Threat Assessment and Proactive Decision-Making for Crash Avoidance in Autonomous Vehicles

Vanshaj Khattar

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Masters of Science
in
Electrical Engineering

Azim Eskandarian, Chair
Harpreet Singh Dhillon
Thinh T. Doan

May 7, 2021
Blacksburg, Virginia

Keywords: Hidden Markov Models, Stochastic Reachable Sets, Stochastic Model Predictive Control, Threat Assessment

Threat Assessment and Proactive Decision-Making for Crash Avoidance in Autonomous Vehicles

Vanshaj Khattar

(ABSTRACT)

Threat assessment and reliable motion-prediction of surrounding vehicles are some of the major challenges encountered in autonomous vehicles' safe decision-making. Predicting a threat in advance can give an autonomous vehicle enough time to avoid crashes or near-crash situations. Most vehicles on roads are human-driven, making it challenging to predict their intentions and movements due to inherent uncertainty in their behaviors. Moreover, different driver behaviors pose different kinds of threats. Various driver behavior predictive models have been proposed in the literature for motion prediction. However, these models cannot be trusted entirely due to the human drivers' highly uncertain nature. This thesis proposes a novel trust-based driver behavior prediction and stochastic reachable set threat assessment methodology for various dangerous situations on the road. This trust-based methodology allows autonomous vehicles to quantify the degree of trust in their predictions to generate the probabilistically safest trajectory. This approach can be instrumental in the near-crash scenarios where no collision-free trajectory exists. Three different driving behaviors are considered: Normal, Aggressive, and Drowsy. Hidden Markov Models are used for driver behavior prediction. A "trust" in the detected driver is established by combining four driving features: Longitudinal acceleration, lateral acceleration, lane deviation, and velocity. A stochastic reachable set-based approach is used to model these three different driving behaviors. Two measures of threat are proposed: Current Threat and Short Term Prediction Threat which quantify present and the future probability of a crash. The proposed

threat assessment methodology resulted in a lower rate of false positives and negatives. This probabilistic threat assessment methodology is used to address the second challenge in autonomous vehicle safety: crash avoidance decision-making. This thesis presents a fast, proactive decision-making methodology based on Stochastic Model Predictive Control (SMPC). A proactive decision-making approach exploits the surrounding human-driven vehicles' intent to assess the future threat, which helps generate a safe trajectory in advance, unlike reactive decision-making approaches that do not account for the surrounding vehicles' future intent. The crash avoidance problem is formulated as a chance-constrained optimization problem to account for uncertainty in the surrounding vehicle's motion. These chance-constraints always ensure a minimum probabilistic safety of the autonomous vehicle by keeping the probability of crash below a predefined risk parameter. This thesis proposes a tractable and deterministic reformulation of these chance-constraints using convex hull formulation for a fast real-time implementation. The controller's performance is studied for different risk parameters used in the chance-constraint formulation. Simulation results show that the proposed control methodology can avoid crashes in most hazardous situations on the road.

Threat Assessment and Proactive Decision-Making for Crash Avoidance in Autonomous Vehicles

Vanshaj Khattar

(GENERAL AUDIENCE ABSTRACT)

Unexpected road situations frequently arise on the roads which leads to crashes. In an NHTSA study, it was reported that around 94% of car crashes could be attributed to driver errors and misjudgments. This could be attributed to drinking and driving, fatigue, or reckless driving on the roads. Full self-driving cars can significantly reduce the frequency of such accidents. Testing of self-driving cars has recently begun on certain roads, and it is estimated that one in ten cars will be self-driving by the year 2030. This means that these self-driving cars will need to operate in human-driven environments and interact with human-driven vehicles. Therefore, it is crucial for autonomous vehicles to understand the way humans drive on the road to avoid collisions and interact safely with human-driven vehicles on the road. Detecting a threat in advance and generating a safe trajectory for crash avoidance are some of the major challenges faced by autonomous vehicles. We have proposed a reliable decision-making algorithm for crash avoidance in autonomous vehicles. Our framework addresses two core challenges encountered in crash avoidance decision-making in autonomous vehicles: 1. The outside challenge: Reliable motion prediction of surrounding vehicles to continuously assess the threat to the autonomous vehicle. 2. The inside challenge: Generating a safe trajectory for the autonomous vehicle in case of future predicted threat. The outside challenge is to predict the motion of surrounding vehicles. This requires building a reliable model through which future evolution of their position states can be predicted. Building these models is not trivial, as the surrounding vehicles' motion depends on human

driver intentions and behaviors, which are highly uncertain. Various driver behavior predictive models have been proposed in the literature. However, most do not quantify trust in their predictions. We have proposed a trust-based driver behavior prediction method which combines all sensor measurements to output the probability (trust value) of a certain driver being "drowsy", "aggressive", or "normal". This method allows the autonomous vehicle to choose how much to trust a particular prediction. Once a picture is painted of surrounding vehicles, we can generate safe trajectories in advance – the inside challenge. Most existing approaches use stochastic optimal control methods, which are computationally expensive and impractical for fast real-time decision-making in crash scenarios. We have proposed a fast, proactive decision-making algorithm to generate crash avoidance trajectories based on Stochastic Model Predictive Control (SMPC). We reformulate the SMPC probabilistic constraints as deterministic constraints using convex hull formulation, allowing for faster real-time implementation. This deterministic SMPC implementation ensures in real-time that the vehicle maintains a minimum probabilistic safety

Acknowledgments

First and foremost, I would like to thank my advisor Dr. Azim Eskandarian for his invaluable guidance and support during this project. This thesis would not have been possible without his continuous guidance and inputs. I am grateful to him for giving me an opportunity to work on this exciting research project in the ASIM Lab. He always motivated me to achieve my best. I am also highly grateful to Dr. Harpreet Singh Dhillon for his support and understanding during my issue in taking ECE research credit hours. I would also like to thank Dr. Thinh Doan who has given me new perspectives on extending my research using reinforcement learning. I would also like to thank my ASIM labmates for their suggestions and valuable research discussions. Last but not the least, I am highly grateful for the unconditional support and love from my parents and my sister. They have always encouraged me to follow my passion. I am also fortunate and grateful to my girlfriend Manu for her constant support during my Masters and her valuable suggestions on my writing. This thesis would not have been possible without the support of all these people in my life.

Contents

List of Figures

List of Tables

List of Abbreviations

$\mathbb{N}$	Set of all Natural Numbers
$\mathbb{N}_+$	Set of positive Natural Numbers
$\mathbb{R}$	Set of Real Numbers
$Pr[.]$	Probability operator

Chapter 1

Introduction

1.1 Background and Motivation

According to a Global status report on road safety 2018 from World Health Organization[3], approximately 1 million people die worldwide in road accidents every year. Around 30 million people suffer from injuries resulting in long-term disabilities. This could be attributed to drinking and driving, getting sleepy or reckless driving on the roads. In an NHTSA study, it was reported that around 94% of car crashes could be attributed to driver errors and misjudgments. Therefore, it is crucial to detect anomalous driving behaviors on the road and react quickly and appropriately in a near-crash situation. Various Advanced Driver Assistance Systems(ADAS) have been developed to improve the safety of the cars. For example, methods like Lane Keeping Assistance(LKA), Collision warning systems, Adaptive cruise control, and blind angle vehicle detection are being extensively deployed to improve vehicles' safety.

Full self-driving cars can significantly reduce the accidents caused due to human errors and misjudgments. Self-driving cars have recently started being tested on certain roads, and it is estimated that one in 10 cars will be self-driving by the year 2030 [4]. This means that these self-driving cars will need to operate in human-driven environments and interact with human-driven vehicles. Therefore, it is crucial for autonomous vehicles to understand the way humans drive on the road to avoid collisions and interact safely with human-driven

vehicles on the road.

There are various kinds of dangerous drivers that drive on the roads, e.g., drowsy, drunk, or aggressive drivers. Each driver's behavior type poses a different threat to the autonomous vehicle, as each driver's behavior has different driving patterns. Various driver behavior predictive models have been proposed in the literature [5, 6] that can predict driver behaviors and intentions. The driver assistance systems use these models to assess future threat and intentions of the surrounding vehicles. As human behaviors are highly uncertain, these predictive models cannot be trusted entirely.

Moreover, many of these predictive models do not quantify the confidence in their driver behavior predictions. A wrong prediction can lead to a fatal crash with the surrounding vehicle. This makes it necessary to have a degree of trust in predictions that can assess the future threat. This is the first challenge addressed in this thesis, where a reliable motion prediction model of the surrounding vehicles is developed with a degree of trust in its predictions.

Reachability analysis [7] has been widely used for motion prediction of the surrounding vehicles. It gives information about all the surrounding vehicle's reachable states in a finite time horizon given the initial set of states. As the surrounding human-driven vehicles have uncertain inputs, stochastic reachable sets [7] can be used to represent the most likely occupancy regions of the surrounding vehicle. This information can be used by the motion planning module of the autonomous vehicle (ego vehicle) to keep its planned trajectories outside the stochastic reachable set areas.

Model Predictive Control (MPC) has been one of the most widely used planning and control strategies for autonomous vehicles operating in uncertain environments. MPC offers an advantage of simultaneous trajectory generation and control of the vehicle while handling the

environmental uncertainty and the constraints on vehicle dynamics. Robust MPC (RMPC) approaches have been previously used in the autonomous vehicles path planning [8]. RMPC uses deterministic and bounded descriptions of the system and environmental uncertainties. This bounded description can lead to large sets of occupancy regions of the surrounding vehicle, as it covers all the surrounding vehicles' possible movements. Using these large bounded reachable sets description of the surrounding vehicles in the MPC controller lead to conservative control action or infeasible solutions [9].

Stochastic Model Predictive Control (SMPC) approaches have recently proven to be highly effective for controlling systems in highly uncertain environments [10]. SMPC approaches can handle the system and the environmental uncertainty using probabilistic measures. SMPC approaches are able to use this probabilistic framework to formulate chance-constraints for the optimization problem. These chance-constraints guarantee constraint satisfaction with a minimum probability for the closed-loop performance. This leads to feasible and less conservative control actions. This chance-constrained framework can be highly effective for ensuring minimum probabilistic safety of autonomous vehicles in crash or near-crash scenarios. However, solving SMPC chance-constrained problems can be computationally expensive. This is the second challenge addressed in this thesis, where a fast SMPC framework for crash avoidance decision-making is proposed.

1.2 Problem Statement and Objectives

This thesis addresses the decision-making methodology for autonomous vehicles encountering various unexpected hazardous situations on the road. Real-time decision-making in autonomous vehicles involves addressing two core challenges:

1. **The outside challenge:** The motion prediction of the surrounding vehicles to continuously assess the threat to the autonomous vehicle (ego vehicle) from surrounding cars.

2. **The inside challenge:** Generating a safe trajectory for the autonomous vehicle in case of a future predicted threat.

Motion prediction requires building a reliable model of the motion of the surrounding vehicle through which their future evolution of position states can be predicted. Building these reliable models is not trivial as the surrounding vehicles' motion depends on human drivers' intentions and behaviors, which are highly uncertain. This thesis's first objective is to propose a reliable motion model of the surrounding vehicles with a degree of trust in its predictions. These motion predictions will allow formulating a measure of threat to the ego vehicle from the surrounding vehicles.

Predicting a possible crash (threat) in advance can give ego vehicle enough time to take action to avoid a crash. This is called proactive decision-making in autonomous vehicles. A proactive decision-making approach exploits the surrounding human-driven vehicles' intent to assess the future threat. This helps to generate a safe trajectory in advance, unlike reactive decision-making approaches[11] that do not account for the surrounding vehicles' future intent. This thesis's second objective is to propose a fast real-time proactive decision-making algorithm for crash avoidance in autonomous vehicles. A fallback trajectory has to be generated to avoid the crash with the surrounding vehicle or result in a minimum threat value.

1.3 Thesis Contributions

The motion prediction problem is solved by modeling the surrounding vehicles using driver behavior-based Stochastic Reachable (SR) sets. These SR sets allow formulating a surrounding driver's possible future movements depending on their predicted driving behavior. Three different driver behaviors are modeled: normal, aggressive, and drowsy. Each driver's behavior type has certain kinds of motion patterns and poses a different threat to the ego vehicle.

The second problem of generating a safe trajectory is solved using a proactive decision-making approach. A Stochastic Model Predictive Control Problem (SMPC) problem is formulated, which keeps the value of short-term prediction threat under a minimum probabilistic risk. A positive value of threat at any time instant implies that the ego vehicle must take a safety action to avoid a collision or crash. For a fast real-time implementation of the SMPC framework, it is reformulated as a deterministic MPC problem. The chance-constraints are converted to deterministic constraints using convex hull formulation. This convexification of the chance-constraints leads to a fast real-time implementation of the crash avoidance algorithm.

The contributions of this thesis are summarised as follows:

1. A trust-based driver behavior prediction method is proposed by combining all sensor measurements so that a degree of trust can be associated with the driver behavior prediction.

2. Driver behavior-based stochastic reachable sets are proposed, which consider the degree of "trust" of the predicted driver behavior. Three driver behaviors are considered: Normal, Aggressive, and Drowsy.

3. A Short Term Prediction Threat (STPT) measure is proposed, which gives information about the probability of collision at each time step instead of a combined probability of crash as done in [1].

4. Chance-constrained problem of Stochastic Model Predictive Control is solved by reformulating it in terms of deterministic convex hull constraints for a fast real-time implementation.

1.4 Thesis Outline

This thesis's overall goal is to design a real-time proactive decision-making methodology for crash avoidance in autonomous vehicles. Chapter 2 covers the literature review and preliminaries. It covers the recently proposed methodologies in motion prediction and motion planning of autonomous vehicles. Drawbacks and potential improvements in the proposed methods are highlighted. This chapter also covers the background of the three core concepts used in this thesis: Markov Chains, Hidden Markov Models, and Stochastic Model Predictive Control.

Chapter 3 covers the proposed stochastic reachable set threat assessment methodology using trust-based driver behavior prediction. Threat Assessment module architecture is presented. Chapter 3 also proposes two probabilistic threat assessment measures: Current Threat and the Short Term Prediction Threat.

Chapter 4 covers the proposed proactive decision-making algorithm for crash avoidance. Crash avoidance problem is formulated as a chance-constrained optimization problem using Stochastic Model Predictive Control (SMPC). A deterministic reformulation of the SMPC framework is presented using deterministic convex hull constraints.

Chapter 5 provides the validation and simulation results for the proposed threat assessment and decision-making methodologies. The control methodology is tested for three different hazardous road scenarios and different initial conditions. The advantages and limitations of the proposed methodology are highlighted. Chapter 6 concludes the thesis with final remarks and suggestions for future work.

Chapter 2

Literature Review and Preliminaries

2.1 Related Research

2.1.1 Motion Prediction and Threat Assessment

There has been extensive research on motion prediction and threat assessment for autonomous vehicles. A comprehensive review by Lefavre et al. [12] classified motion prediction models into three categories: 1.)Physics-based motion models; 2.) Maneuver-based; 3.) Interaction-aware motion models. Physics-based motion models are described using kinematic and dynamic motion models, which lead to computationally fast implementation of motion prediction of vehicles. Kinematic and dynamic bicycle model is an important model used in various motion prediction strategies [13, 14, 15]. The drawback of using physics-based models is that motion prediction can only be made for a short time horizon. Moreover, human driver behaviors and intentions are not incorporated in the models, leading to faulty predictions. Various maneuver-based motion models have been developed to generate a long-term prediction and include driver behaviors in the motion models. These models are mostly built using learning (like Hidden Markov Models[16], Gaussian Process Regression[17] etc.) and filtering techniques [18, 19]. These maneuver-based motion models have also been used for various anomaly detection methods to warn the driver of a risky driver behavior [20, 21]. The drawback with maneuver-based motion models is that they are

modeled only for specific road geometry and situations, making them less reliable to be used on different roads. Interaction-aware motion models account for the interaction between the ego vehicle and the surrounding vehicle and incorporate the dependence of the ego vehicle's actions on the surrounding vehicle's actions. This results in a more reliable motion prediction and risk assessment for traffic situations. In [22], a multiple model Kalman filtering scheme was used to estimate the movement of multiple vehicles using a hierarchical approach based on a priority list. [23] proposed a data-driven interaction-aware motion model for pedestrian environments and static obstacles. The major drawback with the interaction-aware motion models is their increasing computational complexity with the increasing number of traffic participants. This makes these algorithms incompatible for real-time implementation on roads.

A review by Li et al. [24] categorizes threat assessment metrics into five types. First are the time-based (TTX) metrics which use time as a measure to quantify threat. Time to Collision (TTC) [25], Time to Reaction (TTR) [26] and Time to Headway (THW) [27] are some of the earliest proposed threat assessment methodologies. These methods do not account for the geometry of the roads and can give false positives. Second are the kinematic-based metrics that use distance [28], velocity and acceleration [29] for assessing the threat around the vehicle. These kinematic metrics are not reliable as they can result in many false positives due to the constant model parameters. The third category is the probabilistic-based threat metric which quantifies the threat using the probability of collision [1]. These methods could be improved using offline computations and machine learning techniques. The fourth category is the potential-field-based metrics [30] which consider surrounding vehicles as a repulsive field. The fifth category of threat assessment metrics is based on unexpected driver behavior-based metrics, which rely on detecting anomalous driver behaviors [31, 32, 33] for threat assessment. The drawback of these methods is that they do not quantify the risk,

and there are many technical challenges in anomaly detection.

Combining driver behavior prediction with motion prediction models has been the recent focus for reliable threat assessment for autonomous vehicles [34, 35]. However, still, these methods do not provide any confidence or trust measures on the predicted driver behavior. Driver behaviors are highly uncertain, and no motion model is perfect.

We propose a novel trust-based driver behavior prediction method used to build driver behavior-based stochastic reachable sets. Most driver behavior prediction methods focus on anomalous driver behaviors but do not quantify the measure of threat from the anomalous driver. Our motion prediction approach is inspired from [1], where surrounding vehicle dynamics are represented as Markov Chains. We extend their method to driver behavior-based stochastic reachable sets where a degree of trust of predicted driver behavior is used to represent the Stochastic reachable sets.

2.1.2 Motion Planning and Proactive Decision-Making in Autonomous Vehicles

Various motion planning algorithms have been proposed in the literature for safe navigation and collision avoidance systems. An extensive review by Gonzalez et.al.[36] covers modern motion planning and decision-making techniques for autonomous vehicles. They classified the motion planning techniques into the following categories: interpolating curve-based planning, graph-search planners, sample-based planners, and optimization planners.

Interpolating curve-based planners use a set of control points to generate a trajectory. Some examples of interpolating path planners are Bezier curves [11, 37], spline curves [12], clothoid curves [38] and polynomial curves [39]. These planners have been widely used in the DARPA urban challenge. The drawback of these methods is that they do not consider the dynamic

constraints on the vehicle. Moreover, they can only be used for reactive path planning and not proactive path planning. This makes them a weak choice for urban environments.

Graph search-based planners divide the vehicle's planning space into a grid space and plan the trajectory from the initial grid to the destination grid. This trajectory planning is done using the Dijkstra's algorithm, D algorithm [40], A algorithm [41]. These methods are usually used for global path planning and are computationally expensive for higher-dimensional problems.

Sampling-based methods sample the planning state space into many vertices and keep adding these vertices until the goal is reached. Approaches like Rapidly-exploring Random Tree (RRT) [42] and RRT^* [43] create a tree-like map that keeps expanding stochastically until the goal is reached. The drawback with these methods is that there can be many iterations until an optimal path is reached. These methods cannot be used for sudden unexpected hazardous situations where a vehicle has to react quickly.

Another kind of motion planning technique is the potential-field-based approach. The main idea in these approaches is that goal is represented by the attractive forces, and repulsive forces represent the obstacles. The resultant force results in a final trajectory towards the goal. Ji et al. proposed cosine potential-fields for lanes, and exponential potential-fields for the surrounding vehicles [44]. Wang et al. used the Artificial potential-field method combined with MPC for safe decision-making in autonomous vehicles [45]. One of the drawbacks of the potential-field methods is the local minima problem [46] which makes the ego vehicle or robot oscillate infinitely between obstacles. This occurs due to the resultant high degree of non-linearity of the potential fields of all obstacles. Various approaches have been proposed to overcome this, like improved-APF [47], Hybrid Fuzzy potential field [48]. Most of these approaches are only applicable in stringent conditions and are not very flexible.

With the rise of sensing and computational capabilities, optimization-based or MPC-based

motion planning techniques have shown immense potential for autonomous vehicles in uncertain environments [49, 50]. These approaches consider the constraints on vehicle dynamics, environmental uncertainty, safety constraints, and actuator constraints to generate an optimal trajectory. However, these approaches deal with the uncertainty using worst-case measures, leading to conservative or infeasible solutions. Recently Stochastic MPC (SMPC) approaches have been proposed for handling uncertainty using probabilistic measures so that feasible solutions can be computed. SMPC allows for a chance-constrained optimization problem formulation where minimum probabilistic safety guarantees are provided during the operation of the system. Some of the works that have used SMPC for autonomous vehicle decision-making are covered in [10, 51, 52].

This thesis focuses on developing an SMPC solution based on a stochastic reachable set threat assessment approach for avoiding crashes with the surrounding vehicles. The crash avoidance problem is formulated as an SMPC problem where chance-constraints are specified to account for uncertainty in the surrounding vehicle's motion. These chance-constraints always ensure a minimum probabilistic safety of the autonomous vehicle by keeping the probability of crash below a predefined risk parameter. For a fast-real time implementation during the crash scenarios, the SMPC chance-constrained formulation will be transformed to a deterministic reformulation.

2.2 Preliminaries

2.2.1 Markov Chains

Markov chains are examples of a stochastic process with discrete states as random variables $Y \in \mathbb{N}^+$. In this thesis, discrete-time Markov chains are considered for the pro-

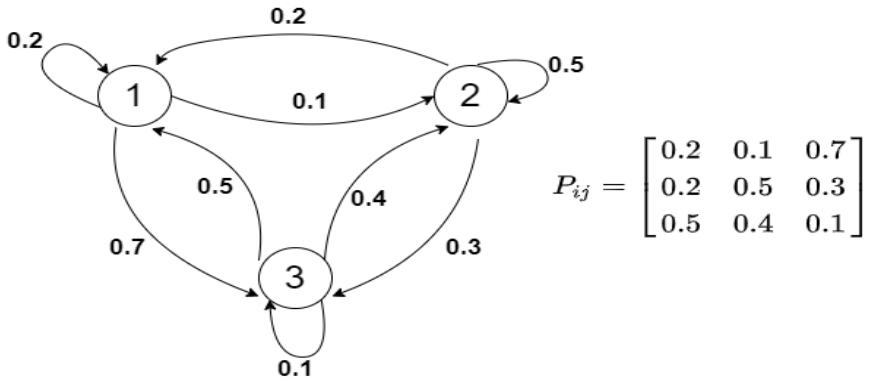


Figure 2.1: An example of a Markov Chain with 3 discrete states and its transition probability graph

posed motion prediction methodology. Markov chains are a collection of random variables $Y = \{Y_0, Y_1, Y_2, ...\}$.

A stochastic process Y is called a Markov chain if it follows the Markov property for all time steps k:

$$\mathbb{P}(Y_k = j | Y_{k-1} = i, Y_{k-2} = i_{k-2}, \cdots, Y_0 = i_0) = \mathbb{P}(Y_k = j | Y_{k-1} = i) \triangleq P_{ij} \tag{2.1}$$

The Markov property implies that the probability distribution at a future time step $k+1$ depends only on the current probability distribution at time step k. This implies that the future distributions are independent of the past probability distributions. If an initial probability distribution for discrete states of the Markov chains is given as p^0, then the future evolution of probability distributions can be found using:

$$p(k+1) = P_{ij} p(k) \tag{2.2}$$

Where P_{ij} is called the transition probability matrix for the Markov chain. An example of a Markov chain is given in Figure 2.1.

2.2.2 Stochastic Reachable Sets

Reachability analysis [7] gives information about all the possible set of states that a system can achieve, given the initial set of states and disturbances of a system. Information about the reachable sets of the surrounding vehicles can help in motion planning for the ego vehicle. The major drawback of using reachable sets is that they are usually very large as they cover almost all the possible states that a vehicle can achieve. This leads to infeasible and conservative motion planning for the ego vehicle, making it impractical to be used for real-time motion planning.

Stochastic reachability analysis addresses this problem of large reachable sets by assigning probability values to the states in the reachable set. This information can help the ego vehicle to know the probability of reaching a particular state at some time step. The reachable states of the surrounding vehicle with low values of probability can be used in the ego vehicle's motion planning space. Moreover, probability of crash and probabilistic safety guarantees can be provided for the ego vehicle. This can help overcome the drawbacks of using simple reachability analysis, i.e., conservative behavior and infeasible solutions for the ego vehicle.

Stochastic reachable sets of a system are defined by the probability of a system to be in a certain state at some time step. They are represented using the probability distribution function over the system's reachable states for some time step k. Let there be a random dynamical system with the random state vector X. Stochastic reachable set at time step k can be represented using the following probability distribution function:

$$f_X(x, t = k) = Pr(X = x) \qquad \text{at time step } k \tag{2.3}$$

Where, $Pr(.)$ is the probability operator. The stochastic reachable set includes all the states

where the probability distribution function is positive. Figure 2.2 shows a visual representation of the stochastic reachable sets at different time steps for a surrounding vehicle.

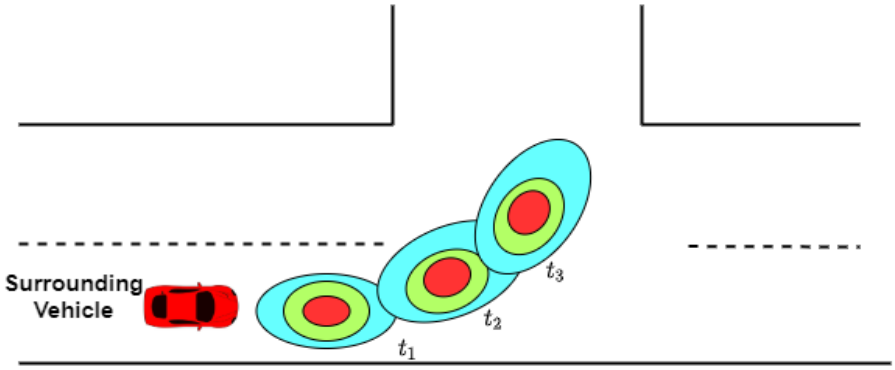


Figure 2.2: Visualization of the Stochastic Reachable sets at different time steps. Adapted from [1]

2.2.3 Hidden Markov Models

Hidden Markov Models (HMM) are statistical Markov models that can predict the hidden information from an observable sequence of states of a stochastic system. The hidden information is termed as hidden states. The stochastic processes considered in the HMM are assumed to be Markov processes, i.e., the future state only depends on the present state. HMM is widely used in modeling and reasoning about the stochastic systems (Markov) whose states cannot be observed directly. Other observable states of the system are used to infer the sequence of hidden states that the given Markov system went through. The non-observable sequence is termed as the Hidden states, and the observable sequence is termed as the Emission states. Hidden Markov Models are widely used in biomedical engineering, audio signal processing, statistical-mechanics, pattern recognition, time-series prediction, and countless others. Figure 2.3 shows the representation of a Hidden Markov Model.

A HMM is defined using following quantities:

- T = Length of the observed sequence
- $S = \{S_0, S_1, \cdots, S_T\}$ is the set of hidden states that a HMM went through
- $O = \{O_0, O_1, \cdots, O_T\}$ is the set of emitted sequence or the observable states of the stochastic system
- $\pi : S \to [0, 1]$ is the initial probability distribution of the hidden states, which provides the probability of starting in each state.
- P is the transition probability matrix for the evolution of hidden states
- E is the emission probability matrix which stores the probability of observing a certain state when the Markov system is in a certain hidden state

Two main assumptions in the HMM are the Markov property assumption and the output independence assumption. The output independence assumption suggests that the observed current output at time t is only dependent on the previous hidden state and independent of previous observations. The main question that will be solved in this thesis using HMM is to decode the sequence of hidden states that a stochastic system went through to generate the observed sequence.

Viterbi algorithm [53] is the famous solution for finding the optimal sequence of hidden states through an observed sequence. A dynamic programming approach is used to find the most likely sequence of hidden states S when an observed sequence O is given.

In this thesis, a surrounding vehicle's driver behavior cannot be directly observed from on-board vehicle sensor measurements, but the vehicle states can be observed. Therefore, vehicle state observations are used to infer the most likely behavior of the surrounding vehicle's driver.

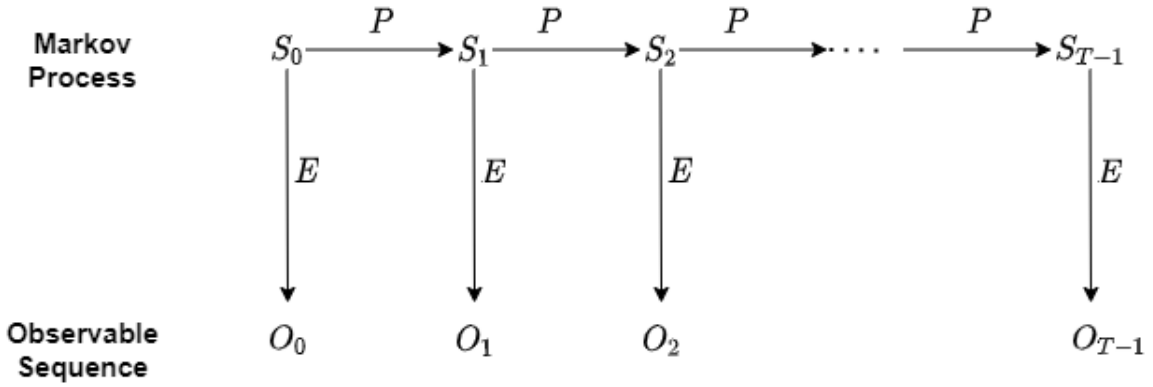


Figure 2.3: Hidden Markov Model, adapted from [2]

2.2.4 Stochastic Model Predictive Control

Model Predictive Control (MPC) has been one of the most widely used optimal control strategies for complex systems with multiple inputs, outputs, and constraints [54]. Robust implementation using MPC's receding horizon control framework handles the uncertainty in the systems by assuming they lie in a bounded set. This deterministic formulation of the uncertainty leads to over-conservative or infeasible solutions due to the bounded set approach's worst-case considerations.

Most of the uncertainty in the system and the environment is probabilistic. Stochastic Model Predictive Control is widely used to handle the system's uncertainty using probabilistic measures and chance-constraints. The cost function for the MPC is satisfied with a minimum probability, which ensures a minimum probabilistic constraint satisfaction during the optimal control.

To formulate a general SMPC problem, consider a discrete time system given by:

$$\begin{aligned} x_{k+1} &= f(x_k, u_k, w_k) \\ y_k &= h(x_k, u_k, v_k) \end{aligned} \tag{2.4}$$

where $k \in \mathbb{N}_+$, $x_k \in \mathbb{R}^{N_x}$, $y_k \in \mathbb{R}^{N_y}$ and $u_k \in \mathbb{U} \subset \mathbb{R}^M$ are the systems states, system outputs and inputs to the system respectively. $\mathbb{U}$ is the measurable set for the inputs, N_x is the number of states, N_y is the number of outputs and M is the number of inputs to the system. w_k and v_k are the system disturbances and sensor noises respectively. f and h are the nonlinear functions describing the dynamics of the system and its output respectively.

Let $N \in \mathbb{N}$ be the prediction horizon for the receding horizon control framework. Let the feedback control policy be:

$$u := \{u_0, u_1, u_2, \cdots u_{N-1}\} \tag{2.5}$$

For a certain control policy u, the cost function for a SMPC problem can be written as:

$$J_N(x_k, u) = \mathbb{E}\left[\sum_{i=0}^{N-1} J_c(\hat{x}_i, u_i) + J_T(\hat{x}_N)\right] \tag{2.6}$$

where $J_N(x_k, u)$ is the total cost of operation using a feedback control policy u. J_c and J_T are the per-stage cost function and terminal cost function at the end of the prediction horizon, respectively. $\hat{x}_i$ represents the predicted systems state at time i where the disturbance is given by w_j where $j \in [0, i-1]$. The minimization of the above cost function is subject to the chance-constraints on the system outputs and states. Let $\hat{y}_i$ be a predicted output at the time i; then the joint chance constraints can be written as [55]:

$$Pr[g_j(\hat{y}_i) \leq 0] \geq \beta_j \text{ for all } j = 1, 2, \cdots s \ ; i = 1, \cdots, N \tag{2.7}$$

where $g_j : \mathbb{R}^{N_y} \to \mathbb{R}$ is some linear or non-linear Borel-measurable function, s is the total number of inequality constraints and $\beta \in [0, 1]$ is the minimum threshold probability value with which the probabilistic constraints $g_j(\hat{y}_i) \leq 0$ have to be satisfied for all $j = 1, \cdots, s$.

The overall SMPC formulation using the system dynamics, cost function and chance constraints can be written as follows:

$$\begin{aligned} J_N^\star(x_k) &:= \min_u J_N(x_k, u) \\ &\text{such that:} \\ \hat{x}_{i+1} &= f(\hat{x}_i, u_i, w_i), && \text{for all i } \in [0, N-1] \\ \hat{y}_i &= h(\hat{x}_i, u_i), && \text{for all i } \in [0, N] \\ u_i &\in \mathbb{U}, && \text{for all i } \in [0, N-1] \\ Pr[g_j(\hat{y}_i) \leq 0) &\text{for all j} = 1, \cdots, \text{s}] \geq \beta, && \text{for all i } \in [1, N] \end{aligned} \tag{2.8}$$

where $J_N^\star(x_k)$ is the optimal cost function for the optimal control policy $u^\star$. The initial state is known to be x_0.

This probabilistic formulation of constraints and the model predictive control allows for the slight violation of constraint satisfaction, resulting in more feasible actions and does not lead to too much conservative behaviors of the system. Moreover, the external uncertainties arising in the system can be appropriately handled using probabilistic inequalities. A more extensive overview, state of the art, and applications for SMPC are covered in the review by Mesbah [55].

2.3 Summary

This chapter reviewed the recent literature on the motion prediction and decision-making algorithms in autonomous vehicles. Section 2.1 highlights the advantages and drawbacks of the current methodologies for crash avoidance in autonomous vehicles. The four core concepts were reviewed in Section 2.2 on which this thesis is based, i.e., Markov chains and Stochastic Reachable sets, Hidden Markov Models, and Stochastic Model Predictive Control.

Chapter 3

Stochastic Reachable Set-Based Threat Assessment

3.1 Introduction

This chapter introduces a novel threat assessment methodology for autonomous vehicles based on stochastic reachable (SR) sets. Driver behavior-based SR sets are introduced for three different driver behaviors: Normal, Aggressive, and Drowsy. Each driver's behavior poses a different threat to the ego vehicle due to each driver's behavior type's different motion patterns. A novel "trust" based method is proposed to quantify the confidence in the predicted driver behavior. Two threat assessment methods are proposed based on the probability of crash: Current Threat (CT) and Short Term Prediction Threat (STPT).

3.2 Threat Assessment Module Architecture

Figure 3.1 shows the proposed threat assessment methodology for autonomous vehicles. The autonomous car's perception module is used for sensor measurements of different features of the surrounding vehicle. These sensor measurements are fed into the intention estimation module and trust-based driver behavior prediction module. Surrounding vehicle dynamics

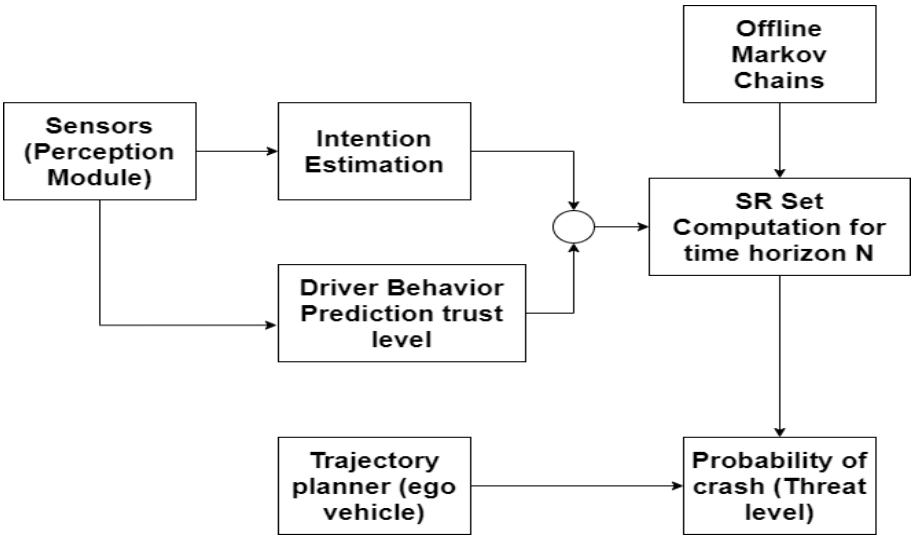


Figure 3.1: Threat Assessment Module Architecture

representation using Markov chains is done offline by generating state and input transition probability matrices for different driver behaviors. Using the information from all the above three modules, driver behavior-based Stochastic Reachable sets are computed for a finite time horizon N. Stochastic reachable sets' information allows for assessing the risk at different time points using the probability of a crash.

3.3 Driver Behavior Prediction and intention estimation

The first stage in assessing the threat around our autonomous car (ego vehicle) is to detect the surrounding vehicle's driver behavior. This allows formulating unique responses to each driver type for any dangerous situation encountered on the road.

A driver behavior dataset UAH-DriveSet[56] is used to build the models for three different driving behaviors: Normal, Aggressive, and Drowsy. The dataset [56] consists of six drivers

emulating all three driving behaviors on two types of roads: motorway and secondary roads. The dataset consists of more than 500 minutes of naturalistic driving data in Madrid city, Spain.

In this thesis, we propose that ego vehicle's on-board vehicle sensor measurements can be used to predict the surrounding vehicle's driver behavior. We assume that all these sensor measurements are available at each time step to the autonomous vehicle. Four sensor measurements or features are used for driver behavior prediction, i.e., Longitudinal Acceleration, Lateral Acceleration, Lane Deviation or weaving, and Velocity. Hidden Markov Model (HMM) technique is applied to each of these four features time-series data collected from UAH-DriveSet. The UAH-Drive dataset was collected at a frequency of 10 Hz. The HMM outputs the most probable sequence of hidden states that a system went through to produce the detected features. In our case, the hidden states are the driver behavior that are producing the detected features. In conclusion, the ego vehicle senses the variables that puts the driver behavior into categories.

3.3.1 HMM Model Training

The HMM models are trained using the time-series data of 4 given features for three driver behaviors obtained from the UAH-DriveSet. Out of 6 drivers, data from the first four drivers is used for training the HMM models. Time-series data corresponding to each feature is normalized to the integer values as HMM only accepts integer values as input. Each of the features time-series data is associated with a certain type of driver behavior. These labels are available from the dataset for supervised training of the HMM. The following three labels or hidden states are defined for the HMM models:

1. B1: Normal behavior

2. B2: Aggressive behavior

3. B3: Drowsy behavior

The Machine Learning Toolbox in MATLAB is used to generate 4 HMM models for each of the four features acting as emission states. These emission states are used to predict driving behavior type. Let X_f be a normalized time series data for a certain feature for all three types of driving behaviors. X_f acts as observations for the training of HMM:

$O_{f,i}$ for f = 1,2,3,4 and $i \in [1, N]$

Where f are the detected features, i is the time step of the data, N is the total number of data time steps, and $O_{f,i}$ is the observation for feature f at time step i. All these observations are taken from the first four drivers' dataset and are labeled accordingly with respective driver behaviors. Each of the four HMM models for each feature has two parameters, i.e., the transition probability matrix and the emission probability matrix. These parameters are computed using the labeled data, observed time-series data $O_{f,i}$ and the Maximum Likelihood Algorithm[57].

The transition probability matrix gives the state transition probability matrix for the three hidden states: driver behaviors. The emission probability matrix gives the probability that a certain state $O_{f,i}$ was observed given the system was in hidden state B_j for $j = 1, 2, 3$. These parameters are used in driver behavior prediction of other unseen observed time-series data.

It has to be noted that, the driving behaviors in the given driver-behavior dataset are quite subjective and have led to a stationary Markov Chain for this experiment. In the real-time scenarios, this assumption of stationarity in the underlying Markov chain might not be entirely applicable. With a bigger driver behavior dataset, we can get closer to the real time scenarios and estimating the underlying behavior transition Markov Chain.

3.3.2 HMM Model Testing

The 4 HMM models for each feature were tested on the data from the remaining two drivers. A time window of 2 seconds was applied for the prediction of driver behavior of the surrounding vehicle. The surrounding vehicle features for the last 2 second time window are considered and were fed as an input to each of the trained HMM. The other inputs to the HMM are the transition probability matrix and the emission matrix computed from each feature's dataset and driver behavior.

An optimization problem is solved for each time window, to infer the most likely driver behavior,

Let $O_{f,i} = o_{f,1}, o_{f,2}, o_{f,3}.....o_{f,T}$ be the observed sequence of feature f of the surrounding car, where T is the time window of 2 seconds. Observations of all four features are fed into each trained HMM model, and their respective transition probability and emission matrices are computed offline.

The following unconstrained optimization problem is solved online using the Viterbi Algorithm[53] for each feature's respectively trained HMM:

$$Z_f^\star(k) = \underset{Z_f}{argmax} \;\; P(Z|O_{f,2}) \quad \text{for f = 1, 2, 3 and 4} \tag{3.1}$$

Where $Z_f^\star$ is the optimal sequence of driver behaviors predicted from the measured data $O_{f,2}$ at time k for feature f over the previous 2 second time window. A sliding time window technique, with a one-time step shift at each iteration, is used to predict the driver behavior continuously. To infer the driver behavior over each time window, a trust ratio (TR) is computed for each driver behavior from each predicted optimal sequence $Z_f^\star$. This is done by counting the number of predicted states of a particular behavior in the optimal sequence

Table 3.1: Driver Behavior Prediction Accuracy (trust values) for all 4 features

	Normal Driving(S1)	**Aggressive Driving(S2)**	**Drowsy Driving(S3)**
Longitudinal Acceleration	91%	83%	89%
Lateral Acceleration	78%	76%	81%
Lane Deviation	93%	78%	90%
Velocity	79%	88%	75%

$Z_f^\star$, by the total number of time steps in the time window:

$$TR_{f,b} = \frac{\text{No. of predicted states of behavior b } Z_f^\star}{\text{No. of time steps in 2 sec time window}} \tag{3.2}$$

Where b = 1, 2, and 3 for three driver behaviors and f is for all the four detected features over the 2 second time window. At each time step, 12 TR ratios will be computed for each feature and driver behavior. These ratios are used for online trust-based driver behavior prediction by fusing all the features, as shown in the next section.

Each feature's behavior prediction accuracy was computed using the ground truth behavior labels in the remaining two drivers dataset. This behavior prediction accuracy for each feature can be interpreted as a trust value for that feature.

The trust values or prediction accuracy for each feature and each driving behavior for the remaining two drivers are given in Table 3.1.

3.3.3 Reliable online driver-behavior prediction using all the features

The values in Table 3.1 can be interpreted as the trust that we can have in each feature's driver behavior predictions. These are termed as the trust weights for each feature or sensor measurement. We propose an online driver behavior prediction method that combines the Trusted Ratios from eq. (2) of predicted behaviors from all 4 features with their trust weight values. This results in Online Trust Ratios for each driver behavior given by the following formulas:

$$OTR_N = \frac{w_{1N}TR_{1,N} + w_{2N}TR_{2,N} + w_{3N}TR_{3,N} + w_{4N}TR_{4,N}}{w_{1N} + w_{2N} + w_{3N} + w_{4N}} \tag{3.3}$$

$$OTR_A = \frac{w_{1A}TR_{1,A} + w_{2A}TR_{2,A} + w_{3A}TR_{3,A} + w_{4A}TR_{4,A}}{w_{1A} + w_{2A} + w_{3A} + w_{4A}} \tag{3.4}$$

$$OTR_D = \frac{w_{1D}TR_{1,D} + w_{2D}TR_{2,D} + w_{3D}TR_{3,D} + w_{4D}TR_{4,D}}{w_{1D} + w_{2D} + w_{3D} + w_{4D}} \tag{3.5}$$

where OTR is the Online Trust Ratio for driver behaviors Normal (N), Aggressive (A), and Drowsy (D) behavior, respectively. $w_{f,b}$ are the trust weights from Table I for each feature f and driving behavior b. $TR_{f,b}$ are individual Trust Ratios (TR) of predicted driving behaviors b and feature f using eq. (2). The architecture of the proposed scheme is given in Figure 3.2.

This real-time detection procedure is done using the 2-second shifting time window technique, where TR values are obtained for all four features. This method's online implementation average time for every time window is 5 ms, making this method very useful for real-time safety applications.

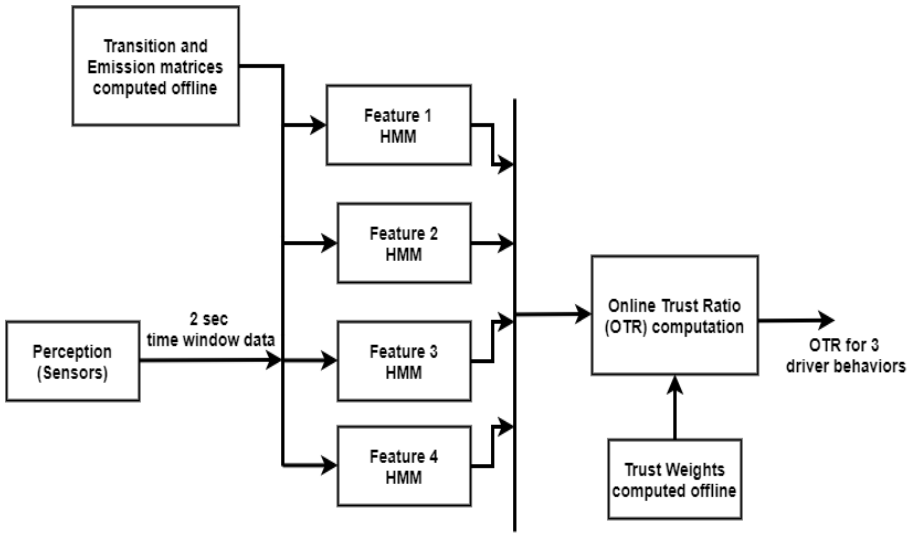


Figure 3.2: Architecture for online trust based driver behavior prediction

3.3.4 Intention estimation using detected features

Other than driver behavior prediction, intention estimation of surrounding vehicles is also necessary to predict contradicting intentions with the ego vehicle. This information of intention estimation is necessary for proactive decision-making in autonomous vehicles.

Intention estimation is done synchronously with the driver behavior prediction computation. Four intentions are being considered for the surrounding vehicle, i.e., going straight, lane change, right turn, and left turn.

The training method used in section 3.3.1, is also used to train the 4 HMM models for intention estimation of the surrounding vehicle. Four HMM models are built for each of the observed feature time-series data. The labels and timestamps of different intentions are available from the UAH driver dataset for four different vehicle intentions. The HMM parameters: transition probability matrix and emission probability matrix are built obtained using the maximum likelihood algorithm and data from the first four drivers in the dataset. The testing of these HMM models is done on the remaining two drivers in the dataset.

Prediction results were obtained and compared to the ground truths available from the

Table 3.2: Driver Intention Prediction Accuracy for all 4 features

	Going Straight	**Lane Change**	**Right turn**	**Left turn**
Longitudinal Acceleration	89%	83%	92%	95%
Lateral Acceleration	76%	84%	79%	69%
Lane Deviation	93%	88%	90%	83%
Velocity	81%	91%	77%	72%

UAH driver dataset, and prediction accuracy for different features in intention estimation are shown in Table 3.2

The average computation time for synchronous intention estimation and driver behavior prediction is 12 ms. We can observe from the table 3.2, that longitudinal acceleration has the maximum average prediction accuracy of 89.75% for all four intentions. Therefore, in this thesis, longitudinal acceleration time-series data will be used for intention estimation of the surrounding vehicle.

3.4 Stochastic Reachable sets for different driver behaviors

Stochastic reachable (SR) sets allow the formulation of a detected vehicle's most likely occupancy regions for a finite time horizon. This can help determine the probability of a crash of ego vehicle with the surrounding vehicle when a threat has been detected. These SR sets of the surrounding vehicle act as the unsafe set for the ego vehicle that has to be avoided. The ego vehicle can determine the best control action using the surrounding vehicle SR set

information so that the probability of a crash is minimized.

In this thesis, the SR sets of the surrounding vehicle are represented using probability distribution over different cross sections of the road as follows:

$$f(x, y) = Pr[x \in X, y \in Y] \quad \text{for } X := [x_1, x_2]; Y := [y_1, y_2] \tag{3.6}$$

where $Pr(.)$ is the probability operator; x_1, x_2 and y_1, y_2 are lower and upper limits of longitudinal and lateral positions respectively. These are also called the rectangular covering functions, which will be discussed at the end of this section. This is done using the random matrices and representing the surrounding vehicle dynamics and the driver inputs as Markov chains. This idea is inspired by [1], and our implementation extends it to driver behavior-based stochastic reachable sets. This gives an SR set that is dependent on the trust level of our driver behavior prediction. The second advantage of the given method is that the computationally intensive Markov chain computations are done offline.

These Markov chains are built offline and can be loaded into the ego vehicle for online threat assessment on the roads. The random matrices computed are converted to Markovian random sets[58] which allows expressing the evolution of the SR sets over time exclusively in terms of the SR set computed for the previous time step.

To incorporate the driver behavior into the SR sets, Online Trust Values of predicted driver behavior are used to represent the Markov random sets. These SR sets are termed as driver behavior-based stochastic reachable sets. The computation of the driver behavior-based SR sets is considered in the following sections.

3.4.1 Surrounding vehicle dynamics representation using Markov Chains

Markov chains are an example of a stochastic dynamic system with discrete states. To represent the surrounding vehicle dynamics as Markov Chains, firstly, the state space is discretized using the cell partitions as done in[1]. The second step for the Markov chain representation is the computation of the transition probability matrices for the system states and driver inputs.

The evolution of the detected vehicle states depends on two factors: 1) Longitudinal dynamics of the surrounding vehicle; 2) Applied throttle and lane deviation of the vehicle, which depends on the driver behavior.

The following hybrid dynamical system[59] is used to determine the probability distribution for the longitudinal dynamics of surrounding vehicle to be represented as a Markov chain:

$$\dot{x} = v \qquad \dot{v} = \begin{cases} a^{max}u, & 0 < v \leq v^{sw}\ OR\ u \leq 0 \\ a^{max}\frac{v^{sw}}{v}u, & v > v^{sw}\ AND\ u > 0 \\ 0, & v \leq 0 \end{cases} \tag{3.7}$$

subject to the constraint:

$$|a| \leq a^{max},\ \text{where}\ |a| = \sqrt{a_N^2 + a_T^2},\ a_N = \frac{v^2}{r(x)},\ a_T = \dot{v} \tag{3.8}$$

Where x is the longitudinal position of the vehicle, a_{max} is the maximum possible acceleration due to the tire friction, and v^{sw} is the velocity at which the car dynamics change due to the aerodynamic drag[59]; $r(x)$ is the radius of curvature of the path followed, and u is the

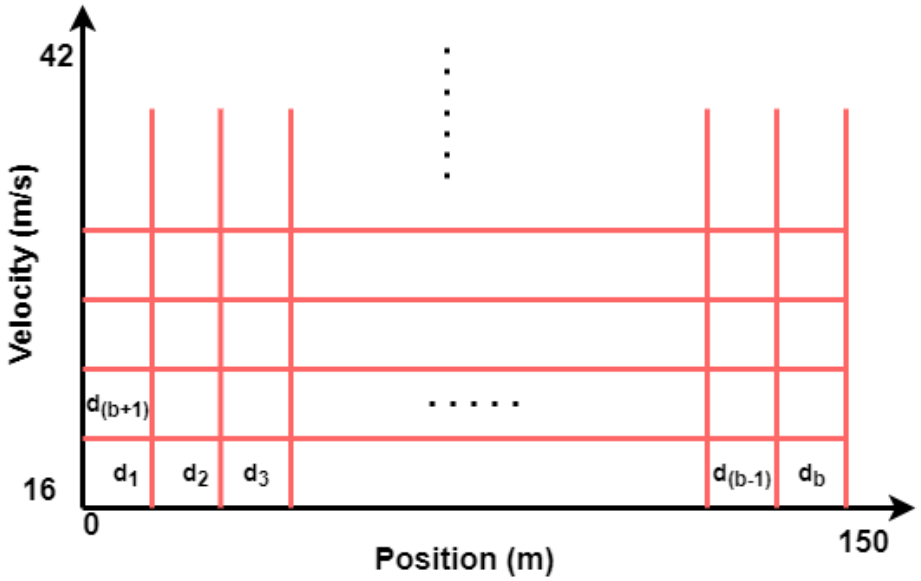


Figure 3.3: State space discretization into $D = ab$ discrete states with a and b partitions on velocity and position

applied acceleration by the driver.

The probability distribution (pdf) for longitudinal dynamics is represented using $f(x)$, where x is the vehicle's detected position.

For Markov Chain representation, the state space and the surrounding vehicle dynamics' input space is discretized as a first step. State-space of the detected vehicle dynamics consists of x (position) and v (velocity). In this study, threat assessment is considered for highway road scenarios, the velocity interval $[16 - 42]m/s$ is selected. This velocity range is divided into a segments. Similarly, position interval can be selected based on the on-board sensor's range. $[0, 150]m$ position interval is considered and divided into b segments. This results in a 2-dimensional discrete state space with $ab = D$ discrete states. Each discrete state d_i is a cell containing a range of velocities and longitudinal positions as shown in Figure 3.3.

For Markov Chain representation of the surrounding vehicle dynamics, transition probability matrix for discrete states D is computed. The transition probability matrix for the evolution

of states from d_i to d_j is computed by applying Monte Carlo simulations to the dynamic model of the surrounding vehicle:

$$T_{ij}^{u_n} = \frac{\text{No. of simulations reaching state } d_j \text{ from state } d_i}{\text{Total No. of simulations started from state } d_i} \tag{3.9}$$

where $T_{i,j}^{u}$ is the state transition probability matrix for a certain input u_n. $T_{ij}^{u_n}$ is used to compute the probability distribution of the next state $p(k+1)$ of the vehicle over the discrete cell divisions when input u_n is applied:

$$p(k+1) = T_{ij}^{u_n} p(k) \tag{3.10}$$

where the entries of the random vector $p(k)$ represent the probabilities of detected vehicle to be in a certain discrete state d_i for $i \in [1, D]$.

The size of $T_{ij}^{u_n}$ for a certain input u_n will be $D \times D$ and size of $p(k)$ will be $D \times 1$ as shown:

$$p(k) = [p_1(k) \;\; p_2(k) \;\; .... \;\; p_D(k)]^T \tag{3.11}$$

where $p_i(k)$ is the probability of being in a discrete state d_i defined by a certain velocity and a position range. Our eventual goal is to determine the probability distribution of the position of the vehicle for threat assessment. The probabilities from the $p(k)$ vector can be transformed to a probability vector $\widetilde{p}_n(k)$ for longitudinal position distribution in 0 to $150m$. $\widetilde{p}(k)$ will have b entries in our case as 150m is divided into b cells. $\widetilde{p}(k)$ can be calculated by taking average of the sum of each velocity cell probability in the same longitudinal cell range:

$$\widetilde{p}_n(k) = \frac{\sum_{i=0}^{a-1} p_{(bi+n)}(k)}{a} \tag{3.12}$$

where $\widetilde{p}_n(k)$ is the n^{th} entry of the new probability vector; a and b are the respective divisions made in velocity and position range. The size of final $\widetilde{p}_n(k)$ will be $b \times 1$.

The main idea is to get probability vectors $p(k)$ for a fixed prediction horizon N using Eqn. 3.10. But the Eqn. 3.10 assumes known deterministic inputs u_n for $n \in [1\ \ 2\ \ .....\ \ N]$ of the detected driver for the prediction horizon N. This will not be the case as the surrounding drivers' acceleration and steering inputs are unknown and driver behavior dependent. The stochastic inputs and their transition probability matrices have to be incorporated in the above computation. Input transition probability matrices generated from driver behavior models are discussed in the next subsection.

3.4.2 Driver behavior modeling

Surrounding vehicle's dynamics also depend on the applied acceleration and the lane deviation of the vehicle. These two quantities are dependent on the driver's behavior and their motivations. To account for transitions in the inputs during a prediction horizon N, probability distributions and input transition probability matrices are computed using the UAH-driver dataset. To model driver behavior in the SR sets, the probability distribution of lane deviations is considered, accounting for steering input changes. The second input: acceleration command, is divided into discrete states/cells, and transition probability matrices are computed for different driver behaviors using the UAH-drive dataset.

Lane Deviation (Weaving)

Lane deviation around the center of the road represents the lateral dynamics of the surrounding vehicle. The probability distribution for lane deviation is represented using $f(\delta)$ using piecewise probability distribution functions.

The lane deviation of the center of the detected vehicle is considered, which can be changed accordingly with the vehicle dimensions. It is taken around the center of the lane where weaving outside the lane is also taken into account for aggressive and drowsy drivers.

We have used trust levels of the predicted driver behavior to represent the probability distributions of the lane deviation. Firstly, lane deviation pdf's are developed for three different driver behaviors without including the trust levels as shown in Figure 3.4. Then a method is proposed to include the trust levels of the predicted diver behavior in the probability distributions of the lane deviations.

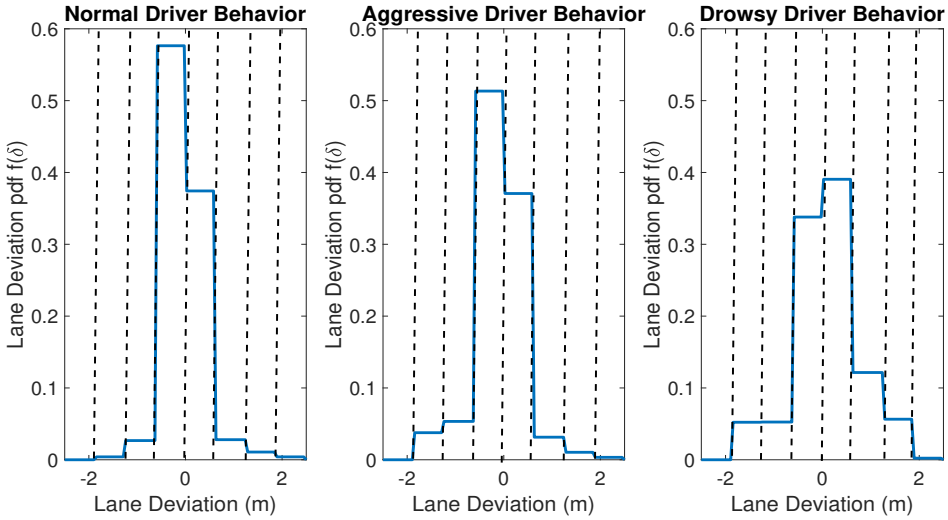


Figure 3.4: Lane Deviation PDF for different driver behaviors

The road lane width is taken as $4m$ and divided into eight cells where the lane's center is taken as zero lane deviation. The computation of the probability distributions for different

driver behaviors is done using the UAH-drive dataset on motorway roads as shown in Figure 3.4. From Figure 3.4 we can observe that drowsy and aggressive drivers are more likely to have larger lane deviations as compared to the normal drivers. Moreover, we can also observe that drowsy driver behaviors have more lane deviation probability than aggressive drivers.

The Online Trust Ratios (OTR) computed from equation 3.3, 3.4 and 3.5 give the trust values of each predicted driver behavior. These values are used to represent dynamically changing probability distributions of lane deviations based on trusted driver behavior ratios. Let OTR_N, OTR_A and OTR_d be the respective online trust ratios computed for a surrounding vehicle at some time point. Let ND, AD and DD be the 8×1 probability vectors representing the lane deviations of 100% normal, aggressive and drowsy driving respectively as given in Figure 3.4. A cumulative probability distribution $c(\delta)$ is formulated based on the OTR values as follows:

$$c(\delta) = OTR_N(ND) + OTR_A(AD) + OTR_D(DD) \tag{3.13}$$

The size of cumulative distribution $c(\delta)$ probability vector for lane deviation is also 8×1. It represents the pdf across eight cell divisions across the lane by combining all three driver behaviors and their OTR values. The Figure shows cumulative pdf $c(\delta)$ for different OTR values of the detected vehicle at some time instant k.

This representation of the lane deviation pdf using the OTR values allows to account for slight uncertainties in the predicted driver behavior.

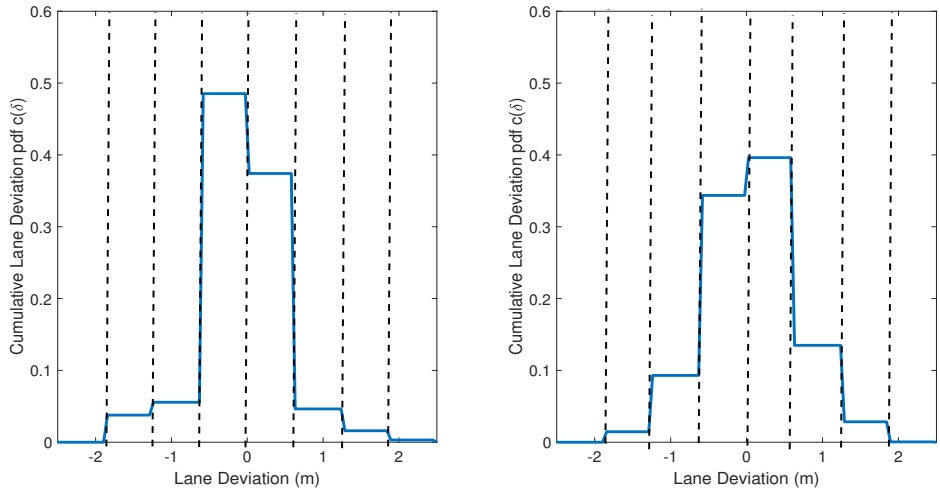


Figure 3.5: Lane Deviation PDF for different driver behaviors. a.) Left figure shows pdf for OTR 0.85 for aggressive, 0.05 drowsy and 0.1 normal driver behavior. b.) Right figure shows pdf for OTR of 0.85 drowsy, 0.12 aggressive and 0.03 normal driver behavior.

Longitudinal Acceleration

The second input of the surrounding driver is the acceleration command. Transition probability matrices for the acceleration input for different driver behaviors and road situations are computed to quantify the changes in a detected driver's acceleration command during a prediction time horizon N. The UAH-driver dataset is used for 4 different actions on the road, i.e., Lane Change (LC), Going straight (GS), Turning Left (TL), and Turning Right (TR). The acceleration command range of $[-4, 2.5] m/s^2$ is considered and divided into 65 ranges of $0.1 m/s^2$ each. The transition probability matrix for input acceleration command is computed for each of the road actions and driver behavior. Each matrix's size is 65×65 defining the probability of transition from one acceleration range to the other for the next time step. These are computed offline and can be loaded into the car during its operation.

The 12 input transition matrices for driver behavior are named using the following convention for each driver behavior and road action: Π_{B_c} where $B \in [N, A, D]$ for normal, aggressive, and

drowsy behaviors respectively, and $c \in [LC, GS, TL, TR]$ for a lane change, going straight, turning left and right respectively. Each entry of i_{th} row and j_{th} column of Π_{B_c} defines the probability of input changing from $u_n = i$ to $u_{n+1} = j$ in the next time step.

Some examples of the longitudinal acceleration input transition matrices for lane change and going straight are shown in Figure 3.6, 3.7 and 3.8.

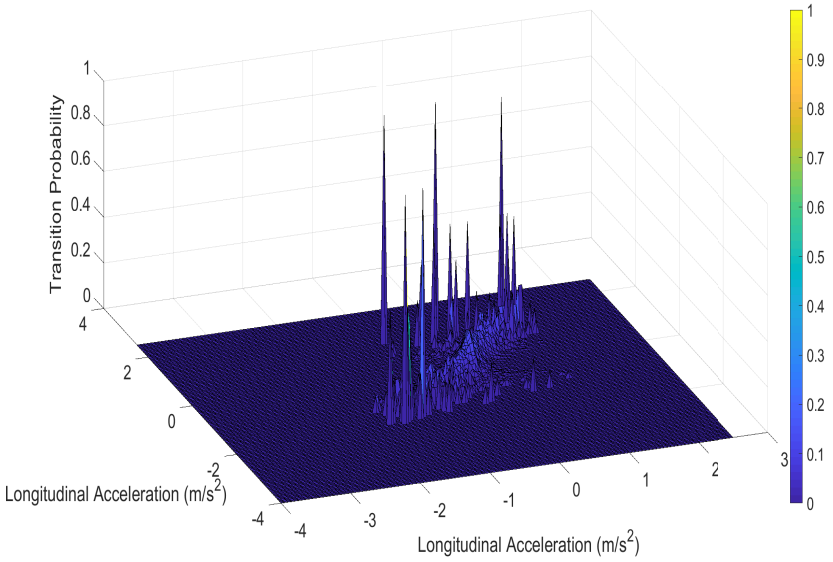


Figure 3.6: Longitudinal acceleration transition probabilities of a normal driver behavior straight driving

It can be seen from these figures that how different driver behavior types have specific driving characteristics. Figure 3.6 shows that normal driver behaviors exhibit only a small range of longitudinal accelerations. In drowsy and aggressive driver behaviors, we can observe from Figure 3.7 and 3.8 that these driver behaviors exhibit higher ranges of longitudinal accelerations.

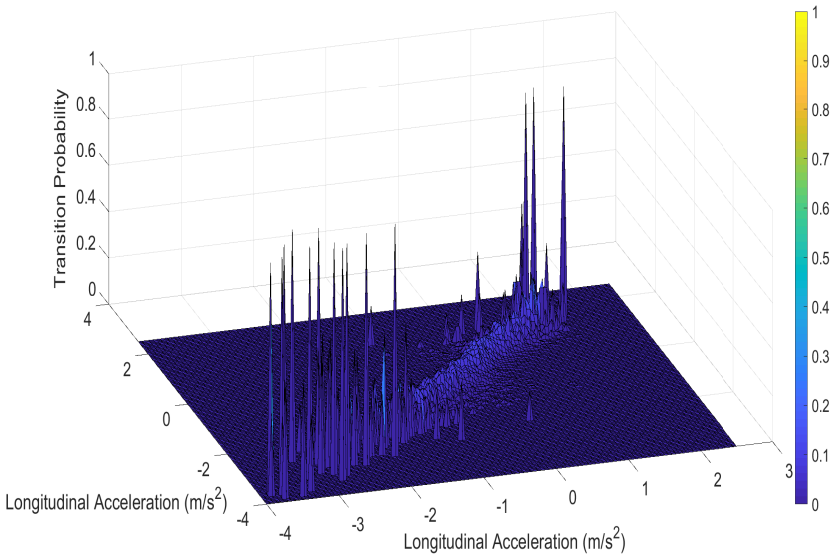


Figure 3.7: Longitudinal acceleration transition probabilities of an Aggressive driver behavior straight driving

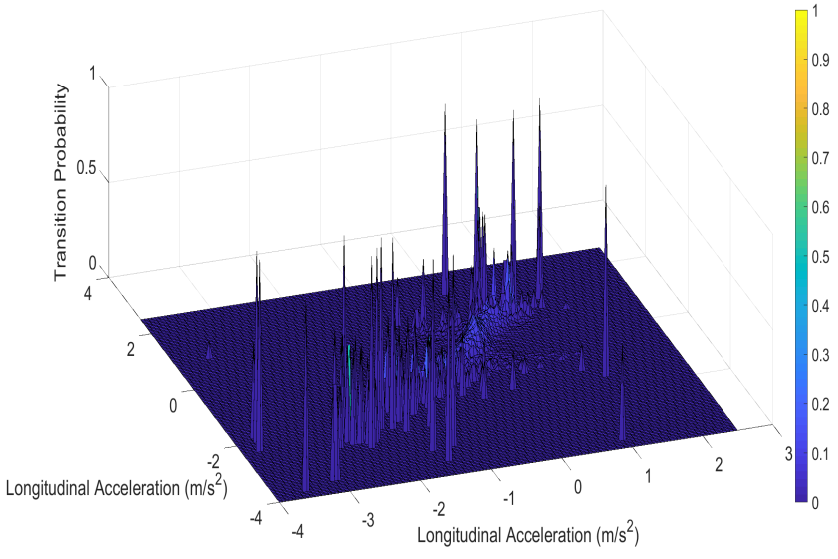


Figure 3.8: Longitudinal acceleration transition probabilities of drowsy driver behavior straight driving

3.4.3 Driver behavior-based stochastic reachable sets

After the state space discretization and offline computation of the transition probability matrices for states and inputs, the detected vehicle's SR sets can be computed online using the current sensor measurements and the dominant driver behavior corresponding to the maximum OTR value. These are called the driver behavior-based stochastic reachable sets.

As stated before, Eqn. 3.10 assumes a known sequence of driver inputs for N time steps. This is not true as the driver inputs are stochastic and unknown. To incorporate driver behavior inputs in the SR set computation, the acceleration input transition probability matrices are incorporated in equation 3.10. This is done using the offline computation of 65 state transition matrices $T_{ij}^{u_n}$ for each possible acceleration input range, with each of them having a size of $D \times D$. All these state transition probability matrices are stored in a sparse matrix as follows:

$$T_{ij} = \begin{bmatrix} T_{ij}^{u_1} & 0 & 0 \cdots & \cdots & 0 \\ 0 & T_{ij}^{u_2} & & & 0 \\ 0 & & T_{ij}^{u_3} & & \vdots \\ \vdots & & & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & T_{ij}^{u_{65}} \end{bmatrix} \tag{3.14}$$

where each $T_{ij}^{u_n}$ is a $D \times D$ state transition probability matrix for each of the 65 input cells. Size of the above matrix T_{ij} is $65D \times 65D$. The input transition probability matrix Π_{B_c} is also written in the time dependent sparse matrix form $\widetilde{\Pi}(k)$ as follows:

$$\widetilde{\Pi}(k) = \begin{bmatrix} \Pi_{B_c} & 0 & 0\cdots & \cdots & 0 \\ 0 & \Pi_{B_c} & & & 0 \\ 0 & & \Pi_{B_c} & & \vdots \\ \vdots & & & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & \Pi_{B_c} \end{bmatrix} \tag{3.15}$$

where each Π_{B_c} is a 65×65 input transition probability matrix for each of the 65 input cells corresponding to predicted driver behavior and intention c at time k. Size of $\widetilde{\Pi}(k)$ is also $65D \times 65D$. The eqn.3.11 representing probability distribution of the discrete states also has to be re-written in accordance with the input probability distribution. Let the input probability distribution at time k be represented by vector $p^u(k)$:

$$p^u(k) = [p_1^u(k)\ \ p_2^u(k)\ \ ....\ \ p_{65}^u(k)]^T \tag{3.16}$$

where each entry in $p^u(k)$ is the probability distribution of driver inputs divided into 65 segments. $p^u(k)$ also addresses the initial sensor measurement uncertainty for the inputs. The $p(k)$ from Eqn. 3.10, can be redefined in terms of $p^u(k)$ as follows:

$$\begin{aligned} \bar{p}(k) = [p_1(k)p_1^u(k),\ \ p_1(k)p_2^u(k)\ \ ....\ \ p_1(k)p_{65}^u(k) \\ p_2(k)p_1^u(k)\ \ p_2(k)p_2^u(k)\ \ ...p_2(k)p_{65}^u(k)\ \ p_3(k)p_1^u(k)\ \ .... \\ p_D(k)p_{65}^u(k)]^T \end{aligned} \tag{3.17}$$

where the values are taken from vector $p^u(k)$ and $p(k)$. Size of the new probability vector $\bar{p}(k)$ is $65D \times 1$ which makes its dimension compatible for the computation of the future

SR sets using the multiplication of the T_{ij} and $\widetilde{\Pi}(k)$. Equation 3.10 for computation of the future SR sets can be rewritten as:

$$\bar{p}(k+1) = \widetilde{\Pi}(k) T_{ij} \bar{p}(k) \tag{3.18}$$

where T_{ij} and $\widetilde{\Pi}(k)$ are taken from equation3.14 and 3.15 respectively. The above equation 3.18 is used for the computation of probability distributions of the longitudinal position of the detected vehicle. Once the N probability vectors $[\bar{p}(k)\ \bar{p}(k+1)\ ...\ \bar{p}(k+N)]$ have been computed, they can be converted into the $D \times 1$ probability vector $p(k)$ by adding all probability values for a certain discrete state as shown:

$$p_n(k) = \sum_{65(n-1)+1}^{65n} \bar{p}_i(k) \tag{3.19}$$

for the n^{th} entry of probability vector $p(k)$ where k can be any time instant. Then using eqn. 3.12, final probability distribution along the longitudinal position can be computed and are stored in $\widetilde{p}(k)$. The lane deviation probability distributions $c(\delta)$ computed from eqn.3.13 are used to describe the probability distributions in the lateral direction. These are stored in the 8×1 vector $L(k)$:

$$L(k) = [c_1(\delta)\ \ c_2(\delta)\ \ ...\ \ c_8(\delta)] \tag{3.20}$$

where $c_1(\delta), c_2(\delta)...c_8()$ are the entries of $c(\delta)$ from eqn.3.13. $L(k)$ will keep changing dynamically over each time step according to eqn.3.13. Using both longitudinal and lateral probability distributions, rectangular random sets are computed to represent driver behavior based stochastic reachable sets.

For fast computation, low values of probability that lie under a threshold p_{th} in longitudinal and lateral probability distributions are taken as zero. The random matrix is computed using the following matrix computation:

$$P_{SR}(k+i) = \begin{bmatrix} \widetilde{p}_1(k+i)L_1(k+i) & \cdots & \widetilde{p}_1(k+i)L_8(k+i) \\ \widetilde{p}_2(k+i)L_1(k+i) & \cdots & \widetilde{p}_2(k+i)L_8(k+i) \\ \vdots & \vdots & \vdots \\ \vdots & & \\ \widetilde{p}_b(k+i)L_1(k+i) & \cdots & \widetilde{p}_b(k+i)L_8(k+i) \end{bmatrix} \tag{3.21}$$

where $P_{SR}(k+i)$ is the i_{th} random matrix associated with the i_{th} SR set in the future and b is the total number of divisions in the longitudinal position. For a prediction horizon N, i will be in $[0, N]$. $SR(k+i)$ has b rows and 8 columns. The row and column entries represent the longitudinal and lateral probability distribution, respectively, for the cell divisions made along with the longitudinal and lateral positions. The probability values below threshold p^{th} will be taken as zero. The SR set at the time of detection k can simply be calculated with sensor measurements represented through $p(k)$ and cumulative lane deviation distribution $L(k)$.

The finally obtained $P_{SR}(k+i)$ can be represented by a rectangular covering function [60] representing the probabilities of each cell of the SR set. This can be done using the following equation:

$$SR(k+i) = f(x, y) \tag{3.22}$$

$$= Pr[(x, y) \in d] \tag{3.23}$$

where d is a certain discrete state or cell and $Pr[(x, y) \in d]$ is extracted from equation 3.21. An example of a rectangular covering function for the behavior-based SR set is illustrated in Figure 3.9.

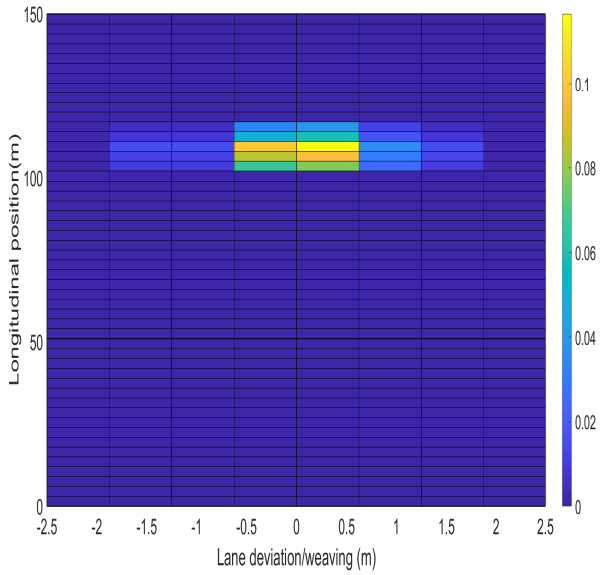


Figure 3.9: $SR(k)$ represented by a rectangular covering function, where each cell has a probability value

The only difference will be during the SR sets' computation during the lane change, where the center will be shifted according to driver behaviors, as shown in Figure.

The ego vehicle computes N future SR sets of the detected vehicle for a fixed prediction horizon N for N time steps. This gives a better risk estimation to the ego vehicle about the surrounding vehicle at each time step.

Method for computation of the SR sets is shown in Algorithm 1.

3.5 Threat Assessment using probability of collision

The SR sets computed act as an unsafe set of positions that have to be avoided by the ego vehicle. The ego vehicle's safety can be guaranteed for a prediction horizon N, if its pre-

Algorithm 1 Future SR sets computation

Input: $p^u(k)$; sensor measurements at time k
Output: [SR(k+1), SR(k+2), $\cdots SR(k+i)$]
Data: Π_{B_c} and T_{ij} matrices computed offline

1: **while** $i < N$ **do**
2: Predict dominant driver behavior B and intention c at time $k+i$
3: Compute $\widetilde{\Pi}(k+i)$ using Eq. 3.15
4: Compute $\bar{p}(k+i)$ using Eqn. 3.18
5: Apply Eq. 3.19 to $\bar{p}(k+i)$
6: Compute lateral pdf $L(k+i)$ according to the Eqn. 3.13
7: Compute $P_{SR}(k+i)$ using Eqn. 3.21 and replace entries lower than p_{th} with 0
8: Extract the lowest dimension matrix with positive values
9: Compute the probability distribution of the SR set using equation 3.23 and get the rectangular covering function.
10: i← i+1
11: **end while**

planned trajectory for that time horizon does not belong to the computed behavior-based SR sets. The computation of N behavior-based SR sets at any time k can be used to assess current threat and threat for a few seconds ahead to the ego vehicle. In this study, the probability of crash is used as a measure of threat. Two types of threats are formulated at time k, i.e., current threat and the short-term prediction threat to the ego vehicle.

3.5.1 Current Threat

Current Threat (CT) or instantaneous threat is defined as the probability of collision with the surrounding vehicle at the time of detection k. The SR set $SR(k)$ computed at time k is used to compute the probability of collision of ego vehicle with the detected vehicle. Let $[(x_1^e, y_1^e)$, (x_2^e, y_2^e), ..., $(x_N^e, y_N^e)]$ be a pre-planned path of the centre of the ego vehicle for a prediction horizon N. The ego vehicle dimensions are taken as L_{ego} and W_{ego}. The current threat to the ego vehicle at time k from the detected vehicle can be computed using the sum of probability cells that intersect with the ego vehicle's body at time k. This is illustrated

Figure 3.10: a.) Zero Current Threat for a detected oncoming normal driver at time k. b.) Positive Current Threat of 0.1757 for a detected drowsy driver at time k.

Algorithm 2 Current Threat computation

Input: p$^u(k)$; (x_1^e, y_1^e); sensor measurements at time k;
Output: Current Collision Probability

1: Compute SR(k) using the sensor measurements and L(k) using Eq. 3.20
2: Determine intersecting cells of $SR(k)$ with the vehicle body around (x_1^e, y_1^e)
3: Add the probabilities of the intersecting cells in $SR(k)$

in Figure 3.10 .

$$CT(k) = p_1(k) + p_2(k) + \cdots + p_i(k) \tag{3.24}$$

where $p_1(k), p_2(k), \cdots p_i(k)$ are the probabilities of the intersecting cells with the ego vehicle's body at time k.

3.5.2 Short term prediction threat

Short Term Prediction Threat (STPT) is defined as the sum average of positive weighted probabilities of collision with the surrounding vehicle at the time of detection k. These are calculated for the future behavior-based SR sets $[SR(k+1), SR(k+2), \cdots SR(k+N)]$, where N is the prediction horizon. In simple terms, STPT is the sum average of perceived future positive weighted CT values at each future time. The sum of weighted probabilities is taken because the prediction made more into the future would be less trusted than predictions for just after detection. A discount factor γ is used to weight each consecutive probability of collision to compute STPT as shown:

$$STPT(k) = \frac{CT(k) + \gamma CT(k+1) + \cdots (\gamma)^N CT(k+N)}{\text{Number of positive entries in each CT}} \tag{3.25}$$

A positive value of STPT can help in proactive decision-making for the ego vehicle.

There are other alternate measures possible for the evaluation of the probability of crash using the computed Stochastic Reachable sets. One measure could be taking the maximum of the computed Current Threat for a finite time horizon. This can give a better quantification of how dangerous a particular situations is at a given time.

Our STPT approach, differs in a way that the threat is detected at any time STPT goes slightly positive (i.e. no lower threshold used). It just includes an average measure for probability of crash. The magnitude of our proposed STPT, does not exactly reflect how dangerous a certain situation is. Instead, the location of the SR sets will tell how dangerous a situation is and does the autonomous vehicle needs to take an aggressive control action or not.

3.6 Summary

This chapter introduced a novel trust-based driver behavior prediction methodology using the Hidden Markov Models. These HMM models are trained for four features. A novel trust-based driver behavior prediction method is formulated, which combines trust values from all sensor measurements for a reliable motion prediction model. This chapter also introduced a stochastic reachable set-based threat assessment approach that uses the trust level of the predicted driver behavior and intention estimation to evaluate threats around the ego vehicle. Moreover, a continuous threat measure - Short-Term Prediction Threat is proposed, which continuously assesses the threat with weighted probabilities at each time step. As most of

the intensive computations are done offline, it makes the proposed methodology practical for real-time implementation.

Chapter 4

Crash Avoidance using Stochastic Model Predictive Control

4.1 Introduction

This chapter covers the details of the proposed SMPC algorithm for crash avoidance decision-making in autonomous vehicles. Stochastic Model Predictive Control (SMPC) has recently shown immense potential for systems operating in uncertain environments like autonomous vehicles [10, 51]. It allows exploiting the receding horizon control framework with uncertainty formulation using probabilistic or chance-constraints. In the case of autonomous vehicles, SMPC ensures that the probability of crash always stays below a minimum threshold predefined risk parameter. This is done by having an upper bound on the probability of crash with the other vehicle.

The contributions of the proposed SMPC algorithm are as follows:

1. Chance constraints are converted into deterministic constraints using the safe allowable state space's convex hull formulation. This results in a fast real-time implementation.

2. Relationship between controller conservativeness and minimum threshold risk parameter is discussed.

This chapter includes ego vehicle dynamic modeling, computation of chance constraints from SR sets of the surrounding vehicle, deterministic formulation of chance constraints, and control design for crash avoidance decision-making.

4.2 Ego Vehicle Dynamic Modelling

4.2.1 Dynamic Bicycle Model

For the controller design, a dynamic bicycle model adapted from [45] is used to model the dynamics of the ego vehicle. A 2 degree of freedom bicycle model is considered as shown in 4.2. The lateral, longitudinal and yaw dynamics for the ego vehicle are given by the following equations:

$$m(\dot{v}_x - v_y\dot{\theta}) = F_{xT} \tag{4.1}$$

$$m(\dot{v}_y + v_x\dot{\theta}) = F_{yf} + F_{yr} \tag{4.2}$$

$$I_z\ddot{\theta} = F_{yf}l_f - F_{yr}l_r \tag{4.3}$$

Ego Vehicle's motion in global coordinates can be described using the following equations of motion:

$$\dot{X} = v_x cos\theta - v_y sin\theta \tag{4.4}$$

$$\dot{Y} = v_x sin\theta + v_y cos\theta \tag{4.5}$$

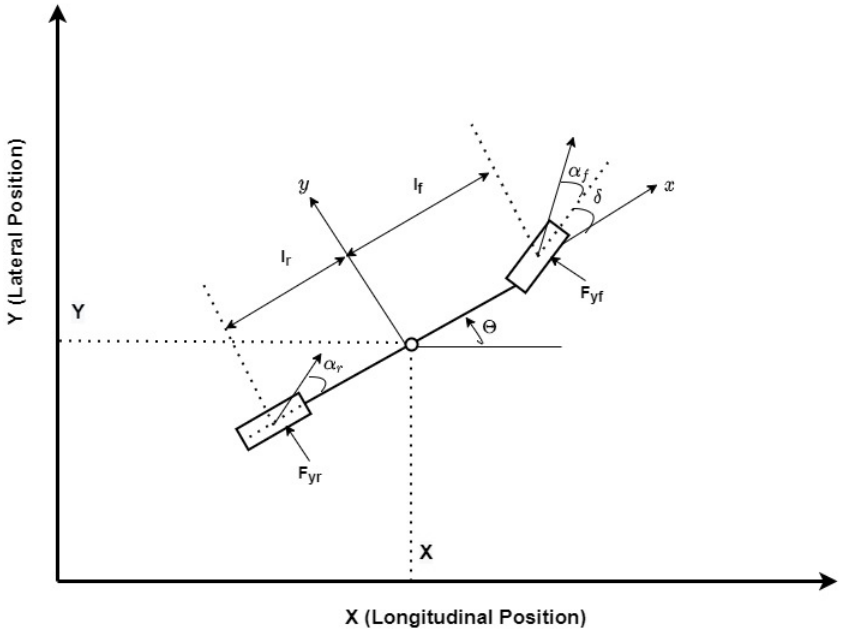


Figure 4.1: Ego Vehicle Bicycle Model

where X, Y are the ego vehicle's longitudinal and lateral positions in the global coordinates; v_x, v_y are the vehicle's longitudinal and lateral velocity; θ is the heading angle of the vehicle; $\dot{\theta}$ is the yaw rate at the center of gravity of the vehicle; m is the mass of the vehicle; I_z is the moment of inertia of the vehicle about the vertical axis. F_{xT} is the longitudinal force on the tires; F_{yf} and F_{yr} are the lateral force on the front and rear tires, respectively.

A linear tire model is used as developed in [61] to model the tire friction:

$$F_{yf} = -C_{\alpha f}\alpha_f = C_{\alpha f}\left(\delta - \frac{v + l_f\dot{\theta}}{v_x}\right) \tag{4.6}$$

$$F_{yr} = -C_{\alpha r}\alpha_r = C_{\alpha r}\left(-\frac{v - l_r\dot{\theta}}{v_x}\right) \tag{4.7}$$

Where f and α_r are the side slip angles for the front and rear tires; δ is the steering angle; C_{α_f} and C_{α_r} are the cornering stiffness for the front and the rear tires respectively.

4.2.2 Linear Parameter Varying (LPV) Model

For the SMPC design, the above dynamic bicycle model is linearized around the current longitudinal velocity v_x to obtain an LPV model of the ego vehicle. In state space form, the dynamics can be written as:

$$\dot{x}^{\text{ego}} = Ax^{\text{ego}} + Bu \tag{4.8}$$

$$y^{\text{ego}} = Cx^{\text{ego}} + Du \tag{4.9}$$

where, $x^{\text{ego}} = \begin{bmatrix} X & v_x & Y & v_y & \theta & \dot{\theta} \end{bmatrix}^T$, $u = \begin{bmatrix} F_{xT} & \delta \end{bmatrix}^T$,

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{C_{\alpha f}+C_{\alpha r}}{mv_x} & 0 & \frac{l_r C_{\alpha r}-l_f C_{\alpha f}}{mv_x} - v_x \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & \frac{l_r C_{\alpha r}-l_f C_{\alpha f}}{I_z v_x} & 0 & -\frac{{l_f}^2 C_{\alpha f}+{l_r}^2 C_{\alpha f}}{I_z v_x} \end{bmatrix}; \; B = \begin{bmatrix} 0 & \frac{1}{m} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{C_{\alpha f}}{m} & 0 & \frac{l_f C_{\alpha f}}{I_z} \end{bmatrix}$$

This model is valid under two assumptions. Firstly, a small cornering angle is assumed. Lateral velocity v_y is also assumed to be small and the term $y\dot{\theta}$ is neglected in equation 4.1. These assumptions are valid as in the high-speed motorway scenarios, lateral and cornering angles will be small.

This continuous state-space model is discretized using the forward-Euler method to be used for a discrete controller design. This discrete dynamic model can be represented by:

$$x_{k+1}^{\text{ego}} = \bar{A}x_k^{\text{ego}} + \bar{B}u_k \tag{4.10}$$

$$x_{k+1}^{\text{ego}} = f(x_k^{\text{ego}}, u_k) \tag{4.11}$$

where $\bar{A}$ and $\bar{B}$ are the discretized state space matrices for some velocity v_x at time k.

4.3 Evaluation of chance-based safety constraints

The ego vehicle uses the SR sets of the surrounding vehicle to infer its possible future movements. A positive measure of STPT will imply that there is an immediate risk to the ego vehicle, and action must be taken at that time to avoid a crash. This is called proactive decision-making in autonomous vehicles. It exploits the intent of the surrounding human-driven vehicles to assess future threat instead of reactive decision-making [11] which does not account for the future behavior of the surrounding vehicles.

For the SMPC formulation, the probabilistic chance constraints have to be defined. In our case, the chance-constraints are defined using the STPT information at each time instant k as follows:

$$STPT(k) \leq p_{th} \tag{4.12}$$

where p_{th} is the tunable predefined risk parameter. The safety constraint above for crash avoidance has to be satisfied with a minimum probability of p_{th}. This probabilistic constraint ensures crash avoidance for a prediction horizon N with atleast $(1 - p_{th})$ probability. From equation 3.25, it could be further written as:

$$\frac{Pr[x_k^{\text{ego}} \in SR(k)] + Pr[x_{k+1}^{\text{ego}} \in SR(k+1)] + \cdots + Pr[x_{k+N}^{\text{ego}} \in SR(k+N)]}{\text{Number of Positive Probabilities}} \leq p_{th} \tag{4.13}$$

where x_k^{ego} is the ego vehicle state at time k. The chance constraint in equation 4.13, can

be simplified using a SR set union representation for a finite prediction horizon N. Let the combined set of N SR sets at time k be represented by K_k^{unsafe}, where K_k^{unsafe} is the union of N predicted SR sets.

$$K_k^{\text{unsafe}} = SR(k) \cup SR(k+1) \cup \cdots \cup SR(k+N) \tag{4.14}$$

Each $SR(k)$ is represented by respective rectangular covering functions, containing each cell's location and probability value. The union of all the SR sets can be viewed as the locations or cells that have to be avoided by the ego vehicle. K_k^{unsafe} is the time-varying probabilistic unsafe set which has to be avoided by the ego vehicle at each time instant k. This unsafe set K_k^{unsafe} will get updated at each instant using the algorithm 1. Using the information from equation 4.14.The updated chance constraints for prediction horizon N can be approximately written as follows:

$$Pr(x_{k,k+N}^{\text{ego}} \in K_k^{\text{unsafe}}) \leq p_{th} \tag{4.15}$$

where $x_{k,k+N}^{\text{ego}}$ is the state of the ego vehicle at time from time k to $k+N$, KK_k^{unsafe}) is the union of the N unsafe SR sets at time k; p_{th} is the tunable predefined risk parameter.These safety constraints for crash avoidance have to be satisfied with a minimum probability of p_{th}.

4.4 Deterministic Formulation for chance constraints

For fast real-time implementation, it is necessary to get a tractable and deterministic representation of the chance constraints from equation 4.15. This can be computationally expensive in our case as the probability distributions are non-Gaussian [62].

A deterministic and hard constraint reformulation method is presented for the chance constraints in equation 4.15. The area outside the K_k^{unsafe} can be considered the safe area for the ego vehicle's path planning. Moreover, to avoid collisions with the vehicles in other lanes, the area outside K_k^{unsafe} and inside the ego vehicle's lane can be considered as the safe area.

Let this safe area be represented by:

$$K_k^{\text{safe}} = (K_k^{\text{unsafe}})' \tag{4.16}$$

where $(K_k^{\text{unsafe}})'$ represents the area outside (K_k^{unsafe}) and inside the the current lane of the ego vehicle. This allows to re-formulate the chance constraints as hard constraints as follows:

$$x_{k,k+N}^{\text{ego}} \in K_k^{\text{safe}} \tag{4.17}$$

This hard constraint ensures the safety of the vehicle with at least p_{th} probability and can be handled in a general optimization framework of Model Predictive Control (MPC) formulation.

4.4.1 Convexification of the deterministic constraint

Just specifying a hard deterministic formulation for the chance-constraints is not sufficient for a fast SMPC real-time implementation. The hard constraint formulation can be converted to convex constraints. This will lead to fast real-time implementation, as the optimization framework for the MPC gets converted to a quadratic programming framework. Convexification of deterministic constraints is done by representing the safe set K_k^{safe} in terms of a convex hull. The main idea is that a convex hull can be represented using linear inequality

constraints of the form [63]:

$$Ax \leq b \tag{4.18}$$

Where A is a matrix, $x \in \mathbb{R}^2$ is the vector of points in the 2-D plane.

Generally, K_k^{safe} is a non-convex area, and it is not easy to represent it using one convex hull. To overcome this, the area K_k^{safe} can be considered as consisting of many successive convex hull regions. The closest safe convex hull enclosing the ego vehicle is computed to avoid crash with the other vehicle. The MPC can use this closest safe convex hull information at time k to guide and steer the ego-vehicle towards that safe region. Algorithm 3 provides the details of how the closest safe convex hull is computed inside the safe set K_k^{safe}:

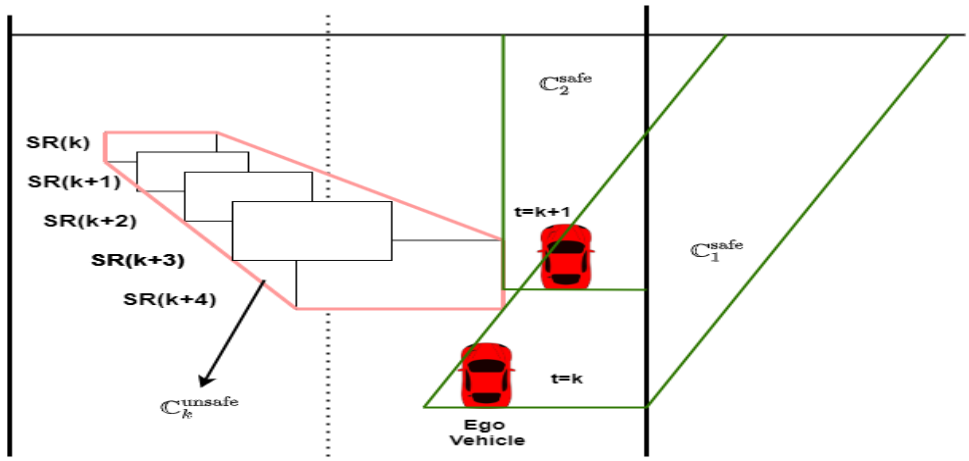


Figure 4.2: Algorithm 3 implementation for the oncoming vehicle

The detail of this procedure is shown in the Figure for two cases: an oncoming vehicle and a cut-in scenario. The safe area K_k^{safe} can be considered as consisting of many convex hulls. The closest safe convex hull inside the K_k^{safe} at time k is represented as $\mathbb{C}_k^{\text{safe}}$ as shown in the Figure . From algorithm 3, linear inequality constraints are obtained for the safe convex hull. A_k^{safe} and b_k^{safe} computed from algorithm 3 are used in the MPC controller design as

Algorithm 3 Computation of linear inequality constraints for the closest safe convex hull

Input: [SR(k), SR(k+1), $\cdots$, $SR(k+N)$], $[x(k), y(k)]$
Output: $\mathrm{A}_k^{\mathrm{safe}}, b_k^{\mathrm{safe}}$

Compute the convex hull of K_k^{unsafe}. Let this convex hull be denoted by $\mathbb{C}_k^{\mathrm{unsafe}}$. Store the vertices of this convex hull in variable **vertices**.
if the vehicle detected on the left lane of the ego vehicle **then**
    vertices = Points having lateral coordinate magnitude bigger than the rear left corner coordinate of the ego vehicle safety region.
    vertexClose = min(vertices(1) - x(k))
    if x(k)< x coordinate of **vertexClose then**
        mClose = slope of the line from the rear left corner of the safety region and the **vertexClose**
        Find the lateral coordinate on this line in the sensor range ($150m$) and store in **coord3**
        Using mClose and rear right corner of the safety region, find lateral coordinate in the sensor range (150)m store in **coord4**
    else
        coord3 = coordinate with same lateral coordinate as rear left corner of the safety region; longitudinal coordinate as the sensor range ($150m$)
        coord4 = coordinate with same lateral coordinate as rear right corner of the safety region; longitudinal coordinate as the sensor range ($150m$)
    end if
else
    Same computations using right rear corner of the safety region to compute **coord3** and **coord4** in the sensor range $150m$.
end if
Store the coordinates **coord3**, **coord4**, rear left and right corners of safety region in the variable **ConvHull**
Convert the vertex representation of **ConvHull** into the linear inequality representation using equation 4.18

follows:

$$A_k^{\text{safe}} x_k^{\text{ego}} \leq b_k^{\text{safe}} \tag{4.19}$$

Where A_k^{safe} and b_k are associated with the closest safe convex hull for the ego vehicle.

4.5 Control Design for Crash Avoidance decision-making

The linear inequality constraints computed using the convexification of the chance-constraints, can be used to formulate a tractable and deterministic version of the SMPC problem. The MPC optimization problem is formulated as:

$$min \;\; J = \min_{U} \sum_{k=1}^{N-1} \Big(\left\| x_k^{\text{ego}} - x_{k,ref}^{\text{ego}} \right\|_{\mathbf{Q}}^2 + \| u_k^{\text{ego}} \|_{\mathbf{R}}^2 \Big) + \Big(\| x_N^{\text{ego}} \|_{\mathbf{S}}^2 \Big) \tag{4.20a}$$

$$s.t. \qquad x_{k+1}^{\text{ego}} = f(x_k^{\text{ego}}, u_k), \;\; k \in \mathbb{N} \tag{4.20b}$$

$$SR(k+i) = g(SR(k+i-1)), \;\; i \in [1, N], k \in [1, N] \tag{4.20c}$$

$$u_k \in U_k, \qquad k = 1, 2, \cdots, N-1 \tag{4.20d}$$

$$x_k^{\text{ego}} \in X_k, \qquad k = 1, 2, \cdots, N \tag{4.20e}$$

$$A_k^{\text{safe}} x_k^{\text{ego}} \leq b_k^{\text{safe}} \tag{4.20f}$$

where, N is the prediction horizon; $U = [u_0, u_1, \cdots u_{N-1}]^T$; and $\|w\|_{\mathbf{W}}^2 = w^T \mathbf{W} w$; and $x_{k,ref}^{\text{ego}}$ is the reference state vector for the ego vehicle; $\mathbf{Q}, \mathbf{S} \in \mathbb{R}^{6\times 6}$ are the state-weighting matrices and $\mathbf{R} \in \mathbb{R}^{2\times 2}$ is the input weighting matrix; SR prediction model denoted by equation 4.20c; the state and input constraints are represented by U_k and X_k given in equation 4.20d

and 4.20e. The tractable and deterministic formulation for chance-constraints is given in equation 4.20f which leads to a fast-SMPC implementation.

4.6 Summary

This chapter presented the proposed proactive decision-making approach using Stochastic Model Predictive Control (SMPC). The information from SR sets of the surrounding vehicle allowed to formulate a chance-constrained problem for the crash avoidance. The chance-constrained framework was reformulated in terms of deterministic convex hull constraints. A safe area is computed at each time instant using the probabilistic reachable sets of the surrounding vehicle and the predefined risk parameter. This safe area is further formulated using convex hull representation, which allows it to be represented using linear inequality constraints. This converts the SMPC formulation into a simple MPC problem. The underlying constraints are still probabilistic depending on the risk factor chosen for the SMPC controller. It is expected that the proposed framework will have a fast real-time implementation due to deterministic MPC reformulation for the SMPC. The next chapter covers the evaluation and validation for the proposed threat assessment and motion prediction methodologies.

Chapter 5

Simulation Results and Discussion

5.1 Overall Flowchart for Threat Assessment and Decision-Making Algorithm

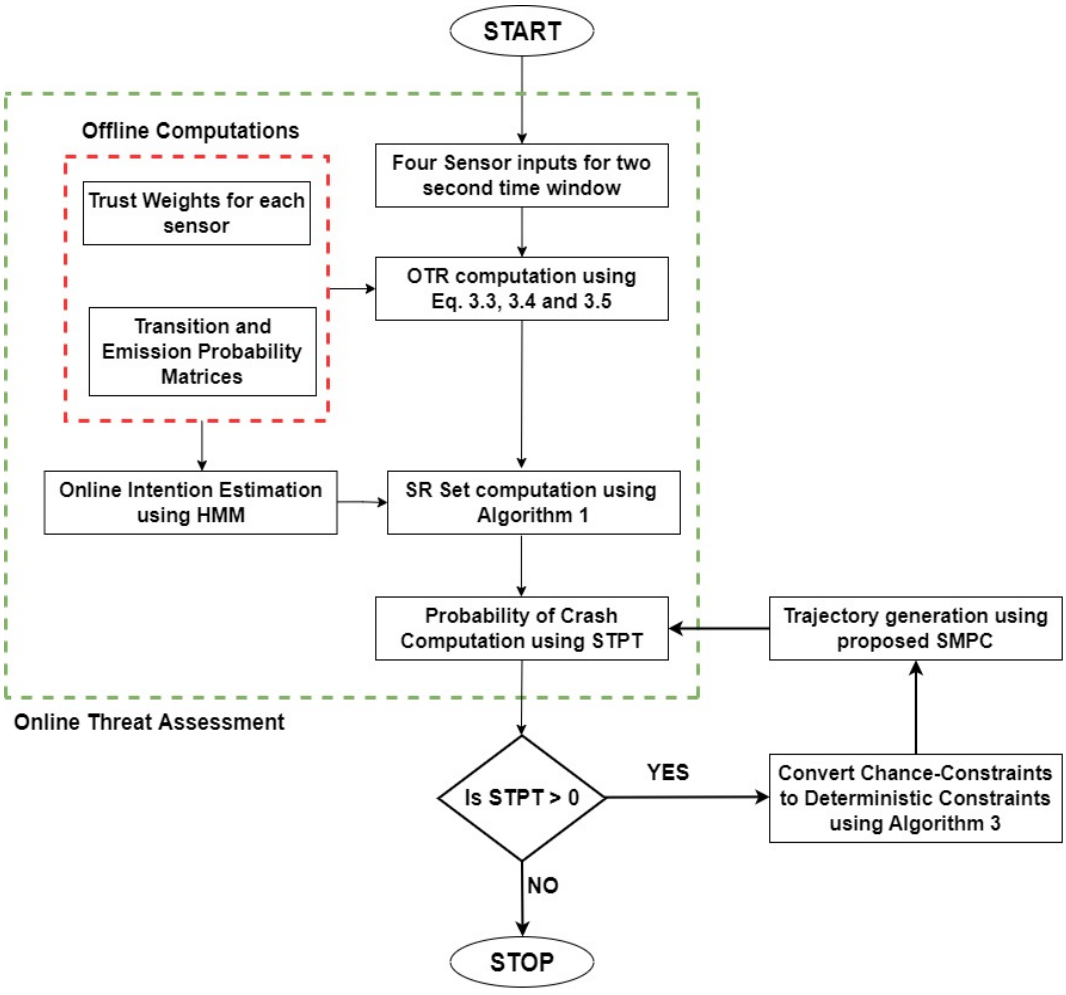


Figure 5.1: Flowchart for the working of the proposed threat assessment and decision-making algorithm

Figure 5.1, provides the overall flowchart for the proposed threat assessment and decision-making algorithm for crash avoidance in autonomous vehicles. The offline computations are done for Trust weights for each sensor, transition and emission probability matrices for HMM. The online implementation continuously computes the STPT value. The decision-making algorithm is implemented whenever STPT value is positive and keeps generating the trajectory until the STPT value reduces to zero. The MATLAB code for the proposed threat assessment and decision-making is available at our Github. [1].

The online threat assessment algorithm is validated and tested using the Driver 6 data from the UAH-Drive dataset. The overall decision-making algorithm using SMPC is also tested on simulations using time-series data of the driver 6 of the UAH-Drive dataset for different driver behaviors. The detailed explanations for the simulation setups are provided in the following sections.

5.2 Validation for Threat Assessment approach

A simulation study is conducted to evaluate and validate the proposed threat assessment method for autonomous vehicles. The time-series data from driver 6 of the UAH-drive dataset is used to conduct the simulation study. The videos provided in the dataset are used to extract the time stamps for the vehicle's different intentions, like a lane change, turning, and going straight. Current Threat and Short Term prediction threat are computed for a pre-planned trajectory of the ego vehicle in three different road scenarios: Oncoming vehicle, cut-in scenario, and a left-turning vehicle at the intersection. For threat assessment computation, a safety margin (Δ) around the vehicle is considered, whose dimensions are shown in Table 5.2. Continuous Reachability Analysis Toolbox(CORA)[64] is used for com-

[1]https://github.com/vanshajkhattar/ThreatAssessmentDecisionMaking

Table 5.1: Parameters for the offline computation

Parameters	**Value**
Velocity Segments(a)	13
Velocity Range	$[16, 42]m/s$
Longitudinal Position segments(b)	50
Longitudinal Position range	[0,150]m
Total Discrete states(D)	650
Surrounding vehicle v^{sw}	$7.2m/s$
One time step(τ)	0.1 sec

Table 5.2: Parameters for the simulation of the ego vehicle

Parameters	**Value**
Length(L)	$5m$
Width(W)	$2m$
Safety Margin (Δ)	$0.3m$
Prediction Horizon (N)	1 sec
γ for STPT	0.7

puting the intersection of the vehicle's safety regions with the computed behavior-based SR sets. The intersecting cells of the vehicle's safety region and the SR sets are used for threat assessment computation using Algorithm 2 and 1. Following parameters are used for the ego vehicle and offline computation of the transition probability matrices of the surrounding vehicle as shown in Table 5.1 and Table 5.2.

Evaluation of the proposed threat assessment method is done using 3 metrics: 1.) Driver behavior prediction accuracy; 2.) Intention estimation accuracy; 3.) Percentage of false positives and false negatives

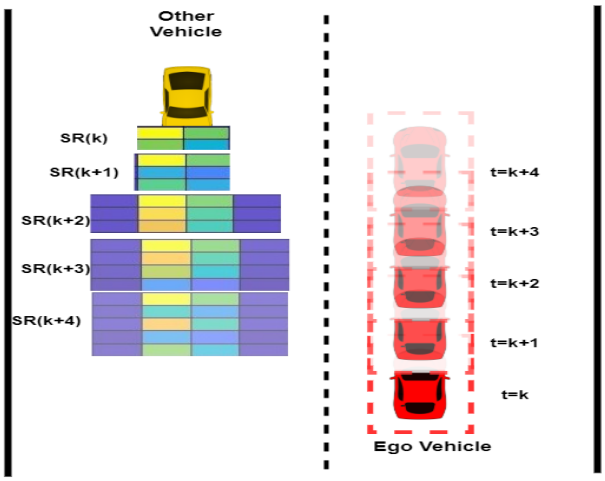


Figure 5.2: An oncoming vehicle with normal driver behavior going straight. The prediction time horizon is shown as 4 time steps in this simulation

5.2.1 Oncoming Vehicle

An oncoming vehicle from the adjacent lane is considered. This vehicle ideally should go straight. Two scenarios are considered where this vehicle will either go straight or illegally make a maneuver onto the ego vehicle's lane. Data points from driver 6 of the UAH-Drive dataset are used for going straight and lane change intentions. 20 simulations are conducted for different driver behaviors and intentions emulated by driver 6. Evaluation of the proposed threat assessment is done for the ego vehicle based on the 3 metrics given above.

At each time instant k, the ego vehicle has the following information: 1.)OTR values for driver behavior; 2.) Intention estimation I; 3.) Dynamic probability distribution for lane change $L(k)$ 4.) $CT(k)$ and $STPT(k)$. Figure 5.2 and figure 5.3 shows two cases of the oncoming vehicle going straight and a second case of making an illegal maneuver onto the ego vehicle's lane.

20 time-series data are taken for each vehicle going straight and vehicle changing lane in front of the ego vehicle. A positive value of STPT implies that there is a positive probability

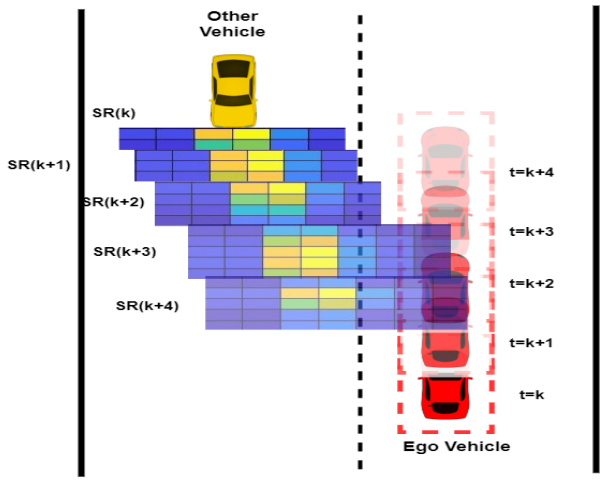


Figure 5.3: An oncoming vehicle with drowsy driver behavior coming into the lane of the ego vehicle. The prediction time horizon is shown as 4 time steps in this simulation

of crash. If the ego vehicle's safety region for the pre-planned path intersects with the time-series data of the surrounding vehicle at any time, it is assumed the crash will happen. In this case, the ideal situation is that there should be a positive value of STPT. Conversely, suppose there is no intersection between the ego vehicle's safety region for the pre-planned path and the time-series data for the surrounding vehicle. In that case, the STPT value should always be zero. False positives are when the STPT value is positive even though no crash is about to happen. False negatives are when the STPT value is zero even though the crash is about to happen. Both false positives and false negatives should be low for a reliable threat assessment methodology.

For the oncoming vehicle simulations, the average STPT value in an oncoming vehicle scenario for no crash or vehicle going straight was 0.07. The average STPT value in an oncoming vehicle for near-crash or crash situations is 0.78. These results imply that the proposed threat assessment methodology is successful in detecting safe and hazardous situations.

5.2.2 Cut-in Scenario

A cut-in scenario is considered where a vehicle from an adjacent lane going in the same direction changes the lane and cuts-in lane of ego vehicle going straight. Two scenarios are considered where the surrounding vehicle goes straight or cuts-into the lane of the ego vehicle.

For the cut-in vehicle simulations, the average STPT value in a cut-in scenario for no crash or vehicle going straight was 0.05. The average STPT value in a vehicle for near-crash or crash situations is 0.81. These results imply that the proposed threat assessment methodology is successful in detecting safe and hazardous situations.

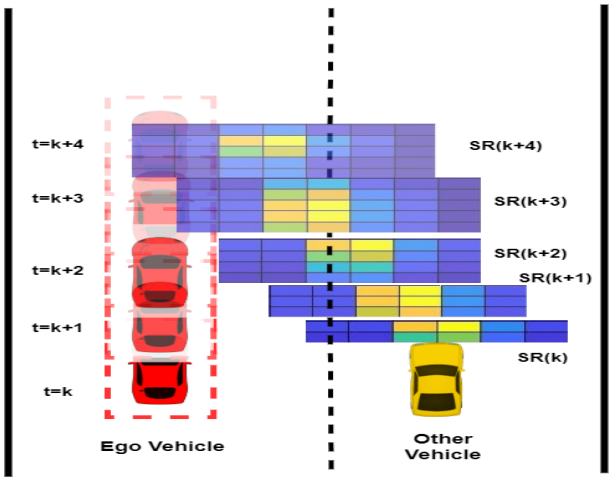


Figure 5.4: A surrounding vehicle with a drowsy driver behavior cuts in to the lane of the ego vehicle. The prediction time horizon is taken as 4 time steps

5.2.3 Intersection Scenario

An intersection scenario is considered where a surrounding vehicle can either go left, right, or straight.

For the intersection scenario simulations, the average STPT value in a cut-in scenario for

no crash or vehicle going straight was 0.10. The average STPT value in a vehicle for near-crash or crash situations is 0.65. These results imply that the proposed threat assessment methodology is successful in detecting safe and hazardous situations.

5.2.4 Evaluation

Table 5.3 shows how our proposed algorithm performs in three different road scenarios: oncoming vehicle, cut-in scenario, and intersection scenario where a vehicle can go straight left or right.

Driver behavior prediction accuracy is determined using the ground truth from driver 6 of the UAH-Drive dataset and our algorithm's predicted dominant driver behavior. The driver behavior gives dominant driver behavior with maximum Online Trust Ratio. Intention estimation accuracy is also computed using the ground truths from driver 6. False positives mean the instances at which there was no crash situation, but a positive value of STPT was computed at different time steps. Ideally, the STPT value should be closer to zero in case of no crash scenario. False negatives are the instances where a positive threat value was not reported in a crash/near-crash scenario. Ideally, STPT values should be high in case of a crash/near-crash scenario. The results are summarised in Table 5.3.

5.3 Simulation results for SMPC framework

The SMPC framework presented in chapter 4 will be tested for various road scenarios on the simulations. This method has been implemented in MATLAB using the Model Predictive Control toolbox. In this section, the simulation setup is discussed, and then three road scenarios are simulated for different initial conditions. This section aims to study the effec-

Table 5.3: Evaluation metrics for proposed threat assessment method on Driver-6 from UAH-Drive dataset

	Oncoming Vehicle	**Cut-in Scenario**	**Intersection Scenario**
Behavior Prediction Accuracy(%)	96%	95%	83%
Intention Estimation Accuracy(%)	89%	82%	76%
False Positives(%)	7%	9%	15%
False Negatives(%)	3%	5%	11%
Average Computation time(ms)	32.5	41.3	35.6

tiveness of the proposed decision-making algorithm. Moreover, the effects of three different initial conditions will be studied: variation in the risk factor p_{th}; variations in the relative distance at the time of detected threat ΔD; variations in the relative velocity at the time of detected threat ΔV.

5.3.1 Simulation Setup

The ego vehicle's dynamic model from equation 4.10 is used. The dimensions and parameters for the ego vehicle are taken from the table 5.2.

The ego vehicle will have the following input constraints U_k during its operation:

$$\delta \in \left[\frac{-\pi}{6}, \frac{\pi}{6}\right] \tag{5.1}$$

$$\Delta\delta \in \left[\frac{-\pi}{45}, \frac{\pi}{45}\right] \tag{5.2}$$

$$F_{xT} \in \left[0, \frac{T_{max}}{R_{eff}}\right] \tag{5.3}$$

and the following state constraints X_k:

$$Y \in [-2, 2]m \tag{5.4}$$

$$v_x \in [0, 42]ms^{-1} \tag{5.5}$$

$$\theta \in \left[\frac{-\pi}{3}, \frac{\pi}{3}\right] \tag{5.6}$$

The sampling time T_s is taken to be two seconds for the discretization. The prediction horizon for the SMPC is taken to be $N = 1$ second, which is equal to five time-steps; State weighting matrix $\mathbf{Q}$ is taken as diag(0.1,0,10,0,0,0); and the input weighting matrix $\mathbf{R}$ is taken as diag(0.1,0.1). Algorithm 3 is used to find the successive convex hull computations. The control horizon is taken as 0.2 seconds, i.e., one time step.

5.3.2 Variations in the risk factor p_{th}

Performance and safety of the proposed SMPC framework is studied for different risk factors p_{th}. Three ranges of risk factors are considered: low ($p_{th} = 0.05$); medium ($p_{th} = 0.10$); high $p_{th} = 0.15$. These variations will be considered for three road scenarios: Oncoming vehicle, cut-in scenario and the left turning vehicle at an intersection. The time-series data for the

surrounding vehicle is taken from the driver 6 data of the driver behavior dataset.

Oncoming Vehicle

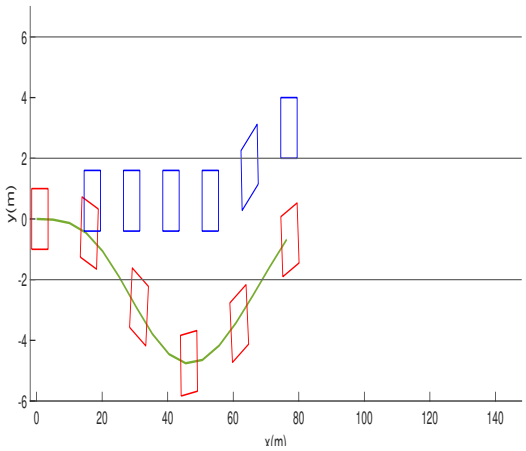
A scenario is considered where an oncoming vehicle from an opposite lane turns into the ego vehicle lane. Driver 6 aggressive waypoints for a lane change are considered. The initial relative distance from the oncoming surrounding vehicle when the threat was detected is taken to be $80m$. The initial velocity of the ego vehicle and the surrounding vehicle is taken to be $25m/s$. The following figures show the simulation results for this scenario.

Figure 5.5 shows three cases of an oncoming vehicle (blue) coming onto the ego vehicle lane. We can see that different values of risk factors affect the performance of the controller. Figure 5.25a shows that low value of p_{th} led to conservative control actions and the ego vehicle went far into the adjacent lane to avoid a crash. This conservative behavior can be dangerous on the road, as the ego vehicle can crash with the other vehicle on the adjacent lane. Figure 5.25b shows an effective response to an oncoming vehicle where the evasive maneuver is not too conservative while not going too much inside the adjacent lane. Figure 5.5 shows that having a high value of p_{th} can lead to a crash. This is due to the small SR sets because of the high p_{th}, which misses much information of the likely achievable states for the surrounding vehicle.

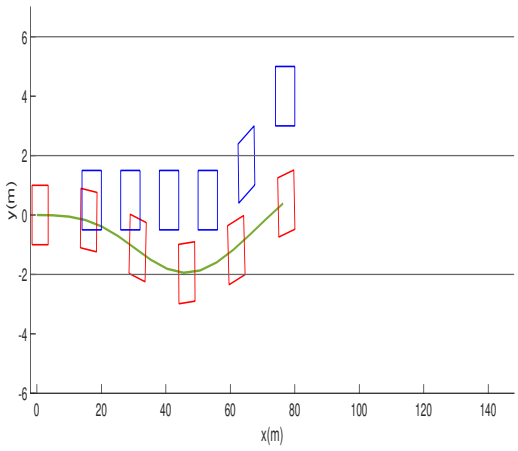
From Figure 5.6 and Figure 5.7, it is evident that a lower value of risk factor p_{th} leads to more aggressive control actions. The maximum and minimum steering angle values for $p_{th} = 0.05$ are atleast two times more than medium and higher p_{th}.

Cut-In Scenario

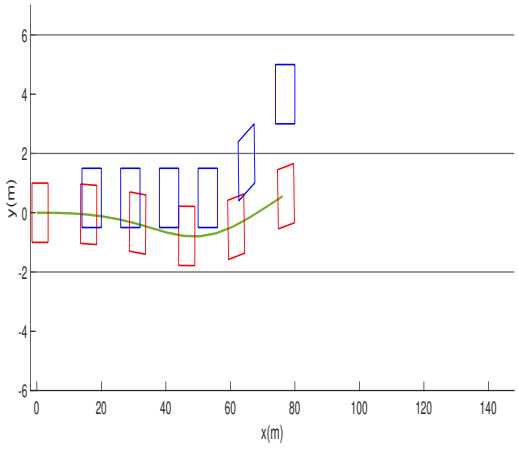
A cut-in scenario is considered where a vehicle from the adjacent lane drives into the ego vehicle lane. Driver 6 aggressive waypoints for a lane change are considered. The initial



(a) Risk Factor $p_{th} = 0.05$ (low)

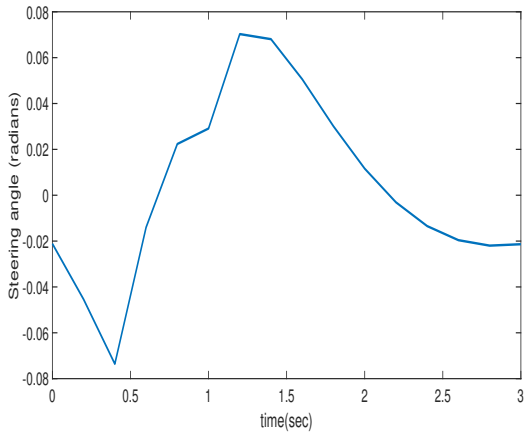


(b) Risk Factor $p_{th} = 0.10$ (medium)

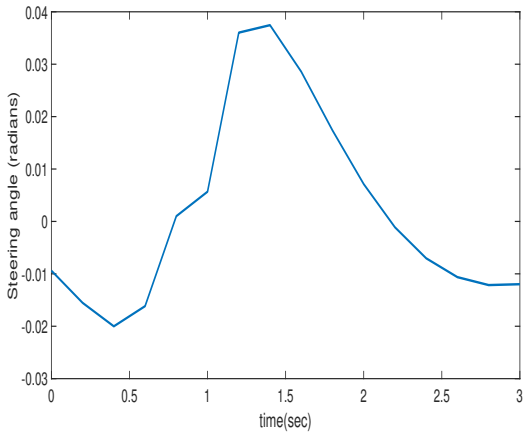


(c) Risk Factor $p_{th} = 0.15$ (high)

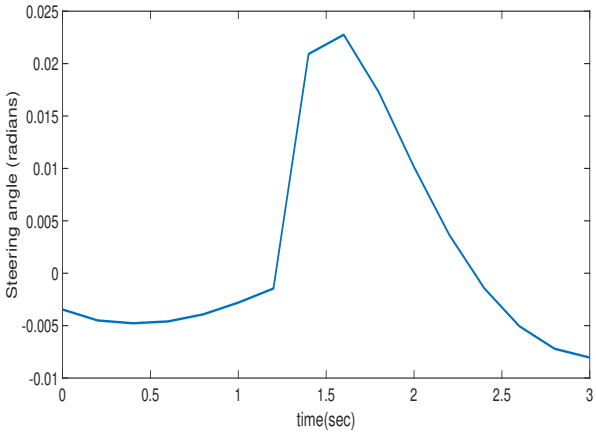
Figure 5.5: Visualization of the evasive maneuver by the ego vehicle (red) when the surrounding vehicle (blue) comes onto the lane of the ego vehicle. All vehicles are plotted at an interval of 0.6 seconds for a total time interval of 3 seconds. The ego vehicle starts from left at $(0, 0)$ and the surrounding vehicle starts from the right at relative distance of $80m$



(a) Steering angle input for the ego vehicle to generate an evasive maneuver for low $p_{th} = 0.05$

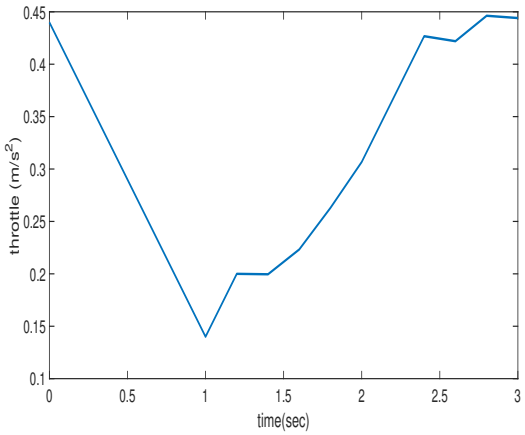


(b) Steering angle input for the ego vehicle to generate an evasive maneuver for medium $p_{th} = 0.10$

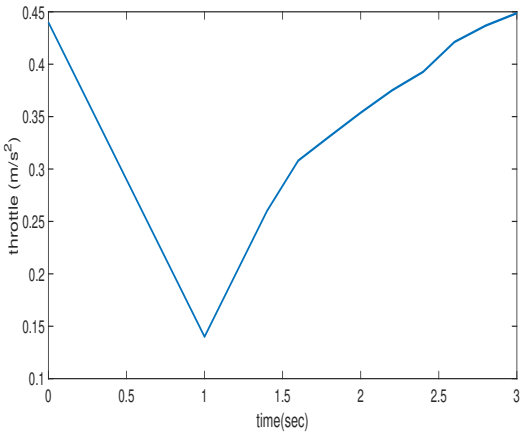


(c) Steering angle input for the ego vehicle to generate an evasive maneuver for high $p_{th} = 0.15$

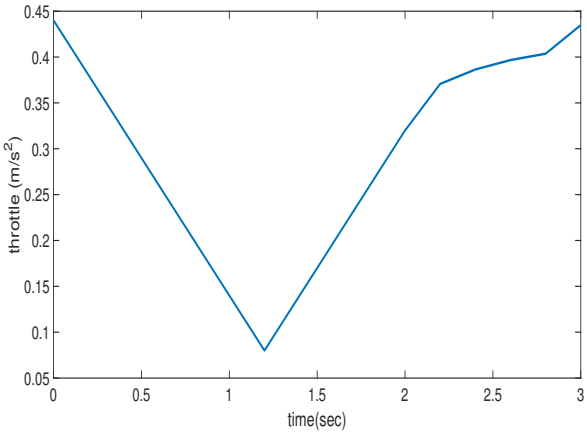
Figure 5.6: Ego vehicle's steering angle for crash avoidance with the oncoming vehicle for three different risk factors



(a) Throttle input for the ego vehicle to generate an evasive maneuver for low $p_{th} = 0.05$



(b) Throttle input for the ego vehicle to generate an evasive maneuver for medium $p_{th} = 0.10$



(c) Throttle input for the ego vehicle to generate an evasive maneuver for high $p_{th} = 0.15$

Figure 5.7: Ego vehicle's throttle input for crash avoidance with the oncoming vehicle for different risk factors.

relative distance between both vehicles when the threat was detected is taken as $10m$. The initial velocity of the ego vehicle and the surrounding vehicle is taken to be $25m/s$. The following figures show the simulation results for this scenario.

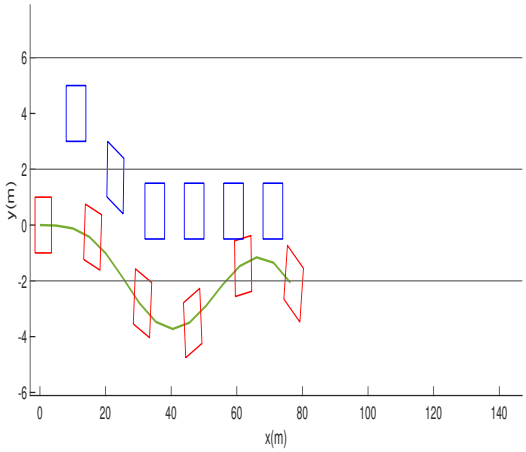
Figure 5.8 shows three cases of a cut-in scenario where the surrounding vehicle (blue) cuts into the lane of the ego vehicle. We can see that different values of risk factors affect the performance of the controller. These results are quite similar to the results in an oncoming vehicle. Figure 5.8a shows that low value of p_{th} led to conservative control actions and the ego vehicle went far into the adjacent lane to avoid a crash. This conservative behavior can be dangerous on the road, as the ego vehicle can crash with the other vehicle on the adjacent lane. Figure 5.8b shows an effective response to the cut-in vehicle where the evasive maneuver is not too conservative while not going too much inside the adjacent lane. Figure 5.8c shows that having a high value of risk factor p_{th} almost led to the crash at the time 1.8 seconds. This is due to the small SR sets from high p_{th}, which tends to miss much information of the likely achievable states for the surrounding vehicle.

In the first two cases, the ego vehicle was able to continuously generate safe convex hulls using algorithm 3 for the crash avoidance.

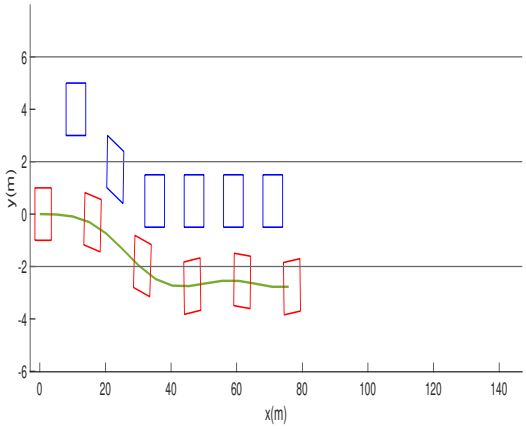
From Figure 5.9 and 5.10, it is evident that a lower value of risk factor p_{th} leads to more aggressive control actions. Significant effect can be observed from the steering angle differences for varying risk factors.

Left turn at the intersection

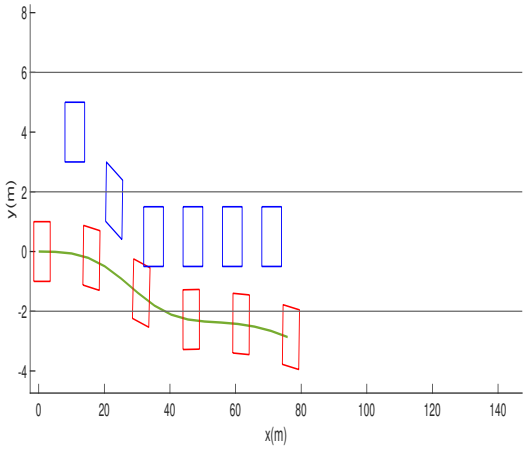
An intersection scenario is considered where a surrounding vehicle from the other side is turning left even though the ego vehicle has the right to go straight. Driver 6 aggressive waypoints for a left turn are considered. The initial relative longitudinal distance between



(a) Risk Factor $p_{th} = 0.05$ (low)

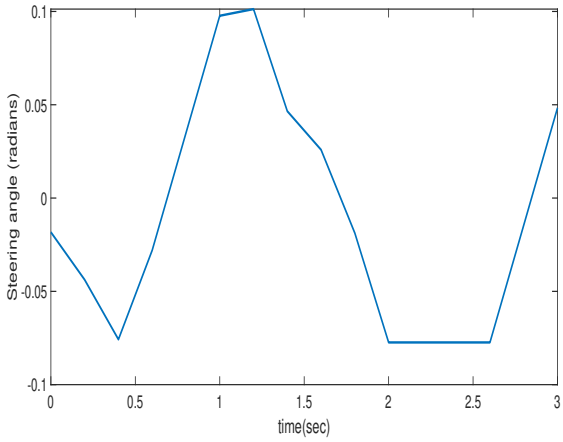


(b) Risk Factor $p_{th} = 0.10$ (medium)

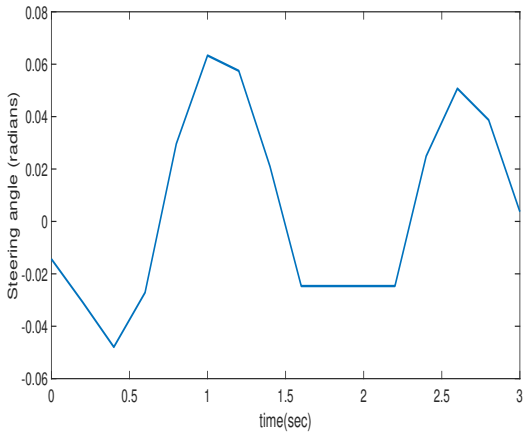


(c) Risk Factor $p_{th} = 0.15$ (high)

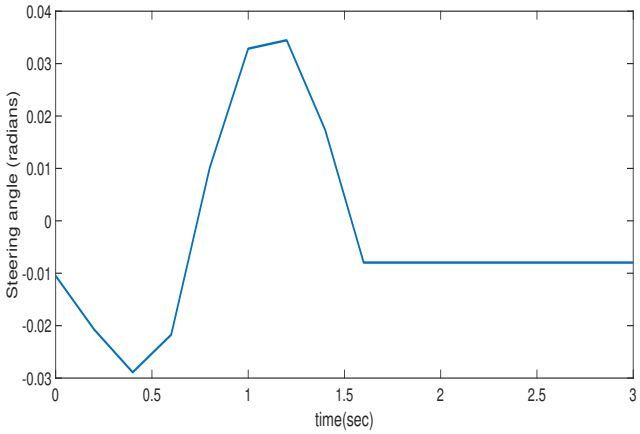
Figure 5.8: Visualization of the evasive maneuver by the ego vehicle (red) when the surrounding vehicle (blue) cuts into the lane of the ego vehicle. All vehicles are plotted at an interval of 0.6 seconds for a time interval of 3 seconds. The ego vehicle starts from left at $(0, 0)$ and the surrounding vehicle also starts from the right at a relative distance of $10m$



(a) Steering angle input for the Risk Factor p_{th} = 0.05 (low)

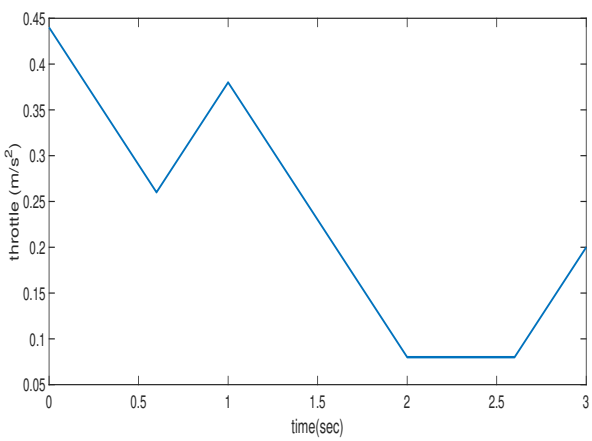


(b) Steering angle for the Risk Factor $p_{th} = 0.10$ (medium)

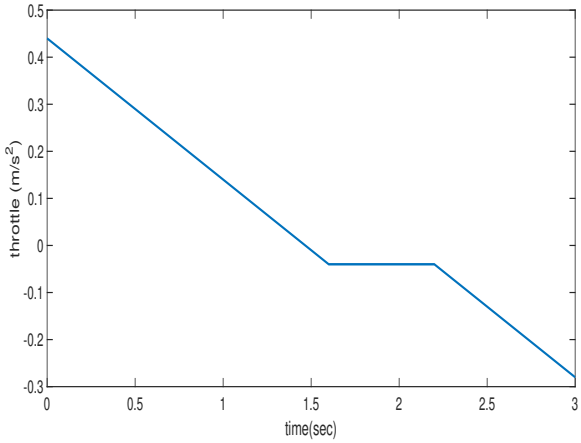


(c) Steering angle for the Risk Factor $p_{th} = 0.15$ (high)

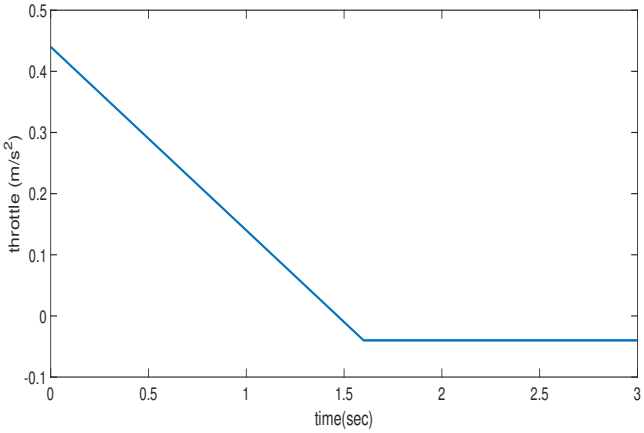
Figure 5.9: Ego vehicle's steering angle for avoiding crash with the cutting in vehicle at different risk factors.



(a) Throttle input for the risk Factor $p_{th} = 0.05$ (low)



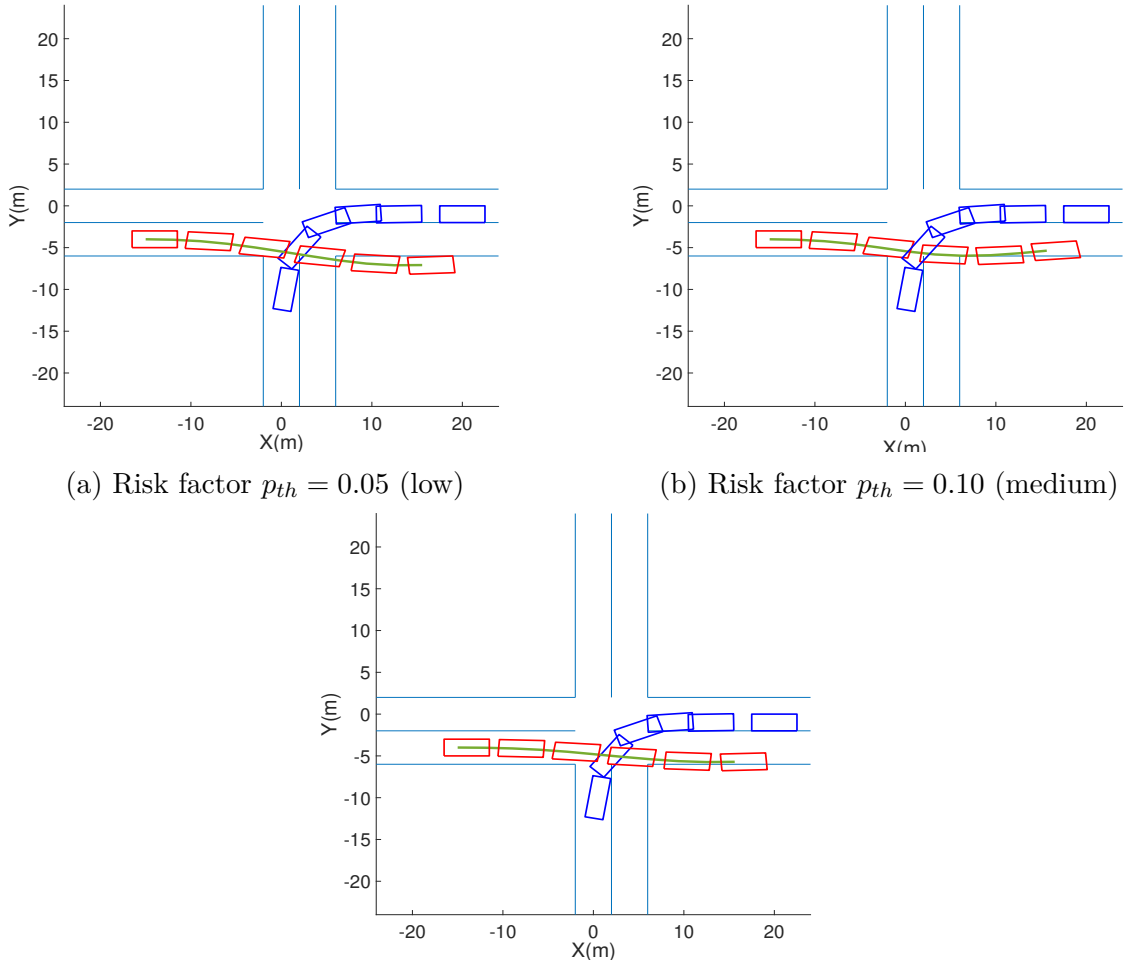
(b) Throttle input for the risk Factor $p_{th} = 0.1$ (medium)



(c) Throttle input for the risk Factor $p_{th} = 0.15$ (high)

Figure 5.10: Ego vehicle's acceleration and steering angle for crash avoidance with the oncoming vehicle

both vehicles when the threat was detected is taken to be $35m$. The initial velocity of both the ego vehicle and the surrounding vehicle is taken to be $15m/s$.



(a) Risk factor $p_{th} = 0.05$ (low)

(b) Risk factor $p_{th} = 0.10$ (medium)

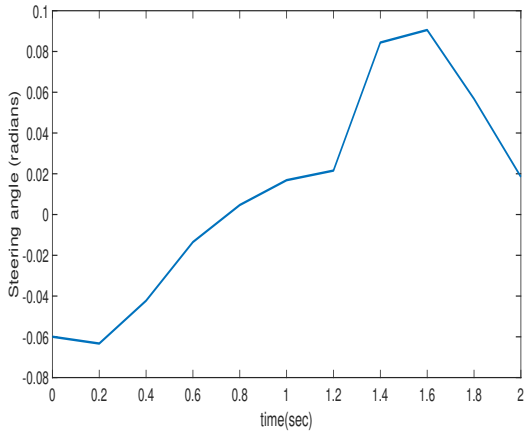
(c) Risk factor $p_{th} = 0.15$ (high)

Figure 5.11: Visualization of the evasive maneuver by the ego vehicle (red) when the surrounding vehicle (blue) turns left from the opposite lane in an intersection scenario. All vehicles are plotted at an interval of 0.4 seconds for a time interval of 2 seconds. The ego vehicle starts from left at $(-2, -15)$ and the surrounding vehicles starts from the right at (20,2).

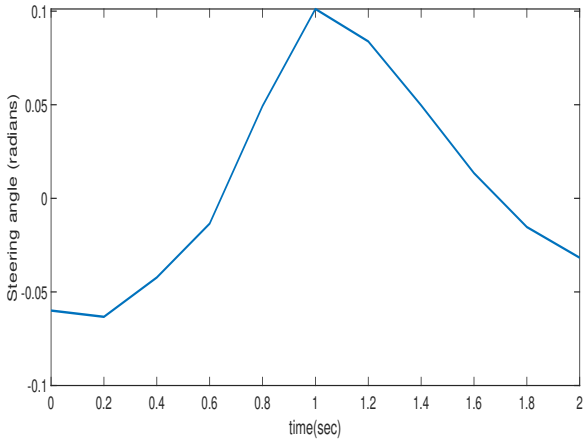
Figure 5.11 shows three cases in an intersection scenario with different risk parameters. Different risk factors resulted in different control actions for the proposed controller. Figure 5.11a shows the conservative control action taken by the controller for risk factor $p_{th} = 0.05$.

It went past the desired lane, which can put the ego vehicle in danger.

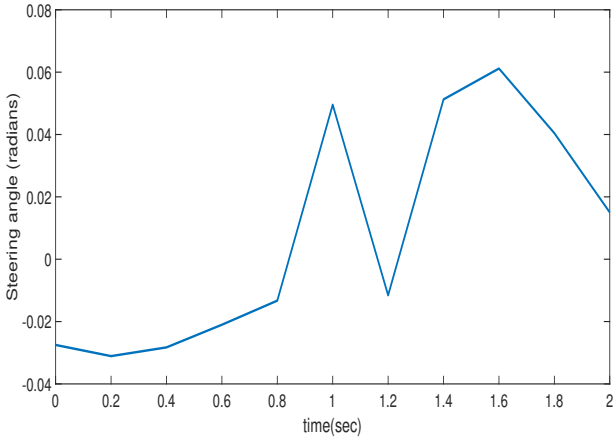
Figure 5.11b shows a better response to the surrounding vehicle turning left. It was able to avoid the crash. Figure 5.11c, the ego vehicle was just able to avoid the surrounding vehicle. The evasive maneuver is not too conservative while not going too much inside the adjacent lane. In the last two cases, the ego vehicle was able to continuously generate safe convex hulls using algorithm 3 for the crash avoidance with the other vehicle.



(a) Steering angle for low risk factor $p_{th} = 0.05$

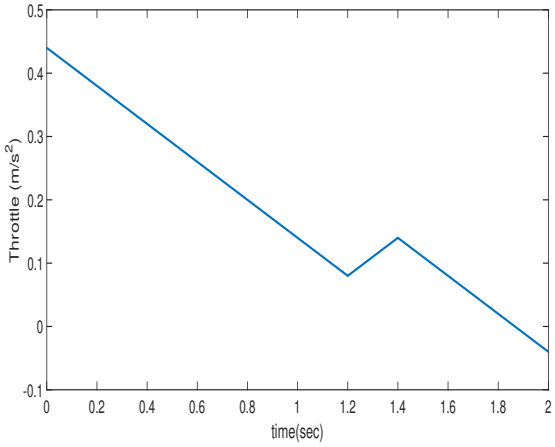


(b) Steering angle for medium risk factor $p_{th} = 0.10$

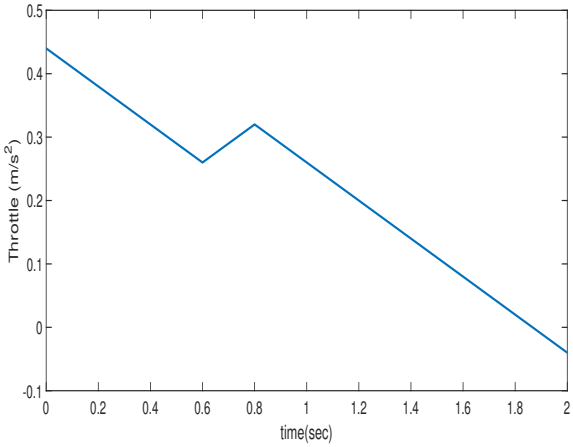


(c) Steering angle for high risk factor $p_{th} = 0.10$

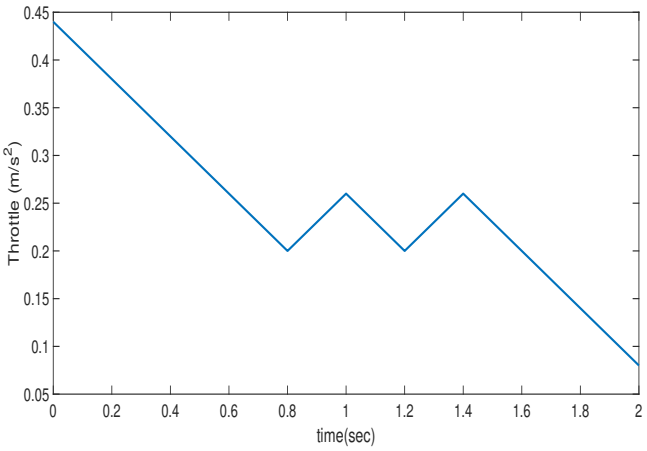
Figure 5.12: Ego vehicle's steering angle input for crash avoidance with the left turning vehicle with variations in the risk parameter.



(a) Throttle input for low risk factor $p_{th} = 0.05$



(b) Throttle input for medium risk factor $p_{th} = 0.10$



(c) Throttle input for high risk factor $p_{th} = 0.15$

Figure 5.13: Ego vehicle's throttle input for crash avoidance with the left turning vehicle with variations in the risk factor.

5.3.3 Variations in the relative longitudinal distance ΔD

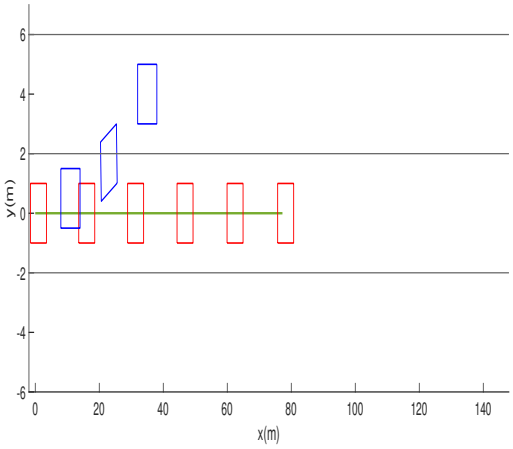
The performance and safety of the proposed SMPC framework are studied for different relative distances ΔD when the threat is detected. Three ranges of relative distances are considered for the oncoming vehicle case: low relative distance $\Delta D \in [0 - 35]m$, medium relative distance $\Delta D \in [35 - 80]m$, and large relative distance $\Delta D \in [80 - 130]m$. These variations will vary for the other two cases: the cut-in scenario and the intersection scenario.

Oncoming Vehicle

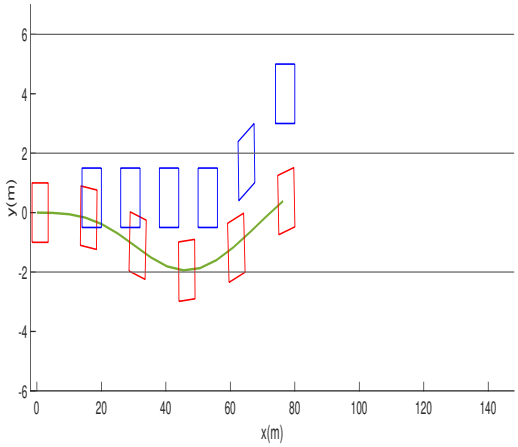
The initial velocity of the ego vehicle and the surrounding vehicle is taken to be $25m/s$. A medium-risk factor of 0.10 is considered to study the variations due to the relative distance ΔD at which the threat was detected. The data for the surrounding vehicle is taken from the driver 6 data of the driver behavior dataset.

Figure 5.14 shows three cases of an oncoming vehicle scenario where the surrounding vehicle (blue) drifts into the lane of the ego vehicle. Different relative distances ΔD resulted in different control actions for the proposed controller. Figure 5.14a shows that there was no optimal solution found for $\Delta D = 35m$, even for a medium risk factor $p_{th} = 0.10$. This implies that if the threat is detected too late, no optimal control action was found to avoid the crash. This will be a topic of future study that how to mitigate a crash in case of an imminent collision.

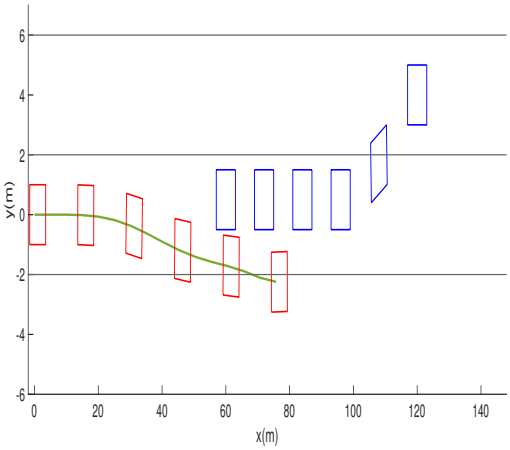
Figure 5.14b shows an effective response to the oncoming vehicle and where the ego vehicle can avoid the potential crash when the threat was detected at $80m$. The evasive maneuver is not too conservative while not going too much inside the adjacent lane. Figure 5.14c shows that having a detected threat at a high relative distance has a comfortable response from the ego vehicle, and it gets enough time to plan an evasive maneuver. This is one of the



(a) Low relative longitudinal distance $\Delta D = 35m$



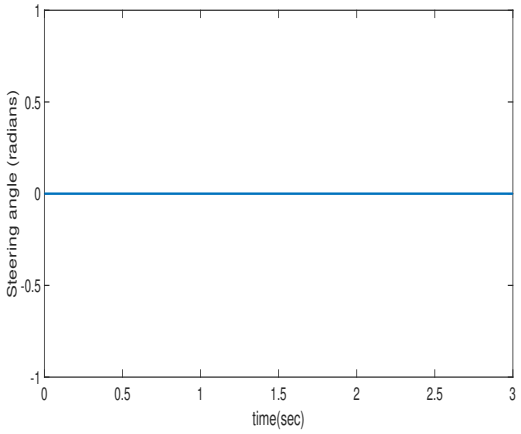
(b) Medium relative longitudinal distance $\Delta D = 80m$



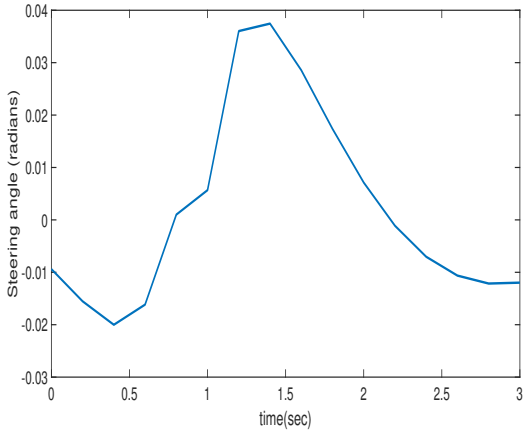
(c) High relative longitudinal distance $\Delta D = 120m$

Figure 5.14: Visualization of the evasive maneuver by the ego vehicle (red) when the surrounding vehicle (blue) drifts into the lane of the ego vehicle. All vehicles are plotted at an interval of 0.6 seconds for a time interval of 3 seconds. The ego vehicle starts from left at $(0, 0)$ and the surrounding vehicles starts from the right at different relative distances. Risk factor of 0.10 is considered.

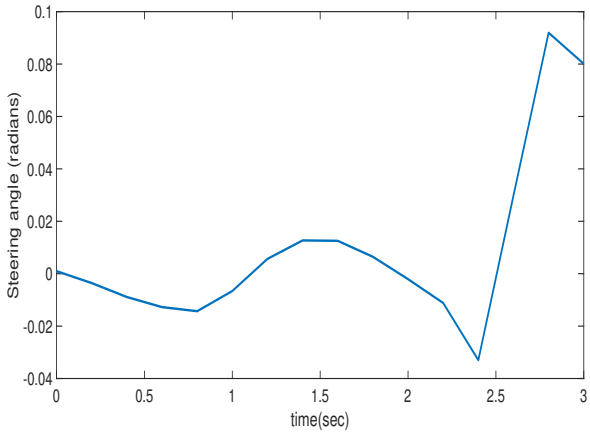
advantages of the proposed proactive decision-making algorithm. In the last two cases, the ego vehicle was able to continuously generate safe convex hulls using algorithm 3 for the crash avoidance.



(a) Steering angle for low relative distance $\Delta D = 35m$

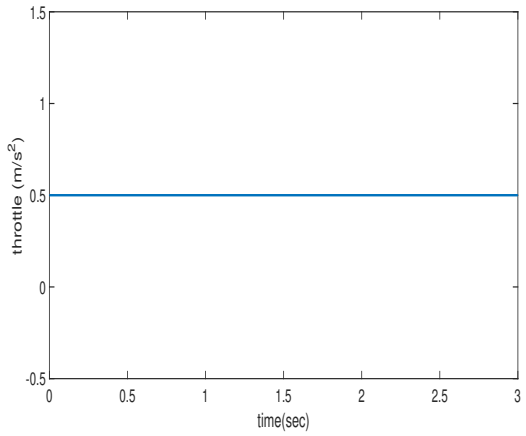


(b) Steering angle for medium relative distance $\Delta D = 80m$

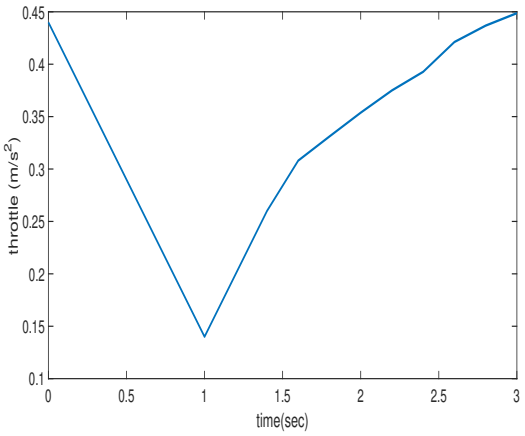


(c) Steering angle for high relative distance $\Delta D = 120m$

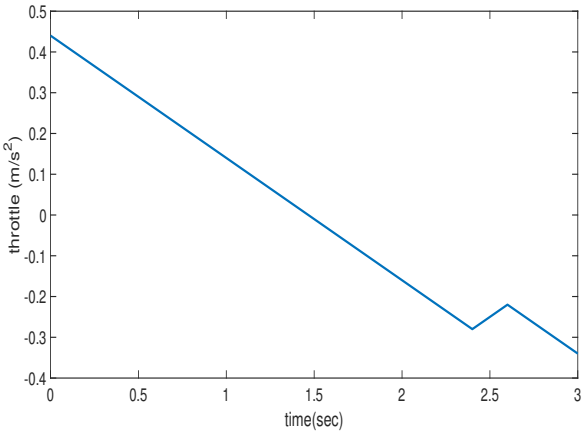
Figure 5.15: Ego vehicle's steering angle input for crash avoidance with the oncoming vehicle with variations in the relative distances of initial detected threat.



(a) Throttle input for low relative distance $\Delta D = 35m$



(b) Throttle input for medium relative distance $\Delta D = 80m$



(c) Throttle input for high relative distance $\Delta D = 120m$

Figure 5.16: Ego vehicle's throttle input for crash avoidance with the oncoming vehicle with variations in the relative distances of initial detected threat.

Cut-in Scenario

In the case of cut-in scenario, different ranges of relative longitudinal distances are considered i.e. : low $10m$, medium $20m$ and high $40m$.

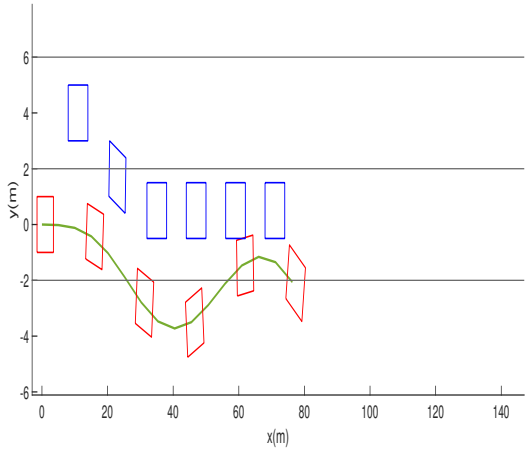
Figure 5.17 shows three cases of a surrounding vehicle (blue) cutting into the lane of the ego vehicle. Different relative distances ΔD resulted in different control actions for the proposed controller. Figure 5.17a shows the aggressive control action and a sharp maneuver taken by the ego vehicle when the threat was detected at $\Delta D = 10m$. This implies that the control action will be aggressive if the relative distance is low.

Figure 5.17b shows an effective response to the oncoming vehicle and where the ego vehicle can brake and maneuver to avoid the potential crash when the threat was detected at $20m$. The evasive maneuver is not too conservative while not going too much inside the adjacent lane. Figure 5.17c shows no need for the ego vehicle to maneuver, and simple braking was enough to avoid a potential collision with the surrounding vehicle. In all three cases, the ego vehicle was able to continuously generate safe convex hulls using algorithm 3 for the crash avoidance.

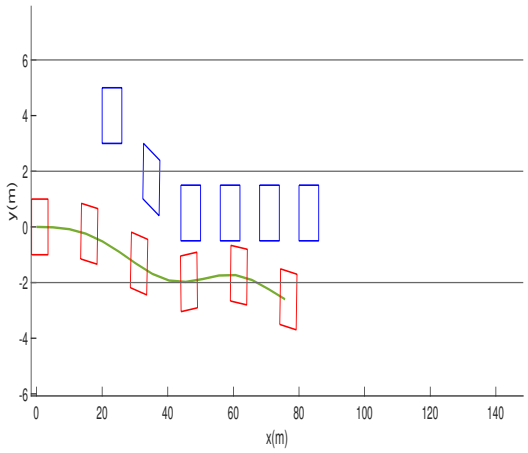
Left turn at the intersection

In this scenario, different ranges of relative longitudinal distances are considered when the threat was detected i.e. : low $25m$, medium $35m$ and high $45m$.

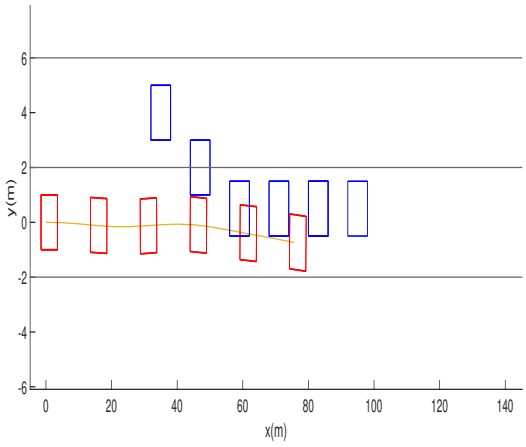
Figure 5.20 shows three cases at an intersection scenario. Different relative distances ΔD resulted in different control actions for the proposed controller. Figure 5.20a shows the ego vehicle could not respond effectively when the threat was detected at a lower relative distance.



(a) Low relative longitudinal distance $\Delta D = 10m$

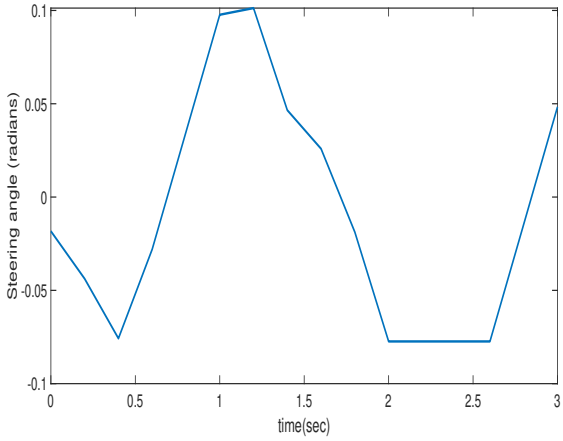


(b) Medium relative longitudinal distance $\Delta D = 20m$

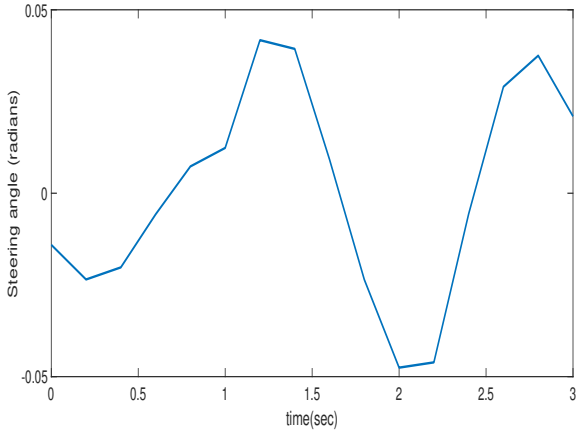


(c) High relative longitudinal distance $\Delta D = 40m$

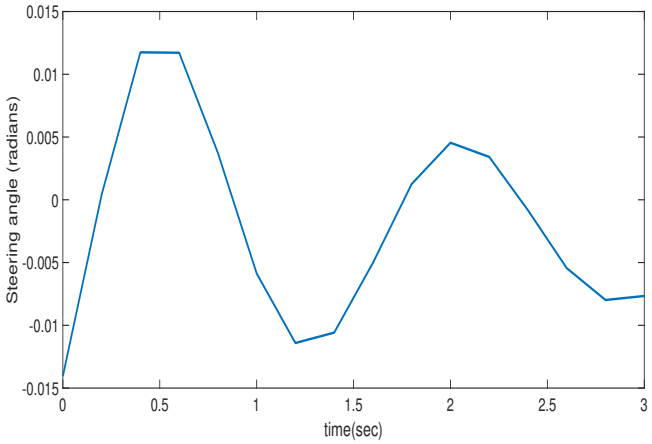
Figure 5.17: Visualization of the evasive maneuver by the ego vehicle (red) when the surrounding vehicle (blue) cuts into the lane of the ego vehicle. All vehicles are plotted at an interval of 0.6 seconds for a time interval of 3 seconds. The ego vehicle starts from left at $(0, 0)$ and the surrounding vehicle also starts from the left at different relative distances. Risk factor of 0.10 is considered.



(a) Steering angle input for low relative longitudinal distance $\Delta D = 10m$

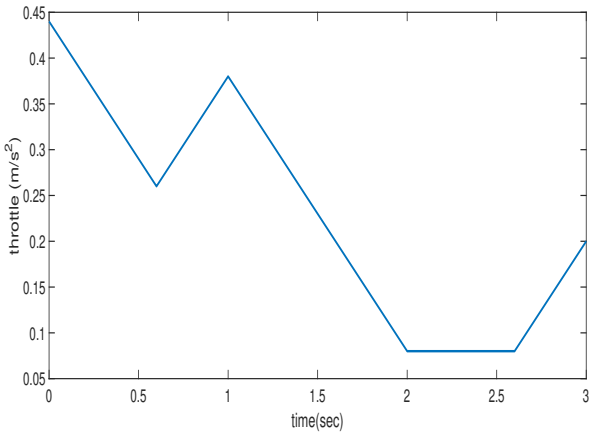


(b) Steering angle input for medium relative longitudinal distance $\Delta D = 20m$

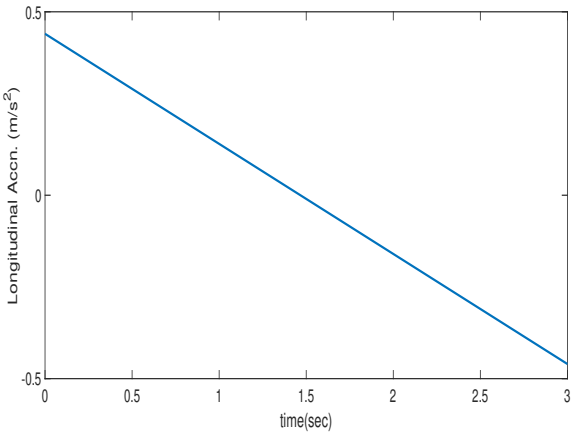


(c) Steering angle input for high relative longitudinal distance $\Delta D = 40m$

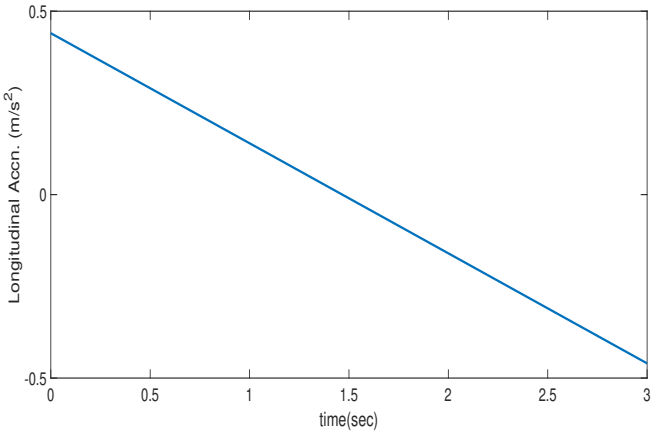
Figure 5.18: Ego vehicle's steering angle input for crash avoidance with the cutting-in vehicle for different relative longitudinal distances.



(a) Throttle input for low relative longitudinal distance $\Delta D = 10m$

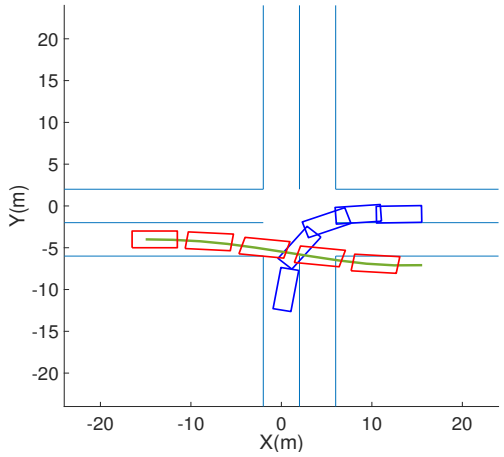


(b) Throttle input for medium relative longitudinal distance $\Delta D = 20m$

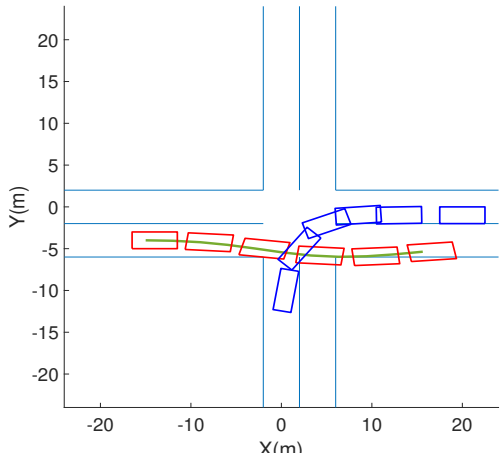


(c) Throttle input for high relative longitudinal distance $\Delta D = 40m$

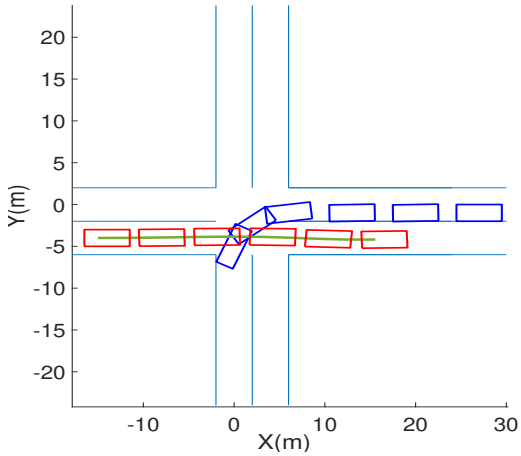
Figure 5.19: Ego vehicle's throttle inputs for crash avoidance with the cut in vehicle for different relative longitudinal distances at which the threat was detected.



(a) Low relative longitudinal distance $\Delta D = 25m$



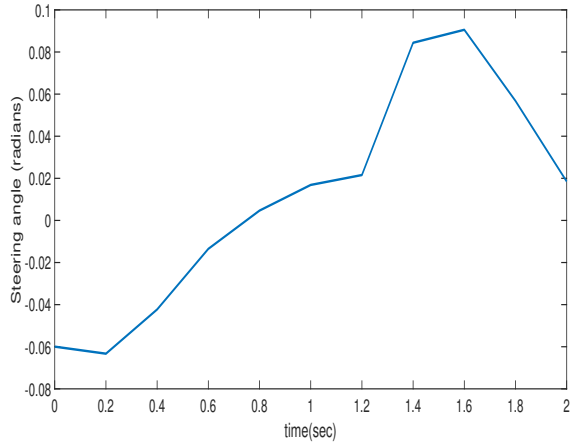
(b) Medium relative longitudinal distance $\Delta D = 35m$



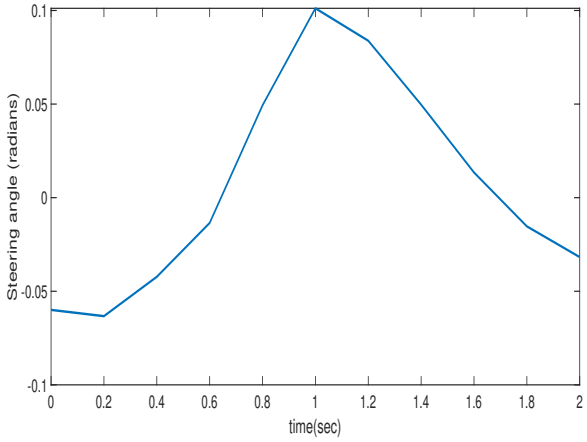
(c) High relative longitudinal distance $\Delta D = 45m$

Figure 5.20: Visualization of the evasive maneuver by the ego vehicle (red) when the surrounding vehicle (blue) turns left. All vehicles are plotted at an interval of 0.4 seconds for a time interval of 2 seconds. The ego vehicle starts from left at $(-2, 15)$ and the surrounding vehicle starts at different relative distances. Risk factor of 0.10 is considered.

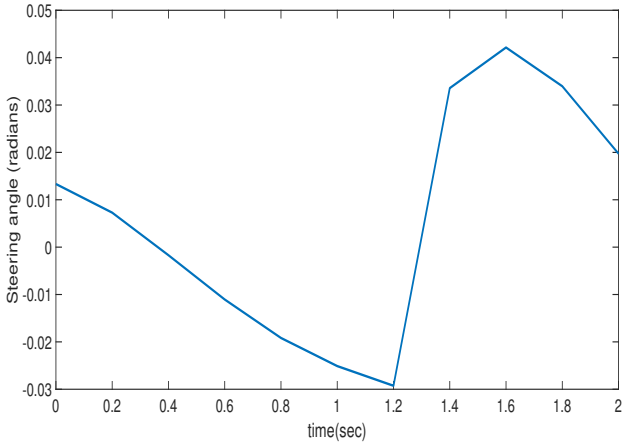
Figure 5.20b shows an effective response to the oncoming vehicle and where the ego vehicle can brake and maneuver to avoid the potential crash when the threat was detected at $35m$. The evasive maneuver is not too conservative while not going too much inside the adjacent lane. Figure 5.20c shows the comfortable control action by the ago vehicle in avoiding the crash. In the last two cases, the ego vehicle was able to effectively and continuously generate safe convex hulls using algorithm 3 for the crash avoidance.



(a) Steering angle input for low relative longitudinal distance $\Delta D = 25m$

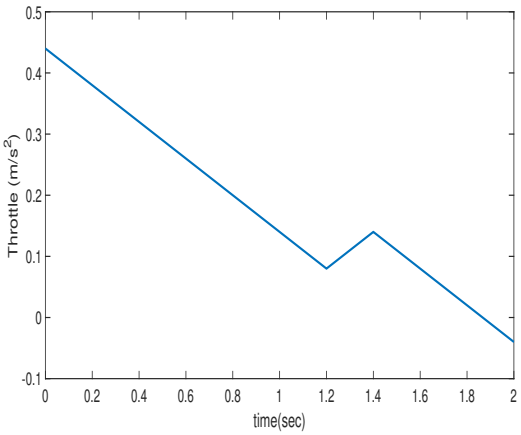


(b) Steering angle input for medium relative longitudinal distance $\Delta D = 35m$

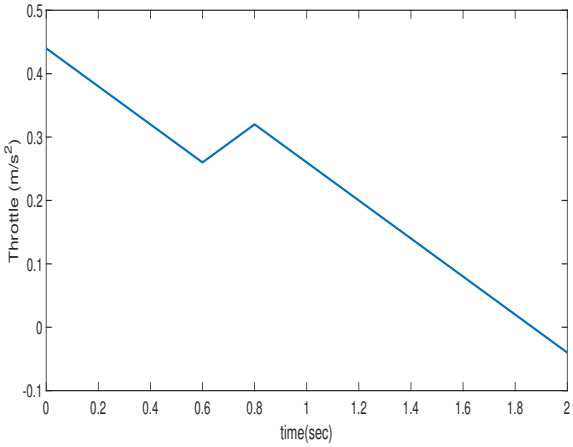


(c) Steering angle input for high relative longitudinal distance $\Delta D = 45m$

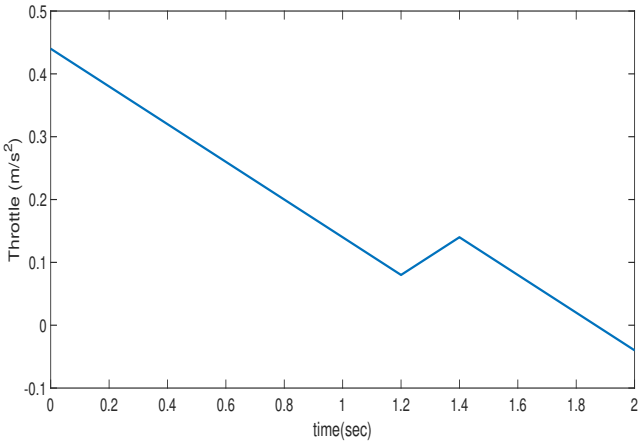
Figure 5.21: Ego vehicle's steering angle input for crash avoidance with the left turning vehicle for different relative longitudinal distances.



(a) Throttle input for low relative longitudinal distance $\Delta D = 25m$



(b) Throttle input for medium relative longitudinal distance $\Delta D = 35m$



(c) Throttle input for high relative longitudinal distance $\Delta D = 45m$

Figure 5.22: Ego vehicle's throttle inputs for crash avoidance with the left turninig for different relative longitudinal distances at which the threat was detected.

5.3.4 Variations in the relative velocities ΔV

The performance and safety of the proposed SMPC framework are studied for different relative velocities ΔV when the threat is detected. Three ranges of relative velocities are considered: low relative velocity $0 - 20m/s$, medium relative velocity $20 - 40m/s$, and large relative velocity $40 - 80/sm$. These variations will be considered for three road scenarios: Oncoming vehicle, cut-in scenario, and the intersection scenario. The risk factor is taken to be 0.10. The data for the surrounding vehicle is taken from the driver 6 data of the driver behavior dataset.

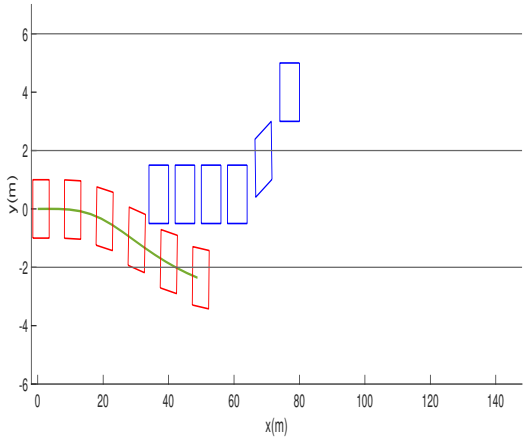
Oncoming Vehicle

We can see from Figure 5.23 that the ego vehicle is able to avoid the oncoming vehicle from an adjacent lane. It is able to continuously generate safe convex hulls using algorithm 3 for the crash avoidance.

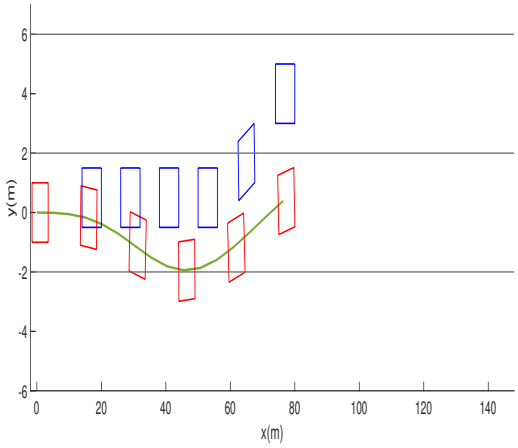
Left turn at the intersection

In this scenario, different ranges of relative velocities are considered when the threat was detected i.e. : low $\Delta V = 20m/s$, medium $\Delta V = 30m/s$ and $\Delta V = 40m/s$. The initial relative distance is taken as $35m$ and the risk factor p_{th} is considered to be 0.10.

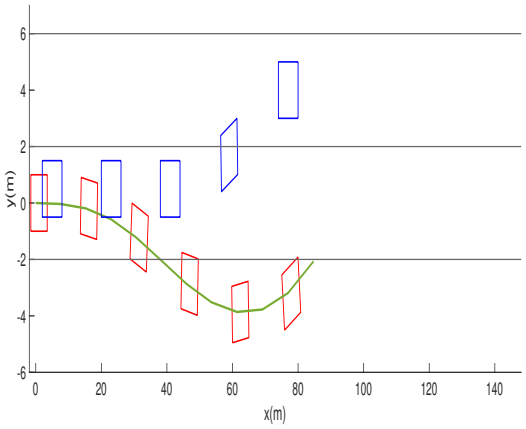
We can see from Figure 5.26a that the ego vehicle is able to avoid the oncoming vehicle from an adjacent lane. It is able to continuously generate safe convex hulls using algorithm 3 for the crash avoidance.



(a) Low relative velocity $\Delta V = 20$m/s

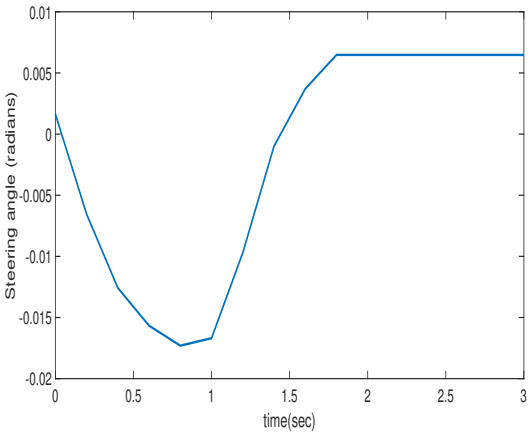


(b) Medium relative velocity $\Delta V = 40$m/s

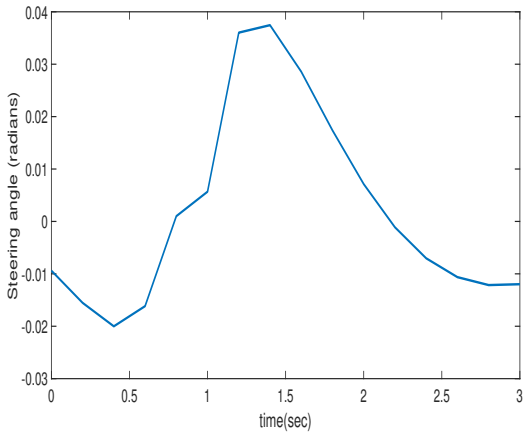


(c) Risk Factor $p_{th} = 0.15$ (high)

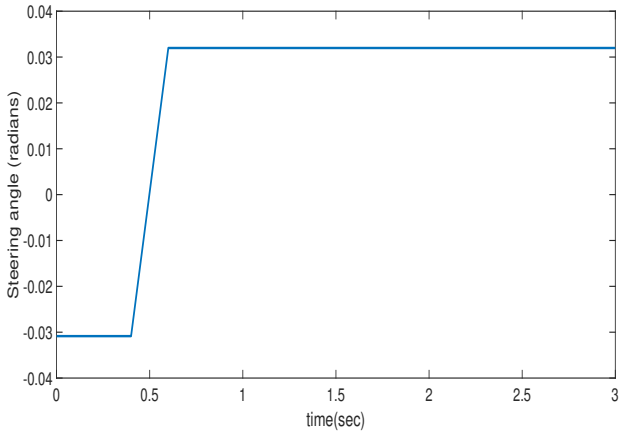
Figure 5.23: Ego vehicle's acceleration and steering angle for crash avoidance with the oncoming vehicle



(a) Steering angle input for low relative velocity $\Delta V = 20m/s$

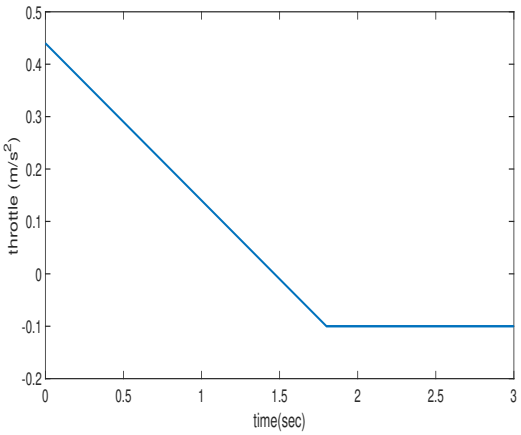


(b) Steering angle input for medium relative velocity $\Delta V = 40m/s$

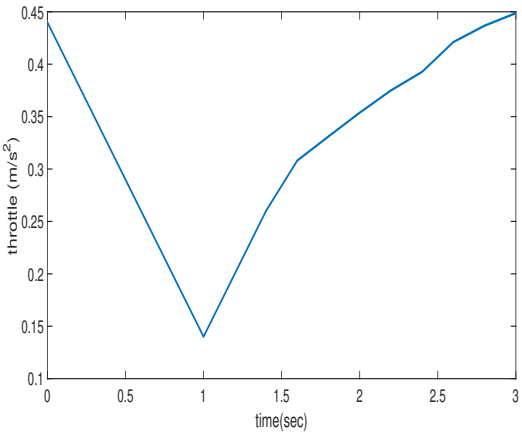


(c) Steering angle input for high relative velocity $\Delta V = 800m/s$

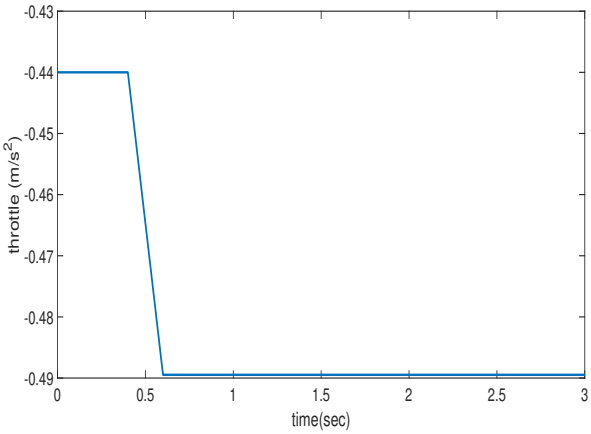
Figure 5.24: Ego vehicle's steering angle input for crash avoidance with the oncoming vehicle w.r.t. varying relative longitudinal distances



(a) Throttle input for low relative velocity $\Delta V = 20m/s$

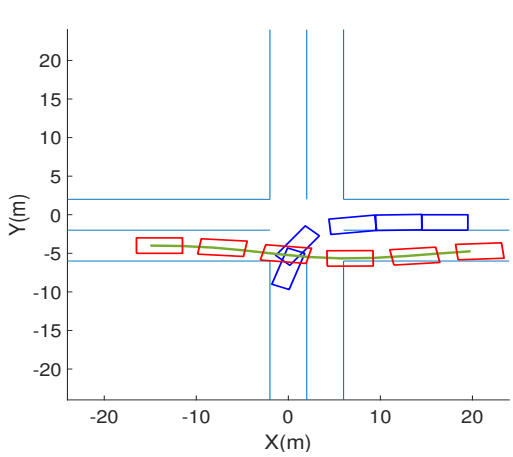


(b) Throttle input for medium relative velocity $\Delta V = 40m/s$

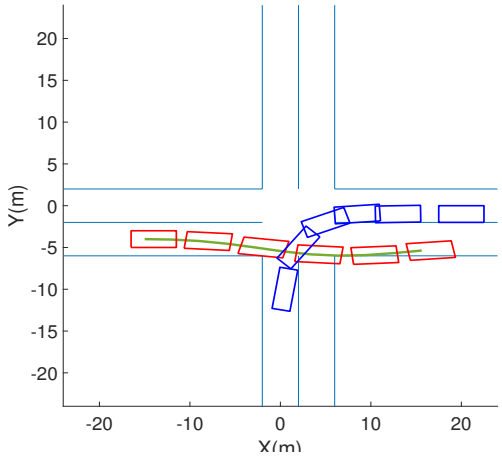


(c) Throttle input for high relative velocity $\Delta V = 80m/s$)

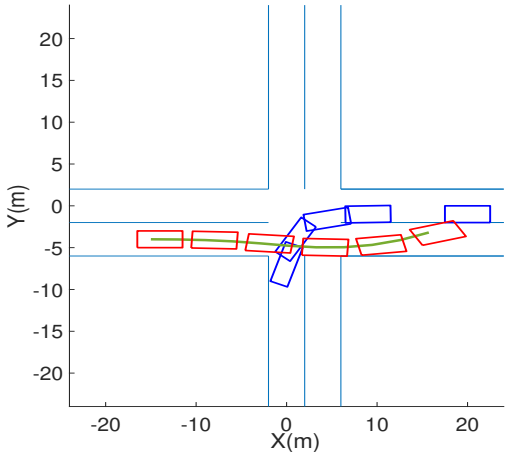
Figure 5.25: Ego vehicle's Throttle input for crash avoidance with the oncoming vehicle w.r.t. varying relative velocities at which threat was detected.



(a) Low relative velocity $\Delta V = 20m/s$

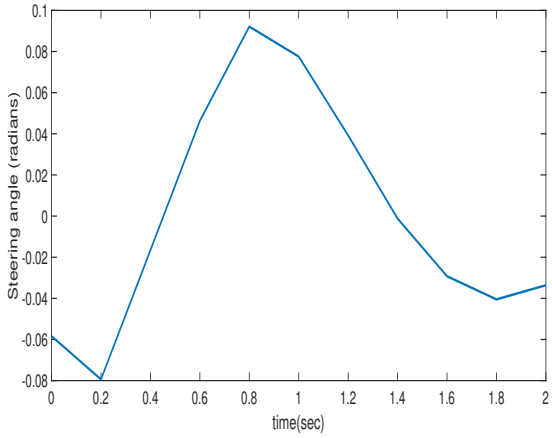


(b) Medium relative velocity $\Delta V = 30m/s$

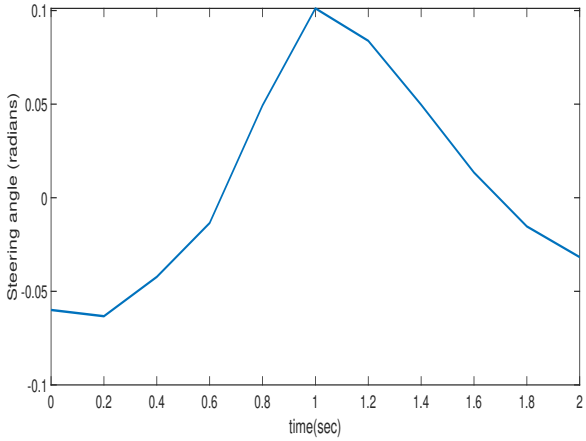


(c) High Relative velocity $\Delta V = 40m/s$

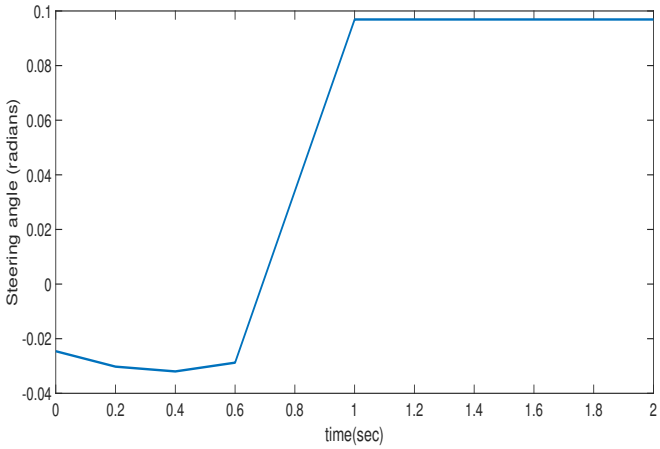
Figure 5.26: Visualization of the evasive maneuver by the ego vehicle (red) when the surrounding vehicle (blue) turns left from the opposite lane in an intersection scenario. All vehicles are plotted at an interval of 0.4 seconds for a time interval of 2 seconds. The ego vehicle starts from left at $(-2, -15)$ and the surrounding vehicles starts from the right at (20,2) with different velocities.



(a) Steering input for low relative velocity $\Delta V = 20m/s$

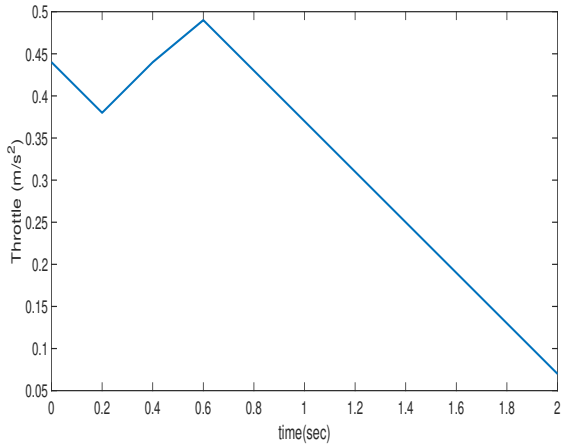


(b) Steering input for medium relative velocity $\Delta V = 30m/s$

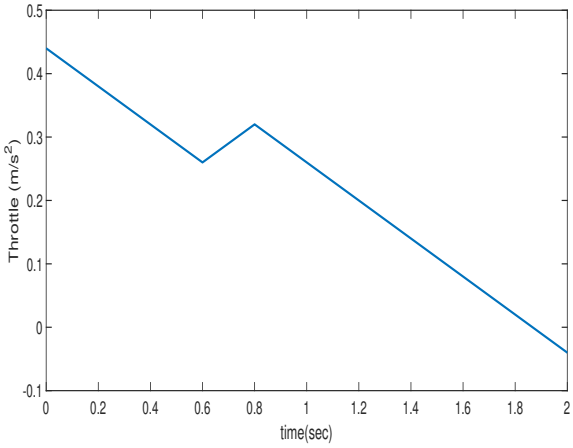


(c) Steering input for high relative velocity $\Delta V = 40m/s$

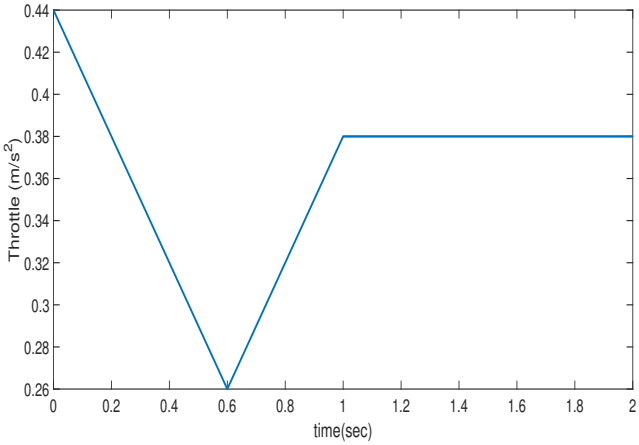
Figure 5.27: Ego vehicle's steering angle for crash avoidance with the left turning vehicle



(a) Throttle input for low relative velocity $\Delta V = 20m/s$



(b) Throttle input for medium relative velocity $\Delta V = 30m/s$



(c) Throttle input for high relative velocity $\Delta V = 40m/s$

Figure 5.28: Ego vehicle's throttle input for crash avoidance with the left turning vehicle with varying relative velocities at the time of detected threat.

5.4 Summary and Discussion

This chapter presented the evaluation and validation of the proposed threat assessment and SMPC decision-making algorithms. The average computation time for the threat assessment algorithm was around 38 ms. This low computation time makes it practical for real-time threat assessment applications. Moreover, the proposed algorithm achieved a high accuracy for driver behavior prediction and intention estimation. The percentage of false positives and negatives was lower than 10% for the cut-in scenario and oncoming vehicles. The proposed threat assessment methodology could be made more robust to outliers using more extensive naturalistic driving data for different driver behaviors. Intention estimation accuracy could also be improved using trust-based intention estimation using trust values from all the on-board sensors.

The proposed SMPC framework is also tested on three frequently occurring hazardous road scenarios on the road, i.e., oncoming vehicle, cut-in scenario, and turning left when there is no right of way. All three scenarios are tested for different initial conditions for the controller, i.e., variations in the risk factor p_{th}, variations in the relative distance ΔD at which the threat was detected, and the relative velocity ΔV at which the threat was detected.

It was observed that the choice of the risk factor p_{th} had a significant impact on the controller's performance. A lower risk factor led to aggressive control actions and can potentially get the vehicle into hazardous situations. Moreover, a low value of the risk factor can also lead to an infeasible solution. On the other hand, a high risk factor leads to smooth control action but ignores many likely positions a surrounding vehicle can occupy. This can result in a crash with the other vehicle, as we saw in some of our simulations with a high risk factor. Overall, it can be said that it is not easy to decide the value of the risk factor for the SMPC controller. A not-so-high risk factor can result in better responses to hazardous situations

on the road.

The relative velocity ΔV and the relative distance ΔD at which the threat is detected also affects the controller's overall performance. The algorithm was not able to produces feasible solutions in a few cases where ΔD was low. Overall, the SMPC decision-making algorithm is able to avoid crashes in most of the hazardous situations.

Some of the limitations for the proposed threat assessment and decision-making framework are as follows:

1. The driver behavior dataset only consists of 6 drivers emulating three different driver behaviors. More naturalistic driving data can make the proposed threat assessment approach more reliable.

2. The proposed method can be computationally expensive for multiple vehicles.

3. Some of the safe regions are not considered during the deterministic reformulation of the SMPC framework, which can sometimes lead to infeasible solutions for near-crash scenarios.

Chapter 6

Conclusion and Future Work

This thesis presented a proactive decision-making algorithm for autonomous vehicles for avoiding crash situations on the road. Two major challenges were addressed in this thesis: the outside challenge (motion prediction and threat assessment); and the inside challenge (decision-making). A reliable motion prediction method was proposed on stochastic reachable sets and driver behavior prediction. A trust value was established for the driver behavior prediction using trust values from all the sensors. This allowed quantifying the reliability of the driver behavior prediction using the proposed method. Using multiple sensors for driver behavior prediction and threat assessment makes the proposed method highly robust to any individual sensor failures. A stochastic reachable set threat assessment was formulated using the probability of collision. Two quantities were formulated for assessing threat: Current Threat (CT) and Short Term Prediction Threat (STPT). A positive value of STPT at any time can give ego vehicle enough time to implement a crash avoidance control action. This was covered in the decision-making part of the thesis.

The proactive decision-making is done using the Stochastic Model Predictive Control, which uses the stochastic reachable sets and the future predicted threat information to take control action for an evasive maneuver. A deterministic reformulation of the chance-constrained optimization problem led to fast real-time implementation of the SMPC framework. This thesis studied the effects of different controller parameters and initial conditions on the proposed decision-making algorithm. The risk factor p_{th} is not trivial to decide. From the

results in Chapter 5, a medium value of p_{th} led to optimal and safe controller actions.

The future work will validate this proposed proactive decision-making approach on mobile robots in a lab setup. Moreover, this decision-making framework will be extended to connected autonomous vehicles, where sensor reliability is a major issue. Trust-based motion predictions in a cooperative vehicles environment can help overcome the sensor reliability challenge. This will allow to test the methodology for multiple surrounding vehicles. The future work will also investigate dynamically changing risk factors for the SMPC controller depending on the threat of the situation. A more extensive naturalistic driving data can be used to validate and make our method more robust.

Bibliography

[1] M. Althoff, O. Stursberg, and M. Buss. Model-based probabilistic collision detection in autonomous driving. *IEEE Transactions on Intelligent Transportation Systems*, 10(2):299–310, 2009.

[2] Mark Stamp. A revealing introduction to hidden markov models. *Department of Computer Science San Jose State University*, pages 26–56, 2004.

[3] World Health Organization. Global status report on road safety, 2018.

[4] Statista Dossier Plus. Statista dossierplus on the market for autonomous vehicles, 2019.

[5] Julian Straub, Sue Zheng, and John W Fisher. Bayesian nonparametric modeling of driver behavior. In *2014 IEEE Intelligent Vehicles Symposium Proceedings*, pages 932–938. IEEE, 2014.

[6] Sara Moridpour, Majid Sarvi, Geoff Rose, and Ehsan Mazloumi. Lane-changing decision model for heavy vehicle drivers. *Journal of Intelligent Transportation Systems*, 16(1):24–35, 2012.

[7] Matthias Althoff. *Reachability Analysis and its Application to the Safety Assessment of Autonomous Cars*. PhD thesis, 07 2010.

[8] Shilp Dixit, Umberto Montanaro, Mehrdad Dianati, David Oxtoby, Tom Mizutani, Alexandros Mouzakitis, and Saber Fallah. Trajectory planning for autonomous high-speed overtaking in structured environments using robust mpc. *IEEE Transactions on Intelligent Transportation Systems*, 21(6):2310–2323, 2019.

[9] W. Langson, I. Chryssochoos, S.V. Raković, and D.Q. Mayne. Robust model predictive control using tubes. *Automatica*, 40(1):125–133, 2004.

[10] J. Suh, H. Chae, and K. Yi. Stochastic model-predictive control for lane change decision of automated driving vehicles. *IEEE Transactions on Vehicular Technology*, 67(6):4771–4782, 2018.

[11] Vanshaj Khattar and Azim Eskandarian. Reactive online motion re-planning for crash mitigation in autonomous vehicles using bezier curve optimization. In *ASME International Mechanical Engineering Congress and Exposition*, volume 84553, page V07BT07A016. American Society of Mechanical Engineers, 2020.

[12] Stephanie Lefevre, Dizan Vasquez, and Christian Laugier. A survey on motion prediction and risk assessment for intelligent vehicles. *Robomech Journal*, 1, 07 2014.

[13] N. Kaempchen, B. Schiele, and K. Dietmayer. Situation assessment of an autonomous emergency brake for arbitrary vehicle-to-vehicle collision scenarios. *IEEE Transactions on Intelligent Transportation Systems*, 10(4):678–687, 2009.

[14] R. Pepy, A. Lambert, and H. Mounier. Reducing navigation errors by planning with realistic vehicle model. In *2006 IEEE Intelligent Vehicles Symposium*, pages 300–307, 2006.

[15] Jihua Huang and Han-Shue Tan. Vehicle future trajectory prediction with a dgps/ins-based positioning system. In *2006 American Control Conference*, pages 6 pp.–, 2006.

[16] Shiwen Liu, Kan Zheng, Long Zhao, and Pingzhi Fan. A driving intention prediction method based on hidden markov model for autonomous driving. *Computer Communications*, 157:143 – 149, 2020.

[17] S. A. Goli, B. H. Far, and A. O. Fapojuwo. Vehicle trajectory prediction with gaussian process regression in connected vehicle environment⋆. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 550–555, 2018.

[18] C. Hermes, C. Wohler, K. Schenk, and F. Kummert. Long-term vehicle motion prediction. In *2009 IEEE Intelligent Vehicles Symposium*, pages 652–657, 2009.

[19] A. Houenou, P. Bonnifait, V. Cherfaoui, and W. Yao. Vehicle trajectory prediction based on motion model and maneuver recognition. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4363–4369, 2013.

[20] C. Yang, A. Renzaglia, A. Paigwar, C. Laugier, and D. Wang. Driving behavior assessment and anomaly detection for intelligent vehicles. In *2019 IEEE International Conference on Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM)*, pages 524–529, 2019.

[21] H. Harkous, C. Bardawil, H. Artail, and N. Daher. Application of hidden markov model on car sensors for detecting drunk drivers. In *2018 IEEE International Multidisciplinary Conference on Engineering Technology (IMCET)*, pages 1–6, 2018.

[22] V. Lefkopoulos, M. Menner, A. Domahidi, and M. N. Zeilinger. Interaction-aware motion prediction for autonomous driving: A multiple model kalman filtering scheme. *IEEE Robotics and Automation Letters*, 6(1):80–87, 2021.

[23] Mark Pfeiffer, Giuseppe Paolo, Hannes Sommer, Juan Nieto, Roland Siegwart, and Cesar Cadena. A data-driven model for interaction-aware pedestrian motion prediction in object cluttered environments, 2018.

[24] Y. Li, Y. Zheng, B. Morys, S. Pan, J. Wang, and K. Li. Threat assessment techniques

in intelligent vehicles: A comparative survey. *IEEE Intelligent Transportation Systems Magazine*, pages 1–1, 2020.

[25] David Lee. A theory of visual control of braking based on information about time-to-collision. *Perception*, 5:437–59, 02 1976.

[26] C. G. Keller, T. Dang, H. Fritz, A. Joos, C. Rabe, and D. M. Gavrila. Active pedestrian safety by automatic braking and evasive steering. *IEEE Transactions on Intelligent Transportation Systems*, 12(4):1292–1304, 2011.

[27] Katja Kircher. A comparison of headway and time to collision as safety indicators. *Accident; analysis and prevention*, 35:427–33, 06 2003.

[28] Rulin Huang, Huawei Liang, Pan Zhao, Biao Yu, and Xinli Geng. Intent-estimation- and motion-model-based collision avoidance method for autonomous vehicles in urban environments. *Applied Sciences*, 7:457, 04 2017.

[29] J. E. Stellet, P. Vogt, J. Schumacher, W. Branz, and J. M. Zöllner. Analytical derivation of performance bounds of autonomous emergency brake systems. In *2016 IEEE Intelligent Vehicles Symposium (IV)*, pages 220–226, 2016.

[30] Pongsathorn Raksincharoensak, Yuta Akamatsu, Katsumi Moro, and Masao Nagai. Driver speed control modeling for predictive braking assistance system based on risk potential in intersections. *Journal of Robotics and Mechatronics*, 26:628–637, 10 2014.

[31] C. Yang, A. Renzaglia, A. Paigwar, C. Laugier, and D. Wang. Driving behavior assessment and anomaly detection for intelligent vehicles. In *2019 IEEE International Conference on Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM)*, pages 524–529, 2019.

[32] S. Lefèvre, C. Laugier, and J. Ibañez-Guzmán. Evaluating risk at road intersections by detecting conflicting intentions. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4841–4846, 2012.

[33] Nicholas Merrill and Azim Eskandarian. Modified autoencoder training and scoring for robust unsupervised anomaly detection in deep learning. *IEEE Access*, PP:1–1, 05 2020.

[34] Kazuhide Okamoto, Karl Berntorp, and Stefano Di Cairano. Driver intention-based vehicle threat assessment using random forests and particle filtering. *IFAC-PapersOnLine*, 50(1):13860 – 13865, 2017. 20th IFAC World Congress.

[35] Virginia Tech Transportation Institute. Behavior-based predictive safety analytics – pilot study, 2019.

[36] D. González, J. Pérez, V. Milanés, and F. Nashashibi. A review of motion planning techniques for automated vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 17(4):1135–1145, 2016.

[37] J. Choi, R. Curry, and G. Elkaim. Path planning based on bézier curve for autonomous ground vehicles. In *Advances in Electrical and Electronics Engineering - IAENG Special Edition of the World Congress on Engineering and Computer Science 2008*, pages 158–166, 2008.

[38] J. A. R. Silva and V. Grassi. Clothoid-based global path planning for autonomous vehicles in urban scenarios. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4312–4318, 2018.

[39] C. G. Keller, T. Dang, H. Fritz, A. Joos, C. Rabe, and D. M. Gavrila. Active pedestrian safety by automatic braking and evasive steering. *IEEE Transactions on Intelligent Transportation Systems*, 12(4):1292–1304, 2011.

[40] Maxim Likhachev and Dave Ferguson. Planning long dynamically feasible maneuvers for autonomous vehicles. *The International Journal of Robotics Research*, 28(8):933–945, 2009.

[41] Dmitri Dolgov, Sebastian Thrun, Michael Montemerlo, and James Diebel. Path planning for autonomous vehicles in unknown semi-structured environments. *The International Journal of Robotics Research*, 29(5):485–501, 2010.

[42] Y. Kuwata, J. Teo, G. Fiore, S. Karaman, E. Frazzoli, and J. P. How. Real-time motion planning with applications to autonomous urban driving. *IEEE Transactions on Control Systems Technology*, 17(5):1105–1118, 2009.

[43] S. Karaman, M. R. Walter, A. Perez, E. Frazzoli, and S. Teller. Anytime motion planning using the rrt*. In *2011 IEEE International Conference on Robotics and Automation*, pages 1478–1483, 2011.

[44] J. Ji, A. Khajepour, W. W. Melek, and Y. Huang. Path planning and tracking for vehicle collision avoidance based on model predictive control with multiconstraints. *IEEE Transactions on Vehicular Technology*, 66(2):952–964, 2017.

[45] Hong Wang, Yanjun Huang, Amir Khajepour, Yubiao Zhang, Yadollah Rasekhipour, and Dongpu Cao. Crash mitigation in motion planning for autonomous vehicles. *IEEE transactions on intelligent transportation systems*, 20(9):3313–3323, 2019.

[46] Min Park, Jae Jeon, and Min Cheol Lee. Obstacle avoidance for mobile robots using artificial potential field approach with simulated annealing. volume 3, pages 1530 – 1535 vol.3, 02 2001.

[47] Steven Byrne, Wasif Naeem, and Stuart Ferguson. Improved apf strategies for dual-

arm local motion planning. *Transactions of the Institute of Measurement and Control*, 37(1):73–90, 2015.

[48] Mohammad Jaradat, Mohammad Garibeh, and Eyad Feilat. Autonomous mobile robot dynamic motion planning using hybrid fuzzy potential field. *Soft Computing*, 16:153–164, 06 2012.

[49] Yiqi Gao, Theresa Lin, Francesco Borrelli, Eric Tseng, and Davor Hrovat. Predictive control of autonomous ground vehicles with obstacle avoidance on slippery roads. 01 2010.

[50] Andrew Gray, Mohammad Ali, Yiqi Gao, J Hedrick, and Francesco Borrelli. Semi-autonomous vehicle control for road departure and obstacle avoidance. *IFAC control of transportation systems*, pages 1–6, 2012.

[51] A. Carvalho, Y. Gao, S. Lefevre, and F. Borrelli. Stochastic predictive control of autonomous vehicles in uncertain environments. 2014.

[52] A. Gray, Y. Gao, T. Lin, J. K. Hedrick, and F. Borrelli. Stochastic predictive control for semi-autonomous vehicles with an uncertain driver model. In *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*, pages 2329–2334, 2013.

[53] G. D. Forney. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, 1973.

[54] David Q. Mayne. Model predictive control: Recent developments and future promise. *Automatica*, 50(12):2967–2986, 2014.

[55] A. Mesbah. Stochastic model predictive control: An overview and perspectives for future research. *IEEE Control Systems Magazine*, 36(6):30–44, 2016.

[56] E. Romera, L. M. Bergasa, and R. Arroyo. Need data for driver behaviour analysis? presenting the public uah-driveset. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pages 387–392, 2016.

[57] Erling Bernhard Andersen. Asymptotic properties of conditional maximum-likelihood estimators. *Journal of the Royal Statistical Society. Series B (Methodological)*, 32(2):283–301, 1970.

[58] Nikolai Vladimirovich Krylov and Aleksandr Adol'fovich Yushkevich. Markov random sets. *Trudy Moskovskogo Matematicheskogo Obshchestva*, 13:114–135, 1965.

[59] A. Eidehall and L. Petersson. Statistical threat assessment for general road scenes using monte carlo sampling. *IEEE Transactions on Intelligent Transportation Systems*, 9(1):137–147, 2008.

[60] Sean Summers, Maryam Kamgarpour, John Lygeros, and Claire Tomlin. A stochastic reach-avoid problem with random obstacles. In *Proceedings of the 14th International Conference on Hybrid Systems: Computation and Control*, HSCC '11, page 251–260, New York, NY, USA, 2011. Association for Computing Machinery.

[61] Damoon Soudbakhsh and Azim Eskandarian. *Vehicle Lateral and Steering Control*, pages 209–232. Springer London, London, 2012.

[62] Tim Brüdigam, Fulvio di Luzio, Lucia Pallottino, Dirk Wollherr, and Marion Leibold. Grid-based stochastic model predictive control for trajectory planning in uncertain environments, 2020.

[63] Reinhard Klette and Azriel Rosenfeld. Chapter 13 - hulls and diagrams. In Reinhard Klette and Azriel Rosenfeld, editors, *Digital Geometry*, The Morgan Kaufmann Series in Computer Graphics, pages 427–454. Morgan Kaufmann, San Francisco, 2004.

[64] Matthias Althoff. An introduction to cora 2015. In Goran Frehse and Matthias Althoff, editors, *ARCH14-15. 1st and 2nd International Workshop on Applied veRification for Continuous and Hybrid Systems*, volume 34 of *EPiC Series in Computing*, pages 120–151. EasyChair, 2015.