feedback over TV states. The policy class includes mode-dependent parameters that enable the same policy to produce different responses for varying TV behavior realizations. Optimization over this policy class enlarges the feasible set versus optimization over fixed open-loop input sequences that must satisfy constraints for all likely TV behavior realizations.

We incorporated this feedback policy class into a SMPC formulation for EV trajectory tracking, where the EV was tasked with following a given reference trajectory at a traffic intersection while avoiding nearby TVs. The resulting SMPC optimization problem was shown to be a SOCP, which is computationally tractable and amenable to efficient state-of-the-art solvers like Gurobi.

The CARLA simulator was used to evaluate our approach in closed-loop simulations. For a set of interactive traffic intersection scenarios, we demonstrated the benefits of our approach along axes of mobility, comfort, conservatism, and computational efficiency. Our approach resulted in closed-loop trajectories that remained near the original reference trajectory and exhibited improved comfort and mobility as compared to the open-loop baselines. In addition, our approach was able to remain 100% feasible in all scenarios evaluated, reducing conservatism compared to the open-loop baselines.

There are multiple areas of improvement in this work. First, the ego reference trajectory is dynamically feasible but does not incorporate the obstacle avoidance constraints. This means that, for safety, the EV may need to apply inputs that deviate significantly from the reference (e. g., emergency braking). This can result in large deviations from the linearization, increasing model mismatch of the LTV model considered in our framework from the original nonlinear model. To address this point, in practice, iterative methods like trust-region sequential quadratic programming (SQP) could be applied to our framework to mitigate the impact of model mismatch due to an unsafe reference trajectory [60]. Another challenge is scaling this approach to several target vehicles, where there may be interactive behaviors among non-ego agents. Incorporating interaction-aware prediction models (like [74]) that generate GMMs per agent in the scene would be an interesting extension to this problem.

Chapter 5

Conclusion

Despite recent advances in and many potential benefits of AVs, there still remains several challenges until widescale adoption. The largest gap is in trust, where both passengers and proximal road users need to feel comfortable using and sharing the road with AVs. To bridge this gap, both prediction and planning algorithms need to include high fidelity driver models. This will help AVs to integrate better in mixed roadways and behave in a manner that conforms to the expectations of human stakeholders, helping to improve trust and acceptance.

The key challenge, in predicting human driver behaviors, is to balance the competing objectives of precise versus accurate predictions. We would like to keep prediction uncertainties small to allow for efficient route progress and reduce the incidence of infeasible control behaviors for the AV. Yet, if we are too optimistic in our prediction capability of other agents, then unexpected and edge case behaviors (such as drunk drivers, inclement weather, emergency vehicles) would not be adequately captured. This could lead to unanticipated accidents or near-collisions. In both extremes, the AV performance would be suboptimal and harm trust – overly timid and conservative AVs would travel too slowly for passengers yet overly aggressive AVs could drive unsafely and rashly. Finding the right sweet spot is vital for widescale acceptance and adoption of AVs.

In this dissertation, we sought to address some of these points through algorithmic frameworks that could adapt to observed target vehicle behavior and balance route efficiency with safety.

In Chapter 2, we first introduced a nominal multimodal prediction model, that given knowledge of potential goals in a parking lot (i. e., the unoccupied parking spots), could improve trajectory prediction of human parking behaviors.

This was then extended to set-based, multimodal predictions in more general road networks in Chapter 3. By maintaining a confidence threshold and adapting to observed prediction errors observed online, we demonstrated the benefits in terms of closed-loop motion planning with collision avoidance in a simulated traffic intersection in CARLA. In particular, faced with an agent adhering to normative behavior learned by the prediction model, the AV was able to proceed without unnecessarily braking. In contrast, faced with an irrational driver that exhibited swerving and poor speed-following, the AV was able to slow down and avoid collisions before proceeding.

Finally, in Chapter 4, we investigated an alternative approach to incorporating these uncertain predictions. Rather than tuning confidence level bounds, the idea was to use feedback policies to allow for anticipatory control responses to target vehicle behaviors. This is in contrast to open-loop baseline methods, for which a single control input sequence is determined that must satisfy collision avoidance constraints for all likely target vehicle behavior realizations. Benefits in metrics of mobility, comfort, and efficiency demonstrated the advantages of incorporating adaptive feedback policies for multimodal predictions.

There are several areas of improvement in this work. Model-based methods could be combined with data-driven methods, both in terms of the prediction architecture and in terms of the confidence-aware framework. For the former, there exists prediction models like [74] that predict control input distributions and use a dynamics model to propagate these input distributions to determine the associated state distributions. In addition, training losses are augmented with penalties based on considerations like kinematic feasibility [20] and road geometry [35]. These ideas seek to reduce generalization error and improve likelihood of observed behavior.

Yet, there will likely always be anomalous or unexpected behavior, where having model-based and data-driven models in a confidence-aware framework could be beneficial. One approach could be to apply switched driver models to model traffic agents, as in [49]. The data-driven models would be deployed in nominal cases, where other vehicles are behaving as expected. As the prediction error degrades, backward reachability techniques could be used to prioritize safety faced with irrational agents. Methods like not-at-fault planning based on offline computation of parametrized forward sets [90] and improvements in online computation of reachable sets through warm start initializations [39] could enable these switched confidence systems in the future. Another idea is to use reachability for short-term horizons, for which the impact of disturbances may be smaller, and use the data-driven models for longer horizons where reachable sets would be overconservative. This would allow efficient driving in most cases, while allowing for safety-preserving actions to be taken in near-collision states. Determining a safe switching strategy would be an important consideration in this design. Alternatively, the composition of data-driven models with a planning framework could be analyzed through formal methods. In particular, the use of falsification to identify edge cases that violate a safety specification could help in synthesizing adversarial examples for further training [80, 29].

With respect to the stochastic MPC framework introduced, a key limitation is the need for linearization in order to make solving the problem real-time feasible. This means that careful handling of linearization errors, for example when large deviations from the input sequence linearization point, is required. Iterative methods like trust-region Sequential Quadratic Programming (SQP) may help to improve solution quality at the expense of increased computation time [60]. Another challenge is scaling up the stochastic MPC framework to several target vehicles, where there may be interactive behaviors among non-ego agents. Improvements to the base Gaussian mixture model that incorporate some interaction terms may help to better model the joint scene evolution dynamics in the optimization problem.

Bibliography

[1] National Highway Traffic Safety Administration. “Automated Vehicles for Safety”. In: *https://www.nhtsa.gov/technology-innovation/automated-vehicles-safety* (2020).

[2] National Highway Traffic Safety Administration. “Crash factors in intersection-related crashes: An on-scene perspective”. In: *http://www-nrd.nhtsa.dot.gov/Pubs/811366.pdf* (2010).

[3] National Highway Traffic Safety Administration et al. “The visual detection of DWI motorists”. In: *DOT HS 808* 677 (2004).

[4] Matthias Althoff, Olaf Stursberg, and Martin Buss. “Model-based probabilistic collision detection in autonomous driving”. In: *IEEE Transactions on Intelligent Transportation Systems* 10.2 (2009), pp. 299–310.

[5] Partners for Automated Vehicle Education. *Pave Poll: Fact Sheet.* URL: `https://pavecampaign.org/wp-content/uploads/2020/05/PAVE-Poll_Fact-Sheet.pdf` (visited on 11/13/2021).

[6] Mayank Bansal, Alex Krizhevsky, and Abhijit Ogale. “Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst”. In: *arXiv preprint arXiv:1812.03079* (2018).

[7] Yaakov Bar-Shalom, X Rong Li, and Thiagalingam Kirubarajan. *Estimation with applications to tracking and navigation: theory algorithms and software.* John Wiley & Sons, 2004.

[8] I. Batkovic et al. “Real-Time Constrained Trajectory Planning and Vehicle Control for Proactive Autonomous Driving With Road Users”. In: *2019 18th European Control Conference (ECC).* June 2019, pp. 256–262. DOI: `10.23919/ECC.2019.8796099`.

[9] Ivo Batkovic et al. “A Robust Scenario MPC Approach for Uncertain Multi-modal Obstacles”. In: *IEEE Control Systems Letters* 5.3 (2020), pp. 947–952.

[10] H. A. P. Blom and Y. Bar-Shalom. “The interacting multiple model algorithm for systems with Markovian switching coefficients”. In: *IEEE Transactions on Automatic Control* 33.8 (1988), pp. 780–783. DOI: `10.1109/9.1299`.

[11] Freddy A Boulton, Elena Corina Grigore, and Eric M Wolff. “Motion prediction using trajectory sets and self-driving domain knowledge”. In: *arXiv preprint arXiv:2006.04767* (2020).

[12] Rodney Brooks. "The big problem with self-driving cars is people". In: *IEEE spectrum: technology, engineering, and science News* 27 (2017).

[13] Holger Caesar et al. "nuScenes: A multimodal dataset for autonomous driving". In: *arXiv preprint arXiv:1903.11027* (2019).

[14] Ashwin Carvalho et al. "Automated driving: The role of forecasts and uncertainty—A control perspective". In: *European Journal of Control* 24 (2015), pp. 14–32.

[15] Ashwin Carvalho et al. "Stochastic predictive control of autonomous vehicles in uncertain environments". In: *12th International Symposium on Advanced Vehicle Control*. 2014, pp. 712–719.

[16] Sergio Casas, Wenjie Luo, and Raquel Urtasun. "IntentNet: Learning to Predict Intention from Raw Sensor Data". In: *CoRL* 87.CoRL (2018), pp. 947–956.

[17] V Causevic et al. "Information-Constrained Model Predictive Control with Application to Vehicle Platooning". In: *IFAC-PapersOnLine* 53.2 (2020), pp. 3124–3130.

[18] Yuning Chai et al. "Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction". In: *arXiv preprint arXiv:1910.05449* (2019).

[19] Zhuang Jie Chong et al. "Synthetic 2D LIDAR for precise vehicle localization in 3D urban environment". In: *2013 IEEE International Conference on Robotics and Automation*. IEEE. 2013, pp. 1554–1559.

[20] Henggang Cui et al. "Deep Kinematic Models for Physically Realistic Prediction of Vehicle Trajectories". In: *arXiv preprint arXiv:1908.00219* (2019).

[21] Henggang Cui et al. "Multimodal Trajectory Predictions for Autonomous Driving using Deep Convolutional Networks". In: (2019), pp. 2090–2096. DOI: `10.1109/icra.2019.8793868`.

[22] Nachiket Deo and Mohan M. Trivedi. "Multi-Modal Trajectory Prediction of Surrounding Vehicles with Maneuver based LSTMs". In: *IEEE Intelligent Vehicles Symposium, Proceedings* 2018-June.Iv (2018), pp. 1179–1184. DOI: `10.1109/IVS.2018.8500493`.

[23] Jeffrey Donahue et al. "Long-term recurrent convolutional networks for visual recognition and description". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 2625–2634.

[24] Alexey Dosovitskiy et al. "CARLA: An Open Urban Driving Simulator". In: *Proceedings of the 1st Annual Conference on Robot Learning*. 2017, pp. 1–16.

[25] Alexey Dosovitskiy et al. "CARLA: An Open Urban Driving Simulator". In: *Proceedings of the 1st Annual Conference on Robot Learning*. 2017, pp. 1–16.

[26] K. Driggs-Campbell, R. Dong, and R. Bajcsy. "Robust, Informative Human-in-the-Loop Predictions via Empirical Reachable Sets". In: *IEEE Transactions on Intelligent Vehicles* 3.3 (2018), pp. 300–309. DOI: `10.1109/TIV.2018.2843125`.

[27] Katherine Driggs-Campbell, Vijay Govindarajan, and Ruzena Bajcsy. “Integrating intuitive driver models in autonomous planning for interactive maneuvers”. In: *IEEE Transactions on Intelligent Transportation Systems* 18.12 (2017), pp. 3461–3472.

[28] The Economist. *Special Report on Autonomous Vehicles*. URL: `https://www.economist.com/special-report/2018-03-03` (visited on 11/13/2021).

[29] Daniel J. Fremont et al. “Scenic: A Language for Scenario Specification and Data Generation”. In: *CoRR* abs/2010.06580 (2020). arXiv: `2010.06580`. URL: `https://arxiv.org/abs/2010.06580`.

[30] David Fridovich-Keil et al. “Confidence-aware motion prediction for real-time collision avoidance”. In: *International Journal of Robotics Research* 39.2-3 (2020), pp. 250–265. ISSN: 17413176. DOI: `10.1177/0278364919859436`. URL: `http://journals.sagepub.com/doi/10.1177/0278364919859436`.

[31] Tobias Gindele, Sebastian Brechtel, and Rudiger Dillmann. “Learning driver behavior models from traffic observations for decision making and planning”. In: *IEEE Intelligent Transportation Systems Magazine* 7.1 (2015), pp. 69–79. ISSN: 19391390. DOI: `10.1109/MITS.2014.2357038`.

[32] Paul J Goulart, Eric C Kerrigan, and Jan M Maciejowski. “Optimization over state feedback policies for robust control with constraints”. In: *Automatica* 42.4 (2006), pp. 523–533.

[33] Vijay Govindarajan, Katherine Driggs-Campbell, and Ruzena Bajcsy. “Data-driven reachability analysis for human-in-the-loop systems”. In: *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*. 2017, pp. 2617–2622. DOI: `10.1109/CDC.2017.8264039`.

[34] Andrew Gray et al. “Stochastic predictive control for semi-autonomous vehicles with an uncertain driver model”. In: *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*. IEEE. 2013, pp. 2329–2334.

[35] Ross Greer, Nachiket Deo, and Mohan Trivedi. “Trajectory Prediction in Autonomous Driving with a Lane Heading Auxiliary Loss”. In: *IEEE Robotics and Automation Letters* 6.3 (2021), pp. 4907–4914.

[36] Jacopo Guanetti, Yeojun Kim, and Francesco Borrelli. “Control of connected and automated vehicles: State of the art and future challenges”. In: *Annual Reviews in Control* 45 (2018), pp. 18–40.

[37] Gurobi Optimization, LLC. *Gurobi Optimizer Reference Manual*. 2021. URL: `https://www.gurobi.com`.

[38] Kaiming He et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[39] Sylvia L. Herbert et al. "Reachability-Based Safety Guarantees using Efficient Initializations". In: *2019 IEEE 58th Conference on Decision and Control (CDC)*. 2019, pp. 4810–4816. DOI: `10.1109/CDC40024.2019.9029575`.

[40] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory". In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[41] Adam Houenou et al. "Vehicle trajectory prediction based on motion model and maneuver recognition". In: *2013 IEEE/RSJ international conference on intelligent robots and systems*. IEEE. 2013, pp. 4363–4369.

[42] John Houston et al. *One Thousand and One Hours: Self-driving Motion Prediction Dataset*. 2020. arXiv: `2006.14480 [cs.CV]`.

[43] Y. Hu, W. Zhan, and M. Tomizuka. "Probabilistic Prediction of Vehicle Semantic Intention and Motion". In: *2018 IEEE Intelligent Vehicles Symposium (IV)*. June 2018, pp. 307–313. DOI: `10.1109/IVS.2018.8500419`.

[44] Joel Janai et al. "Computer vision for autonomous vehicles: Problems, datasets and state-of-the-art". In: *arXiv preprint arXiv:1704.05519* (2017).

[45] Nitin R Kapania and J Christian Gerdes. "Design of a feedback-feedforward steering controller for accurate path tracking and stability at the limits of handling". In: *Vehicle System Dynamics* 53.12 (2015), pp. 1687–1704.

[46] Byeoung Do Kim et al. "Probabilistic vehicle trajectory prediction over occupancy grid map via recurrent neural network". In: *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC* 2018-March (2018), pp. 399–404. DOI: `10.1109/ITSC.2017.8317943`.

[47] Jason Kong et al. "Kinematic and dynamic vehicle models for autonomous driving control design". In: *2015 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2015, pp. 1094–1099.

[48] Kirsten Korosec. *Cruise can now give passengers rides in driverless cars in California*. URL: `https://techcrunch.com/2021/06/04/cruise-can-now-give-passengers-rides-in-driverless-cars-in-california/` (visited on 11/13/2021).

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

[50] Stéphanie Lefèvre, Dizan Vasquez, and Christian Laugier. "A survey on motion prediction and risk assessment for intelligent vehicles". In: *ROBOMECH Journal* 1.1 (Dec. 2014), p. 1. ISSN: 2197-4225. DOI: `10.1186/s40648-014-0001-z`. URL: `http://www.robomechjournal.com/content/1/1/1`.

[51] Stéphanie Lefèvre, Dizan Vasquez, and Christian Laugier. "A survey on motion prediction and risk assessment for intelligent vehicles". In: *ROBOMECH journal* 1.1 (2014), pp. 1–14.

[52] Aarian Marshall. *Why Are Parking Lots So Tricky for Self-Driving Cars?* URL: `https://www.wired.com/story/why-are-parking-lots-so-tricky-for-self-driving-cars/` (visited on 01/24/2020).

[53] Bruce Mehler et al. "Impact of incremental increases in cognitive workload on physiological arousal and performance in young adult drivers". In: *Transportation Research Record* 2138.1 (2009), pp. 6–12.

[54] Ali Mesbah. "Stochastic model predictive control: An overview and perspectives for future research". In: *IEEE Control Systems Magazine* 36.6 (2016), pp. 30–44.

[55] Kaouther Messaoud et al. *Trajectory Prediction for Autonomous Driving based on Multi-Head Attention with Joint Agent-Map Representation.* 2020. arXiv: `2005.02545 [cs.CV]`.

[56] Manfred Morari and Jay H Lee. "Model predictive control: past, present and future". In: *Computers & Chemical Engineering* 23.4-5 (1999), pp. 667–682.

[57] Dominik Moser et al. "Flexible spacing adaptive cruise control using stochastic model predictive control". In: *IEEE Transactions on Control Systems Technology* 26.1 (2017), pp. 114–127.

[58] Motional. *nuScenes Prediction Task Leaderboard.* URL: `https://www.nuscenes.org/prediction` (visited on 11/13/2021).

[59] Siddharth H Nair et al. "Stochastic MPC with Multi-modal Predictions for Traffic Intersections". In: *arXiv preprint arXiv:2109.09792* (2021).

[60] Jorge Nocedal and Stephen Wright. *Numerical optimization.* Springer Science & Business Media, 2006.

[61] Sven Nyholm and Jilles Smids. "Automated cars meet human drivers: responsible human-robot coordination and the ethics of mixed traffic". In: *Ethics and Information Technology* (2018), pp. 1–10.

[62] Seong Hyeon Park et al. "Sequence-to-Sequence Prediction of Vehicle Trajectory via LSTM Encoder-Decoder Architecture". In: *IEEE Intelligent Vehicles Symposium, Proceedings* 2018-June.Iv (2018), pp. 1672–1678. DOI: `10.1109/IVS.2018.8500658`.

[63] Dominik Petrich et al. "Assessing map-based maneuver hypotheses using probabilistic methods and evidence theory". In: *17th International IEEE Conference on Intelligent Transportation Systems (ITSC).* 2014, pp. 995–1002. DOI: `10.1109/ITSC.2014.6957818`.

[64] Dominik Petrich et al. "Map-based long term motion prediction for vehicles in traffic environments". In: *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013).* 2013, pp. 2166–2172. DOI: `10.1109/ITSC.2013.6728549`.

[65] Rajesh Rajamani. *Vehicle dynamics and control.* Springer Science & Business Media, 2011.

[66] Ben Recht. *What We've Learned to Control.* 2020. URL: https://www.argmin.net/2020/06/29/tour-revisited/.

[67] Ugo Rosolia, Xiaojing Zhang, and Francesco Borrelli. "Data-driven predictive control for autonomous systems". In: *Annual Review of Control, Robotics, and Autonomous Systems* 1 (2018), pp. 259–286.

[68] Sam Roweis and Zoubin Ghahramani. "A unifying review of linear Gaussian models". In: *Neural computation* 11.2 (1999), pp. 305–345.

[69] Andrey Rudenko et al. "Human motion trajectory prediction: A survey". In: *The International Journal of Robotics Research* 39.8 (2020), pp. 895–935.

[70] Dorsa Sadigh, S Shankar Sastry, and Sanjit A Seshia. "Verifying robustness of human-aware autonomous cars". In: *IFAC-PapersOnLine* 51.34 (2019), pp. 131–138.

[71] Dorsa Sadigh et al. "Data-driven probabilistic modeling and verification of human driver behavior". In: *2014 AAAI Spring Symposium Series.* 2014.

[72] Dorsa Sadigh et al. "Information gathering actions over human internal state". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).* IEEE. 2016, pp. 66–73.

[73] Dorsa Sadigh et al. "Planning for autonomous cars that leverage effects on human actions." In: *Robotics: Science and Systems.* Vol. 2. Ann Arbor, MI, USA. 2016, pp. 1–9.

[74] Tim Salzmann et al. "Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data". In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVIII 16.* Springer. 2020, pp. 683–700.

[75] Georg Schildbach and Francesco Borrelli. "Scenario model predictive control for lane change assistance on highways". In: *2015 IEEE Intelligent Vehicles Symposium (IV).* IEEE. 2015, pp. 611–616.

[76] Georg Schildbach, Matthias Soppert, and Francesco Borrelli. "A collision avoidance system at intersections using robust model predictive control". In: *2016 IEEE Intelligent Vehicles Symposium (IV).* IEEE. 2016, pp. 233–238.

[77] Robin Schubert, Eric Richter, and Gerd Wanielik. "Comparison and evaluation of advanced motion models for vehicle tracking". In: *2008 11th international conference on information fusion.* IEEE. 2008, pp. 1–6.

[78] Jens Schulz et al. "Interaction-Aware Probabilistic Behavior Prediction in Urban Environments". In: *IEEE International Conference on Intelligent Robots and Systems* (2018), pp. 3999–4006. ISSN: 21530866. DOI: 10.1109/IROS.2018.8594095.

[79] Wilko Schwarting, Javier Alonso-Mora, and Daniela Rus. "Planning and Decision-Making for Autonomous Vehicles". In: *Annual Review of Control, Robotics, and Autonomous Systems* 1.1 (2018), pp. 187–210. DOI: 10.1146/annurev-control-060117-105157.

[80] Sanjit A Seshia, Dorsa Sadigh, and S Shankar Sastry. "Towards verified artificial intelligence". In: *arXiv preprint arXiv:1606.08514* (2016).

[81] Xu Shen, Xiaojing Zhang, and Francesco Borrelli. "Autonomous Parking of Vehicle Fleet in Tight Environments". In: (2019). arXiv: `1910.02349`. URL: `http://arxiv.org/abs/1910.02349`.

[82] Xu Shen et al. "Parkpredict: Motion and intent prediction of vehicles in parking lots". In: *2020 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2020, pp. 1170–1175.

[83] Victor A. Shia et al. "Semiautonomous Vehicular Control Using Driver Modeling". In: *IEEE Transactions on Intelligent Transportation Systems* 15.6 (2014), pp. 2696–2709. DOI: `10.1109/TITS.2014.2325776`.

[84] Thomas Streubel and Karl Heinz Hoffmann. "Prediction of driver intended path at intersections". In: *IEEE Intelligent Vehicles Symposium, Proceedings* Iv (2014), pp. 134–139. DOI: `10.1109/IVS.2014.6856508`.

[85] Pei Sun et al. "Scalability in perception for autonomous driving: Waymo open dataset". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 2446–2454.

[86] Romain Tavenard et al. "Tslearn, A Machine Learning Toolkit for Time Series Data". In: *Journal of Machine Learning Research* 21.118 (2020), pp. 1–6. URL: `http://jmlr.org/papers/v21/20-091.html`.

[87] Martin Treiber and Arne Kesting. "Traffic flow dynamics". In: *Traffic Flow Dynamics: Data, Models and Simulation, Springer-Verlag Berlin Heidelberg* (2013).

[88] Charlott Vallon et al. "A machine learning approach for personalized autonomous lane change initiation and control". In: *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2017, pp. 1590–1595.

[89] Sean Vaskov et al. "Guaranteed Safe Reachability-based Trajectory Design for a High-Fidelity Model of an Autonomous Passenger Vehicle". In: *2019 American Control Conference (ACC)*. 2019, pp. 705–710. DOI: `10.23919/ACC.2019.8814853`.

[90] Sean Vaskov et al. "Not-at-Fault Driving in Traffic: A Reachability-Based Approach". In: *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. 2019, pp. 2785–2790. DOI: `10.1109/ITSC.2019.8917052`.

[91] Dizan Alejandro Vasquez Govea, Thierry Fraichard, and Christian Laugier. "Growing Hidden Markov Models: An Incremental Tool for Learning and Predicting Human and Vehicle Motion". In: *The International Journal of Robotics Research* 28.11-12 (Nov. 2009), pp. 1486–1506.

[92] Andreas Wächter and Lorenz T Biegler. "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming". In: *Mathematical programming* 106.1 (2006), pp. 25–57.

[93] Allen Wang, Ashkan Jasour, and Brian C Williams. “Non-gaussian chance-constrained trajectory planning for autonomous vehicles under agent uncertainty”. In: *IEEE Robotics and Automation Letters* 5.4 (2020), pp. 6041–6048.

[94] Lianzhen Wei et al. “Autonomous Driving Strategies at Intersections: Scenarios, State-of-the-Art, and Future Outlooks”. In: *arXiv preprint arXiv:2106.13052* (2021).

[95] Huazhe Xu et al. “End-to-end learning of driving models from large-scale video datasets”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2174–2182.

[96] Huazhe Xu et al. “End-to-end learning of driving models from large-scale video datasets”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2174–2182.

[97] Xiaojing Zhang et al. “Autonomous Parking Using Optimization-Based Collision Avoidance”. In: *2018 IEEE Conference on Decision and Control (CDC)*. IEEE, Dec. 2019, pp. 4327–4332. ISBN: 978-1-5386-1395-5. DOI: 10.1109/cdc.2018.8619433.

[98] Bingyu Zhou et al. “Joint multi-policy behavior estimation and receding-horizon trajectory planning for automated urban driving”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 2388–2394.

Appendix A

Context-Aware Model Details

A.1 Lane Follower with Context

This section provides more details on the context-aware, model-based lane follower prediction model introduced in Section 3.3.3.

A.1.1 Scene Context Representation

The local scene is represented as a list of nearby lanes, agents, and traffic lights. This information is limited to an observation region of 40 m in front, 10 m behind, and 25 m on the left and right, matching the region considered for scene context images.

Lanes are represented as lists of poses and associated speed limit from OpenStreetMap, if available. The Overpass API[1] was used to make speed limit queries from OpenStreetMap. If any of the active traffic lights impact a lane segment, the corresponding lane segment is marked with speed limit of $0\,\mathrm{m\,s^{-1}}$ to indicate a stopping reference speed.

Vehicle and other agents (e. g., pedestrians) are represented as pose trajectories in the local frame of the target vehicle being considered. The motion of these agents are extrapolated using constant velocity motion models to get a simplified estimate of the scene context in future timesteps.

A.1.2 Lane Prior Distribution

The discrete prior probability distribution over nearby lanes was generated according to the method described in [64] and [63].

First, the prior measurements of the target vehicle were filtered using the constant velocity and turn rate (CVTR) variant of **EKF** model detailed in Section 3.3.3. Let the resulting pose distribution after incorporating all measurements be denoted as $\mathcal{N}(z_{\mathrm{obj}}, \Sigma_{\mathrm{obj}})$.

[1]Accessed using the overpy wrapper, `https://github.com/DinoTools/python-overpy`.

For each lane, the nearest pose is identified using Euclidean distance. Due to discretization of the lane and other sources of error, this pose association process is assumed to be noisy. Consequently, the nearest lane pose is distributed as $\mathcal{N}(z_{\text{lane}}, \Sigma_{\text{lane}})$, where the mean and covariance are expressed in the same local frame as the target vehicle.

Then, per lane i, a Mahalanobis distance d_i can be computed as:

$$d_i = \left(z_{\text{obj}} - z_{\text{lane},i}\right)^\top \left(\Sigma_{obj} + \Sigma_{\text{lane},i}\right)^{-1} \left(z_{\text{obj}} - z_{\text{lane},i}\right)$$

Supposing there are N lanes in view, the probability of a given lane i is determined using a softmax function with learnable temperature term T_1:

$$P\left(\text{lane } i \mid z_{\text{obj}}\right) = \frac{\exp\left(-d_i T_1\right)}{\sum_{i=1}^{N} \exp\left(-d_i T_1\right)}$$

A.1.3 Lane Rollout Generation

Given the initial filtered pose of the target vehicle and a specified lane, the corresponding lane rollout is generated by assuming that this target vehicle seeks to track the lane centerline while respecting any lead vehicles or agents. Again, this approach is similar to the method in [63].

Concretely, the lane rollout is generated through repeatedly applying the following procedure per timestep, assuming the initial state distribution is given with the CVTR EKF model as in the lane prior distribution step.

1. Project current mean state to the lane to identify lane tracking error and speed and curvature reference.

2. Identify the nearest agent that is in front and in the same lane. Call this the lead agent.

3. Generate a control input to reduce lane tracking error and follow the speed limit, subject to the lead agent.

4. Apply the control input in a lane-based EKF based on a simple kinematic model with learned input covariance (Σ_u) and disturbance covariance (Q). This results in the predicted next state distribution for the target vehicle.

The control input consists of curvature and acceleration. The curvature is generated using a feedback/feedforward (FF/FB) control architecture that considers lane error and upcoming road curvature [45]. The acceleration is generated using the Intelligent Driver Model (IDM) with the lead agent used to set the safe braking distance [87]. We defer a detailed discussion of these policies to Appendix B, where a similar approach is used to generate target vehicle behaviors in simulation.

The pose distributions estimated at each step of the above procedure form a unimodal Gaussian trajectory corresponding to tracking that given lane.

A.1.4 Lane Posterior Distribution

Equipped with the lane prior distribution and the corresponding lane rollouts, we generate the discrete posterior probability distribution by a simple cost-based likelihood function. As before, let us assume there are N lanes and that the input trajectory for the i-th lane is denoted $\mathbf{u}^i = \{u_t^i\}_{t=0}^{N_F-1}$, where N_F is the prediction horizon.

Then the cost for a given lane, parametrized by cost matrix $R \in \mathbb{S}_+^2$, is:

$$J_i = \sum_{t=0}^{N_F-1} \left\|u_t^i\right\|_R^2$$

The likelihood is expressed as a softmax function with learnable temperature term T_2:

$$P\left(\mathbf{u}^i \mid \text{lane } i\right) = \frac{\exp\left(-J_i T_2\right)}{\sum_{i=1}^{N} \exp\left(-J_i T_2\right)}$$

The basic intuition here is that lanes for which large inputs are required are more difficult to track and thus less likely to be the one followed by the target vehicle.

The posterior distribution is thus given by Bayes' theorem:

$$P\left(\text{lane } i \mid z_{\text{obj}}, \mathbf{u}^i\right) \quad \alpha \quad P\left(\mathbf{u}^i \mid \text{lane } i\right) P\left(\text{lane } i \mid z_{\text{obj}}\right)$$

Thus, we have a mode distribution given by the lane posterior, with underlying Gaussian distributions described by the lane rollouts, resulting in a Gaussian mixture trajectory distribution.

A.1.5 Learnable Parameters

We fix the parameters for the acceleration (IDM) and curvature (FF/FB) input policies based on the corresponding publications. The learnable parameters are those that capture uncertainty:

- T_1, the temperature parameter for the lane prior distribution.
- T_2, the temperature parameter for the lane cost likelihood.
- R, the cost matrix for the lane cost likelihood.
- Q, the disturbance covariance for the lane-based EKF.
- Σ_u, the input covariance for the lane-based EKF.

A.2 Regression and MultiPath with Context

The backbone network is as depicted in Figure A.1. The scene context image is preprocessed by a ResNet50 [38] network. These image features are combined with motion features generated by passing the pose history, $\{z_{k|t}^{(i)}\}_{k=-N_H}^{0}$, through a LSTM network [40]. The merged features are then processed with a final fully connected network of size 512 with dropout regularization of 20% to produce prediction features, f.

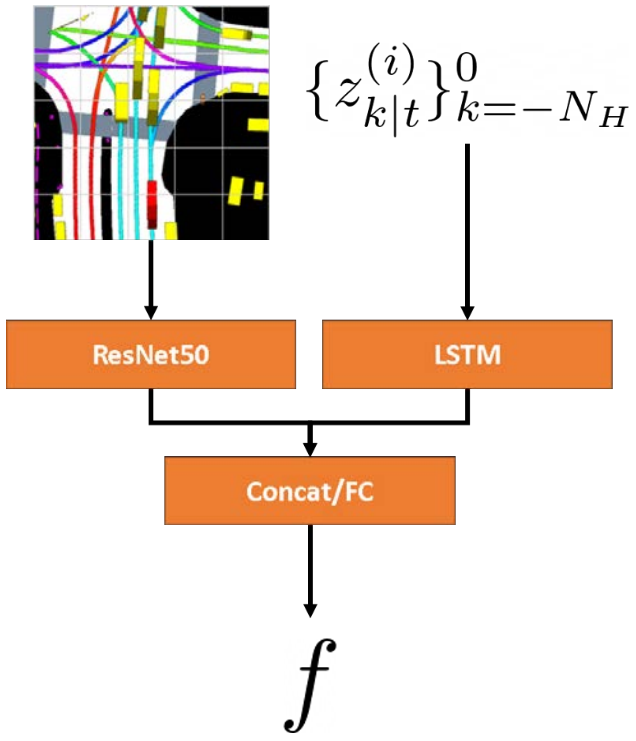


Figure A.1: Backbone network for data-driven prediction models.

The ResNet50 portion of the network was finetuned during training, starting from ImageNet pretrained weights. The network was truncated at the first output block of the conv4 layer, and the initial conv1 and conv2 portions of the truncated network were left frozen for training. The rationale for this design choice is the initial layers correspond to universal features like edges and corners, while the subsequent layers are focused on camera images which may not generalize to semantic images.

The prediction feature f is then used for predicting GMM parameters as detailed in Section 3.3.3. We note some of the key training details:

- The predicted covariance matrices are bounded by clipping the log standard deviations between 0.0 and 5.0. This also avoids numerical issues in computing the log likelihood loss.

- For training, a batch size of 32 was used. The optimizer was Stochastic Gradient Descent (SGD) with Momentum, with gradient clipping by norm 10.0.
- For training, regularization included L_2 weight penalties and dropout of 20% in the final fully connected layer.
- The nuScenes models were trained for 50 epochs, while the Lyft Level 5 models were trained for 20 epochs.

In addition, the learning rate schedule used during training was cosine annealing, where the learning rate at a given epoch follows a sinusoidal pattern between a minimum and maximum learning rate. The first epoch was set to the minimum value (learning rate warmup) and, periodically, the maximum learning rate was decreased with a decay parameter.

Appendix B

Control Policy Implementation in CARLA

B.1 Rational and Irrational Target Vehicle Policies

In order to evaluate the adaptive confidence threshold adjustment approach, the target vehicle policy was varied as detailed in Section 3.6.2. We focus here on describing, in more detail, the base policy and how it was extended to enable rational and irrational target agents. The base policy involves a decoupled control design, with a PI controller for the longitudinal component and a feedback/feedforward approach for the steering angle.

For the longitudinal motion, the desired speed and acceleration were computed as follows:

- Let the speed limit be v_{sl} and the current curvature be κ. There is a specified maximum lateral acceleration, $\bar{a}_{\text{lat}}$, for passenger comfort.
- In addition, we consider a maximum longitudinal acceleration of $\bar{a}$.
- The maximum speed considering the maximum comfortable lateral acceleration and current curvature is given by $v_{\text{cmft}} = \sqrt{\left|\frac{\bar{a}_{\text{lat}}}{\kappa}\right|}$.
- Then the speed reference is given as the minimum of the speed limit and comfort-limited speed, i. e., $v_{\text{des}} = \min\left(v_{\text{sl}}, v_{\text{cmft}}\right)$.
- The acceleration reference is given using the Intelligent Driver Model [87] with no lead obstacle, i. e., $a_{\text{des}} = \bar{a}\left(1 - \frac{v}{v_{\text{des}}}\right)^4$.
- A PI controller was used to control throttle and brake inputs given the speed and acceleration references. To simulate actuation delay, a first order time delay was applied to commanded throttle/brake inputs.

For the lateral motion, the feedback/feedforward steering policy introduced in [45] was chosen and applied to a kinematic bicycle model, for simplicity. In particular, the feedforward term neglects any dynamic effects due to the tire forces, road friction, and drag. The resulting steering angle control law is given by:

$$\delta = -k_{e_y}\left(e_y + x_{\mathrm{LA}} e_\psi\right) + L\kappa$$

where

- k_{e_y} and x_{LA} are tunable controller parameters, similar to PI gains;
- e_y is the lateral error from the lane centerline;
- e_ψ is the heading error from the lane centerline;
- κ is the current curvature as in the longitudinal motion discussion above;
- L is the vehicle wheelbase length.

Similar to the throttle/brake input, there is a first order time delay applied to the commanded steering angle to simulate actuation delay.

B.1.1 Rational Behavior

The rational behavior corresponds to an agent that tracks the lane centerline with low error and with preview of the upcoming lane. The values of k_{e_y} and x_{LA} are chosen to emphasize longer-term lane tracking errors. In addition, the speed and curvature references are provided without any delay given the current lane position of the vehicle. Rather than only considering v_{cmft} at the current lane position, the rational behavior involves considering this comfort speed over a preview of the lane ahead. This allows this agent to slow down for upcoming turns. All error and reference input signals to the policies described above are given without noise or delay.

B.1.2 Irrational Behavior

The irrational behavior models a more oscillatory tracking response. First, the curvature and speed references are provided with a delay. There is no preview of the upcoming lane, so the agent does not slow down early ahead of turns. In addition, the value of k_{e_y} is increased and the value of x_{LA} is decreased, relative to the rational agent, to induce a myopic lane tracking controller. A sinusoidal noise component is added to the measurement signal of e_y and v_{des}. Finally, the time delays used to simulate the throttle, brake, and steering subsystems were set 5 times higher than the rational model. All these perturbations led to a vehicle varying in an oscillatory fashion in speed and deviation from the lane centerline.

B.2 Lane Interval Collision Avoidance Constraints

In order to incorporate collision avoidance in the ego vehicle's policy (detailed in Section 3.6.3), we need to project the collision avoidance sets $\mathcal{O}_{k|t}$ into the ego vehicle's lane. We describe how this process is accomplished and how the interval constraints for the ego vehicle are generated.

First, we assume the lane is discretized by arclength along the route. We sample the lane centerline at a fixed resolution of $0.5\,\mathrm{m}$ to get a collection $\{s_i^l\}_{i=1}^N$. Given a value of s, we can determine the associated global coordinate along the left lane boundary and right lane boundary. The coordinates, evaluated at interpolation point s_i^l, are denoted $p_{l,i}$ and $p_{r,i}$.

B.2.1 Projection of Occupancy Sets

From our prediction framework, we are equipped with collision avoidance sets $\mathcal{O}_{k|t}$, which describe the regions the ego vehicle should avoid in global coordinates k timesteps ahead. To project these global avoidance sets into lane coordinates, we opt for a simple heuristic approach as follows:

- For a given timestep k, we compute a binary occupancy flag for the i-th lane point, denoted $b_{i,k}$.
- Concretely, $b_{i,k}$ is set to 1 (occupied) if either $p_{l,i}$ or $p_{r,i}$ fall inside $\mathcal{O}_{k|t}$. Else, $b_{i,k}$ is 0 (free). Checking for the set inclusion is straightforward, since $\mathcal{O}_{k|t}$ is represented as a collection of polytopes.

Repeating this procedure for all timesteps k in the prediction horizon and lane point indices i, we can compute the lane occupancy over time. We express this as $B_k = \{b_{i,k}\}_{i=1}^N$, which indicates the occupied lane points k timesteps ahead.

B.2.2 Feasible Intervals for Ego

Equipped with an understanding of the lane occupancy (B_k), we can now tackle the problem of determining a set of lane intervals to constrain the ego vehicle's progress. This impacts the feasible state region, $\mathcal{Z}_{k|t}$, used to impose state constraints in the lane tracking MPC problem.

We assume that we have knowledge of the ego vehicle's current lane position s_t and speed v_t. In addition, there is a desired speed v_{des}, determined in a similar manner as the target vehicle policy.

Given this information, we pose the feasible interval identification problem as one of determining when to switch between a speed tracking control and a braking control. In other words, the ego vehicle will try to follow the speed reference for as long as there is no obstacle detected in its upcoming path. This procedure is detailed in Algorithm B.1.

Algorithm B.1 Lane Interval Constraint Identification

```
function IdentifyLaneIntervals($s_t$, $v_t$, $v_{\text{des}}$, $\{B_k\}_{k=1}^{N_F}$, $\{s_i^l\}_{i=1}^{N}$)
    brake ← 0                                ▷ Determines if we should apply a stopping action.
    $s_{\text{ego}} \leftarrow s_t$, $v_{\text{ego}} \leftarrow v_t$                                ▷ Initialize ego state.
    $\mathcal{S}_{\text{lower}} = \{\}$                                ▷ Interval lower bound by timestep.
    $\mathcal{S}_{\text{upper}} = \{\}$                                ▷ Interval upper bound by timestep.
    for $k \leftarrow 1, \cdots, N_F$ do
        if ShouldBrake($s_{\text{ego}}$, $v_{\text{ego}}$, $B_k$, $\{s_i^l\}_{i=1}^{N}$) then
            brake ← 1
        if brake then
            $a = a_{\min}$                                ▷ Stopping action.
        else
            $a = -k_v\left(v_{\text{ego}} - v_{\text{des}}\right)$                                ▷ Speed tracking action.
        $s_{\text{ego}} = s_{\text{ego}} + v_{\text{ego}}\Delta t$
        $v_{\text{ego}} = \max\left(0, v_{\text{ego}} + a\Delta t\right)$
        if brake then
            $s_{\text{lower}} = s_{\text{ego}} - \epsilon$                                ▷ Interval is a buffer about the stopping action.
            $s_{\text{upper}} = s_{\text{ego}} + \epsilon$
        else
            $s_{\text{lower}}$, $s_{\text{upper}}$ ← GetInterval($s_{\text{ego}}$, $B_k$, $\{s_i^l\}_{i=1}^{N}$)
        $\mathcal{S}_{\text{lower}}$.append($s_{\text{lower}}$)
        $\mathcal{S}_{\text{upper}}$.append($s_{\text{upper}}$)
    return $\mathcal{S}_{\text{lower}}$, $\mathcal{S}_{\text{upper}}$
```

```
function ShouldBrake($s_{\text{ego}}$, $v_{\text{ego}}$, $\{b_i\}_{i=1}^{N}$, $\{s_i^l\}_{i=1}^{N}$)
    $s_{\text{stop}} \leftarrow s_{\text{ego}} + \frac{v_{\text{ego}}^2}{2|d_{\text{cmft}}|}$        ▷ Stopping arclength reached applying the brake.
    $i_{\text{start}} \leftarrow \underset{i}{\operatorname{argmin}} \left|s_i^l - s_{\text{ego}}\right|$        ▷ Index of lane point nearest braking interval start.
    $i_{\text{end}} \leftarrow \underset{i}{\operatorname{argmin}} \left|s_i^l - s_{\text{stop}}\right|$        ▷ Index of lane point nearest braking interval end.
    if $\exists j \in \{i_{\text{start}}, i_{\text{start}} + 1, \cdots, i_{\text{end}}\} : b_j = 1$ then
        return 1        ▷ Occupied lane point along braking interval.
    else
        return 0

function GetInterval($s$, $\{b_i\}_{i=1}^{N}$, $\{s_i^l\}_{i=1}^{N}$)
    $i_{\text{curr}} \leftarrow \underset{i}{\operatorname{argmin}} \left|s_i^l - s\right|$        ▷ Index of lane point nearest query $s$.
    $i_{\text{start}} \leftarrow i_{\text{curr}}$        ▷ Lane interval start index.
    $i_{\text{end}} \leftarrow i_{\text{curr}}$        ▷ Lane interval start index.
    while $i_{\text{start}} > 0 \wedge b_{i_{\text{start}}-1}$ do
        $i_{\text{start}} \leftarrow i_{\text{start}} - 1$        ▷ Decrement start index while unoccupied.
    while $i_{\text{end}} < N \wedge b_{i_{\text{end}}+1}$ do
        $i_{\text{end}} \leftarrow i_{\text{end}} + 1$        ▷ Increment end index while unoccupied.
    return $s_{i_{\text{start}}}^l, s_{i_{\text{end}}}^l$        ▷ Return arclength at lane interval start and end.
```

Appendix C

Stochastic MPC Problem Details

C.1 Proof of proposition 1

First note that $P^{ca}_{k|t,j}$ is defined such that $g_{k|t,j}(P^{ca}_{k|t,j}, \mu_{k|t,j}) = 1$. For any convex function $f(x)$, we have $\forall x_0, x : f(x) \geq f(x_0) + \partial_x f(x_0)(x - x_0)$. Since we $g_{k|t,j}(\cdot)$ is convex, we have

$$\begin{aligned} g_{k|t,j}(P, o) \geq & g_{k|t,j}(P^{ca}_{k|t,j}, \mu_{k|t,j}) + g^L_{k|t,j}(P, o) \\ = & 1 + g^L_{k|t,j}(P, o) \end{aligned}$$

So $g^L_{k|t,j}(P, o) \geq 0 \Rightarrow g_{k|t,j}(P, o) \geq 1$, and $\forall k \in \mathcal{I}^N_1$,

$$\begin{aligned} & \mathbb{P}(g^L_{k|t,j}(P_{k|t}, o_{k|t,j}) \geq 0\ ) \geq 1 - \epsilon\ \forall j \in \mathcal{I}^J_1 \\ \Rightarrow & \mathbb{P}(g_{k|t,j}(P_{k|t}, o_{k|t,j}) \geq 1) \geq 1 - \epsilon\ \forall j \in \mathcal{I}^J_1 \\ \Rightarrow & \mathbb{P}(g_{k|t}(P_{k|t}, o_{k|t}) \geq 1) \geq 1 - \epsilon \end{aligned}$$

■

C.2 Matrix definitions

$$\mathbf{h}_t^j = [h_{0|t}^\top \dots h_{\bar{k}-1|t}^\top \; h_{\bar{k}|t}^{j\top} \dots h_{N-1|t}^{j\top}]^\top$$

$$\mathbf{K}_t^j = \text{blkdiag}\left(K_{0|t}, \dots, K_{\bar{k}-1|t}, K_{\bar{k}|t}^j, \dots, K_{N-1|t}^j\right)$$

$$\mathbf{M}_t^j = \begin{bmatrix} O & \dots & \dots & \dots & O \\ M_{0,1|t} & O & \dots & \dots & O \\ \vdots & \vdots & \vdots & \vdots & \\ M_{0,\bar{k}-1|t} & \dots M_{\bar{k}-2,\bar{k}-1|t} & O & \dots & O \\ M_{0,\bar{k}|t}^j & \dots & M_{\bar{k}-1,\bar{k}|t}^j & \dots & O \\ \vdots & \vdots & \vdots & \vdots & \\ M_{0,N-1|t}^j & \dots & \dots & M_{N-2,N-1|t}^j & O \end{bmatrix}$$

$$\mathbf{A}_t = \begin{bmatrix} I_4 \\ A_{0|t} \\ A_{1|t}A_{0|t} \\ \vdots \\ \prod_{k=0}^{N-1} A_{k|t} \end{bmatrix}, \mathbf{B}_t = \begin{bmatrix} O & \dots & \dots & O \\ B_{0|t} & O & \dots & O \\ A_{1|t}B_{0|t} & B_{1|t} & \dots & O \\ \vdots & \ddots & \ddots & \vdots \\ \prod_{k=1}^{N-1} A_{k|t}B_{0|t} & \dots & \dots & B_{N-1|t} \end{bmatrix},$$

$$\mathbf{T}_t^j = \begin{bmatrix} I_2 \\ T_{0|t,j} \\ T_{1|t,j}T_{0|t,j} \\ \vdots \\ \prod_{k=0}^{N-1} T_{k|t,j} \end{bmatrix}, \mathbf{C}_t^j = \begin{bmatrix} O \\ c_{0|t,j} \\ c_{1|t,j} + T_{1|t,j}c_{0|t,j} \\ \vdots \\ c_{N-1|t,j} + \sum_{k=0}^{N-1} \prod_{l=k+1}^{N-1} T_{l|t,j}c_{k|t,j} \end{bmatrix}$$

$$\mathbf{E}_t = \begin{bmatrix} O & \dots & \dots & O \\ I_4 & O & \dots & O \\ A_{1|t} & I_4 & \dots & O \\ \vdots & \ddots & \ddots & \vdots \\ \prod_{k=1}^{N-1} A_{k|t} & \dots & \dots & I_4 \end{bmatrix}, \; \mathbf{L}_t^j = \begin{bmatrix} O & \dots & \dots & O \\ I_2 & O & \dots & O \\ T_{1|t,j} & I_2 & \dots & O \\ \vdots & \ddots & \ddots & \vdots \\ \prod_{k=1}^{N-1} T_{k|t,j} & \dots & \dots & I_2 \end{bmatrix}$$

$$\mathbf{Q} = I_{N+1} \otimes Q, \; \mathbf{R} = I_N \otimes R, \boldsymbol{\Sigma}_w = I_N \otimes \Sigma_w, \boldsymbol{\Sigma}_n = I_N \otimes \Sigma_n$$

C.3 Proof of proposition 2

Let $\boldsymbol{\Delta x}_t = [\Delta x_{0|t}^\top \dots \Delta x_{N|t}^\top]^\top$, $\boldsymbol{\Delta u}_t = [\Delta u_{0|t}^\top \dots \Delta u_{N-1|t}^\top]^\top$, $\mathbf{w}_t = [w_{0|t}^\top \dots w_{N-1|t}^\top]^\top$, $\mathbf{o}_t^j = [o_{0|t}^\top \ o_{1|t,j}^\top \dots o_{N|t,j}^\top]^\top$ and $\mathbf{n}_t^j = [n_{0|t,j}^\top \dots n_{N-1|t,j}^\top]^\top$. It can be verified that the random variable for the closed-loop response $\boldsymbol{\Delta x}_t$ of the EV with policy (4.19) conditioned on the mode $\sigma_t = j$, $\boldsymbol{\Delta u}_t^j = \mathbf{M}_t^j \mathbf{w}_t + \mathbf{h}_t^j + \mathbf{K}_t^j \mathbf{o}_t^j$, is given by

$$\begin{aligned}\boldsymbol{\Delta x}_t =& \mathbf{A}_t \Delta x_{0|t} + \mathbf{B}_t \mathbf{h}_t^j + \mathbf{B}_t \mathbf{K}_t^j (\mathbf{T}_t^j o_t + \mathbf{C}_t^j + \mathbf{L}_t^j \mathbf{n}_t^j) \\ &+ (\mathbf{E}_t + \mathbf{B}_t \mathbf{M}_t^j) \mathbf{w}_t.\end{aligned}$$

Also $\forall k \in \mathcal{I}_0^N$, define the constant matrices S_k^x, S_k^o such that $S_k^x \boldsymbol{\Delta x}_t = \Delta x_{k|t}$ and $S_k^o \mathbf{o}_t^j = o_{k|t,j}$ and similarly define S_k^u. We can rewrite the affine chance constraint (4.18) for each $j \in \mathcal{I}_1^J$, $k \in \mathcal{I}_1^N$ as

$$\begin{aligned}&\mathbb{P}(G^x_{k|t,j} S^x_k \boldsymbol{\Delta x}_t + G^o_{k|t,j} S^o_k \mathbf{o}^j_t \geq \tilde{g}_{k|t,j}) \geq 1 - \epsilon \\ \Rightarrow& \mathbb{P}\Big([G^x_{k|t,j} S^x_k (\mathbf{E}_t + \mathbf{B}_t \mathbf{M}^j_t) \ (G^o_{k|t,j} S^o_k + G^x_{k|t,j} S^x_k \mathbf{B}_t \mathbf{K}^j_t) \mathbf{L}^j_t] \begin{bmatrix} \mathbf{w}_t \\ \mathbf{n}^j_t \end{bmatrix} \\ &\leq \tilde{g}_{k|t,j} - (G^o_{k|t,j} S^o_k + G^x_{k|t,j} S^x_k \mathbf{B}_t \mathbf{K}^j_t)(\mathbf{T}^j_t o_t + \mathbf{C}^j_t) \\ &- G^x_{k|t,j} S^x_k (\mathbf{A}_t \Delta x_{0|t} + \mathbf{B}_t \mathbf{h}^j_t) \Big) \leq \epsilon \\ \Rightarrow& G^x_{k|t,j} S^x_k (\mathbf{A}_t \Delta x_{0|t} + \mathbf{B}_t \mathbf{h}^j_t) + (G^o_{k|t,j} S^o_k + G^x_{k|t,j} S^x_k \mathbf{B}_t \mathbf{K}^j_t)(\mathbf{T}^j_t o_t + \mathbf{C}^j_t) \\ +& \| [G^x_{k|t,j} S^x_k (\mathbf{E}_t + \mathbf{B}_t \mathbf{M}^j_t) \ (G^o_{k|t,j} S^o_k + G^x_{k|t,j} S^x_k \mathbf{B}_t \mathbf{K}^j_t) \mathbf{L}^j_t] \boldsymbol{\Sigma}^{j\,\frac{1}{2}} \|_2 \boldsymbol{\Phi}^{-1}(\epsilon) \\ &\geq \tilde{g}_{k|t,j}\end{aligned}$$

where $\boldsymbol{\Phi}^{-1}(\cdot)$ is the inverse CDF of $\mathcal{N}(0,1)$, $\boldsymbol{\Sigma}^j = \text{blkdiag}(I_N \otimes \Sigma_w, I_{N-1} \otimes \Sigma_n, \Sigma_{1|t,j}), G^x_{k|t,j} = \partial_P g_{k|t,j}(P^{ca}_{k|t,j}, \mu_{k|t,j})$, $G^o_{k|t,j} = \partial_o g_{k|t,j}(P^{ca}_{k|t,j}, \mu_{k|t,j})$, $\tilde{g}_{k|t,j} = G^x_{k|t,j}(P^{ca}_{k|t} - P^{ref}_{k|t,j}) + G^o_{k|t,j} \mu_{k|t,j}$. Since $\epsilon < \frac{1}{2}$, we have $\boldsymbol{\Phi}^{-1}(\epsilon) < 0$ and the above inequality can be shown to be of the form $\|Ax + b\|_2 \leq c^\top x + d$, a Second-Order Cone (SOC) constraint in $x = (\mathbf{h}^j_t, \mathbf{M}^j_t, \mathbf{K}^j_t)$. The intersection of these constraints $\forall j \in \mathcal{I}_1^J, \forall k \in \mathcal{I}_1^N$ is also a SOC set in $\{\mathbf{h}^j_t, \mathbf{M}^j_t, \mathbf{K}^j_t\}_{j=1}^J$. Similarly the affine state and input chance constraints (4.7) can be reformulated into SOC constraints, giving us the desired result. ■

C.4 Proof of proposition 3

We have to show that $\min\{J_t(\mathbf{x}_t, \mathbf{u}_t) \ : \{\mathbf{h}_t^j, \mathbf{M}_t^j, \mathbf{K}_t^j\}_{j=1}^J \in \Pi(x_t, o_t)\}$ is equivalent to (4.22) and is a SOCP. We have already shown that $\{\mathbf{h}_t^j, \mathbf{M}_t^j, \mathbf{K}_t^j\}_{j=1}^J \in \Pi(x_t, o_t)$ is a SOC constraint in the previous proposition. It remains to show that the cost can be reformulated appropriately. For this, we transform the problem into its epigraph form by introducing a additional scalar decision variable s, to get $\min\{s : J(x_t, o_t) \leq s, \{\mathbf{h}_t^j, \mathbf{M}_t^j, \mathbf{K}_t^j\}_{j=1}^J \in \Pi(x_t, o_t)\}$. Then,

$$\begin{aligned}
&s \geq J_t(\mathbf{x}_t, \mathbf{u}_t) \\
&= \sum_{j=1}^J p_{t,j} \mathbb{E}\left(\sum_{k=0}^{N-1} \Delta x_{k+1|t}^\top Q \Delta x_{k+1|t} + \Delta u_{k|t}^\top R \Delta u_{k|t} | \sigma_t = j \right) \\
&= -\Delta x_{0|t}^\top Q \Delta x_{0|t} + \sum_{j=1}^J p_{t,j} \mathbb{E}(\boldsymbol{\Delta x}_t^\top \mathbf{Q} \boldsymbol{\Delta x}_t + \boldsymbol{\Delta u}_t^{j\top} \mathbf{R} \boldsymbol{\Delta u}_t^j | \sigma_t = j) \\
&= \sum_{j=1}^J p_{t,j} ((\mathbf{h}_t^j + \mathbf{K}_t^j(\mathbf{T}_t^j o_t + \mathbf{C}_t^j))^\top (\mathbf{B}_t^\top \mathbf{Q} \mathbf{B}_t + \mathbf{R})(\mathbf{h}_t^j + \mathbf{K}_t^j(\mathbf{T}_t^j o_t + \mathbf{C}_t^j)) \\
&+ \operatorname{Trace}\big(\boldsymbol{\Sigma}_w \mathbf{M}_t^{j\top} (\mathbf{B}_t^\top \mathbf{Q} \mathbf{B}_t + \mathbf{R}) \mathbf{M}_t^j + \mathbf{L}_t^j \boldsymbol{\Sigma}_n \mathbf{L}_t^{j\top} \mathbf{K}_t^{j\top} (\mathbf{B}_t^\top \mathbf{Q} \mathbf{B}_t + \mathbf{R}) \mathbf{K}_t^j \\
&+ 2\Delta x_{0|t}^\top \mathbf{A}_t^\top \mathbf{Q} \mathbf{B}_t (\mathbf{h}_t^j + \mathbf{K}_t^j(\mathbf{T}_t^j o_t + \mathbf{C}_t^j)))) \\
&+ \underbrace{\Delta x_{0|t}^\top (\mathbf{A}_t^\top \mathbf{Q} \mathbf{A}_t - Q) \Delta x_{0|t} + \operatorname{Trace}(\boldsymbol{\Sigma}_w \mathbf{E}_t^\top \mathbf{Q} \mathbf{E}_t)}_{-r_t}
\end{aligned}$$

and the first three terms correspond to $N_t(\cdot)$, which is quadratic and convex because $Q \succ 0, R \succ 0$. Thus this inequality is a convex quadratic constraint in $(s, \{\mathbf{h}_t^j, \mathbf{M}_t^j, \mathbf{K}_t^j\}_{j=1}^J)$, which is a special case of SOC. Since the cost s is linear, we have that (4.22) is a SOCP. ■