NORTHWESTERN UNIVERSITY

Safe and Secure Design of Connected and Autonomous Vehicles

A DISSERTATION

SUBMITTED TO THE GRADUATE SCHOOL
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

for the degree

DOCTOR OF PHILOSOPHY

Field of Computer Engineering

By

Xiangguo Liu

EVANSTON, ILLINOIS

June 2023

ABSTRACT

Machine learning-based techniques have shown great promises in perception, prediction, planning, and general decision-making for improving task performance of autonomous driving. Connectivity technology has also presented great potentials in improving the safety and efficiency of transportation systems by providing information beyond the perception and prediction capabilities of individual vehicles. However, a number of challenges significantly impede their applications in realizing connected and autonomous vehicles. These challenges include (1) increasing difficulty in formally analyzing the behavior of neural network-based planners for ensuring system safety, (2) preventing over-conservative planning in dense and highly interactive traffic environments, (3) increasing complexity in analyzing system behavior and quantifying uncertainty in mixed traffic scenarios, including human-driven and autonomous vehicles, and connected and non-connected vehicles, (4) difficulty in accurately predicting surrounding vehicles' behaviors and trajectories, and (5) defending possible cyber and physical attacks on connected vehicle applications.

To overcome these challenges in connected and autonomous vehicles, we propose several safety-assured planning schemes and a trust framework in this thesis. In the first work, we propose a hierarchical neural network based planner that analyzes the underlying physical scenarios of the system and learns a system-level behavior planning scheme with multiple scenario-specific motion-planning strategies. We then develop an efficient verification method that incorporates overapproximation of the system state reachable set and novel partition and union techniques for formally ensuring system safety under our physics-aware planner. With theoretical analysis, we show that considering the different physical scenarios and building a hierarchical planner based on such analysis may improve system safety and verifiability.

In the second work, we propose a safety-driven interactive planning framework in mixed traffic

scenarios. We identify the driving behavior of surrounding non-connected vehicles and assess their aggressiveness, incorporate the information shared by surrounding connected vehicles, and then adapt the planned trajectory for the ego vehicle accordingly in an interactive manner. The ego vehicle can proceed to execute driving tasks if a safe evasion trajectory exists even in the predicted worst case; otherwise, it can perform a less preferred behavior or follow the pre-computed evasion trajectory.

Thirdly, we propose a novel speculative planning framework based on a prediction-planning interface that quantifies both the behavior-level and trajectory-level uncertainties of surrounding vehicles. Our framework leverages recent prediction algorithms that can provide one or more possible behaviors and trajectories of the surrounding vehicles with probability estimation. It adapts those predictions based on the latest system states and traffic environment, and conducts planning to maximize the expected reward of the ego vehicle by considering the probabilistic predictions of all scenarios and ensure system safety by ruling out actions that may be unsafe in worst case.

For these planner designs, we demonstrate the effectiveness of our approaches and their advantages over other baselines in practical case studies of unprotected left turn, highway merging or lane changing, through extensive simulations with diverse and comprehensive experimental settings, or in real-world scenarios collected by an autonomous vehicle company.

Finally, we propose an efficient dual cyber-physical blockchain framework to build trust and secure communication for CV applications. Our approach incorporates blockchain technology and physical sensing capabilities of vehicles to quickly react to attacks in a large-scale vehicular network, with low resource overhead. We explore the application of our framework to three CV applications, i.e., highway merging, intelligent intersection management, and traffic network with route choices. Simulation results demonstrate the effectiveness of our blockchain-based framework in defending against spoofing attacks, bad mouthing attacks, and Sybil and voting attacks.

ACKNOWLEDGEMENTS

First and foremost, I want to thank my advisor, Prof. Qi Zhu. He has always been supportive and patient. For these challenging and exciting projects, it is him spends so much time and effort mentoring me, discussing with me, and encouraging me. His help not only brings our work to top-tier conferences and journals, but also, more importantly, shapes my thinking and will influence my future career and life. He is always passionate and strives for excellence in his work, and is absolutely a role model for me.

Before I started my PhD journey, I spent about one year under the supervision of Prof. Neda Masoud at the University of Michigan, Ann Arbor. I want to thank her for introducing me to the research area of connected and autonomous vehicles. In that year, I was full of curiosity about everything in the United States. I felt the boredom and excitement of scientific research, as well as the warmth of family. I always miss my time in Ann Arbor.

I want to thank Prof. Ermin Wei and Prof. Jie Gu for being on my prospectus and thesis committee. Their suggestions and comments help me to improve the work and motivate many interesting directions.

During all these years, I have worked and collaborated with brilliant researchers from other groups. They are but not limited to Prof. Chung-Wei Lin, Prof. Qi Alfred Chen, Prof. Nael Abu-Ghazaleh, Prof. Wenchao Li, Prof. Xin Chen, Prof. Chao Huang, Prof. Anahita Khojandi, Bowen Zheng, Baiting Luo, Takami Sato, Guangchen Zhao, Dave Liang, Pin-Chun Chen, Ahmed Abdo, Kevin Kai-Chun Chang, Jiameng Fan, Weichao Zhou, Liren Shan, Yiqi Lou and many others. I want to thank them for those great discussions and suggestions.

With Prof. Zhu's support, I am fortunate to have three internship experiences. I want to thank all my mentors and colleagues at Toyota InfoTechnology Center, XMotors.ai, and Meta Platforms.

Without their help, I cannot learn and work smoothly in these summers. I am also very happy and feel fortunate to meet many new friends in these companies.

I want to thank all my friends and labmates in Northwestern and Michigan. They are but not limited to Hengyi Liang, Shuyue Lan, Zhilu Wang, Shichao Xu, Ruochen Jiao, Yixuan Wang, Lixu Wang, Payal Mohapatra, Anthony Goeckner, Yiyang Wang, Ethan Zhang, Ye Li, Zhibin Chen and many others. Working and playing with them is full of joy.

Finally I want to thank my families and my fiancée Xuefei. Your company and support helped me through the toughest days. I feel energetic and motivated to welcome the next day after communicating with you, hearing warm words from you. I feel fortunate to have you. I cannot make this achievement without your support along this long journey.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION AND BACKGROUND

Neural network-based machine learning techniques have been increasingly leveraged in autonomous driving for perception, prediction, planning, control, etc. In particular, neural networks may greatly improve performance [1] and efficiency for planning and general decision making in various traffic scenarios, such as unprotected left turn, highway merging and lane changing. Moreover, compared with traditional model-based approaches, they can save the time and effort of explicitly modeling systems with complex dynamics and significant uncertainties.

At the same time, connected vehicle (CV) applications are expected to revolutionize traditional transportation system. In a connectivity-enhanced transportation system, vehicles can communicate with each other and/or surrounding infrastructures via dedicated short range communication (DSRC) [2] or cellular vehicle-to-everything (C-V2X) [3]. These communications share important information of vehicles' current states (e.g., location, speed, acceleration) and future intentions (e.g., planned actions and trajectories) that go well beyond the perception and prediction capabilities of individual vehicles, e.g., sharing information that are out of sight of the ego vehicle or intentions that cannot be accurately predicted. Beyond that, vehicles can negotiate and coordinate in distributed manner [4], or follow instructions from a central unit [5] to further optimize the system. This makes connectivity a great complementary to both autonomous vehicles and human-driven vehicles.

However, there are some safety and security challenges that can significantly impede the transition to an intelligent transportation system, even cause accidents and disrupt traffic flow. Thus there is an urgent need to address these challenges in connected and autonomous vehicles.

In this chapter, we start with safety and security challenges in connected and autonomous vehicles, then discuss related works, present the overview of our approaches and finally summarize the contributions.

1.1 Challenges

Recently, a variety of neural network-based planner designs, including hierarchical planners, have been developed for various applications due to their strengths in improving system performance and reducing accident rate in average [6]–[8]. However, a major challenge for the neural network-based planners is to ensure system safety, especially in near-accident scenarios [7], [9], such as unprotected left turn and highway merging in autonomous driving. In those scenarios, with only minor changes in environment states, dramatically different behaviors may need to be performed to avoid accidents, which is difficult for both humans and autonomous systems to handle. The state-of-the-art verification methods have limited efficiency and accuracy for those complex and high-dimensional systems.

Although safety is the most important metric to evaluate planners, performance should not be sacrificed too much. There is a common safety-efficiency dilemma in autonomous driving, especially in those highly interactive and dense traffic scenarios [7], [10]–[14]. Some planners have larger buffer space for safety [15] to handle uncertainties from surrounding vehicles and the environment, however can be overly conservative and inefficient; while other planners put more emphasis on efficiency and task success rate, but risk safety.

It could be even more challenging during the transition period to a fully-automated transportation system, when human-driven and autonomous vehicles need to share the transportation network and interact with each other. In mixed traffic, human drivers have different driving patterns, which can even change over time [16], while autonomous vehicles designed by different companies can

have varied driving strategies for similar scenarios [17]. With increasing difficulty in accurately predicting other vehicles' intentions, the interaction process could be either inefficient or unsafe.

While connectivity technology has the potential to greatly mitigate the challenges in predicting the intention and future trajectory of surrounding vehicles [18], it is expected that there will be a long transition period before the full deployment of connected vehicles and traffic infrastructures. Recent progress has been made to ensure system safety when all vehicles are of the same type [19], [20], or connectivity is not enabled [21]–[25], and only some works [21], [24], [25] can generalize to systems with neural network-based components. It is an open challenge to model the behavior of a general system with all kinds of vehicles, not to mention providing safety guarantee while not overly sacrificing efficiency.

All these make it critically important in autonomous driving to accurately predict the surrounding vehicles' behavior *and* effectively leverage the prediction results in planning. In the literature, some recent prediction algorithms can provide one or more most possible trajectories of a surrounding vehicle, albeit does not emphasize their behavior-level difference [26]–[29]. Some planning algorithms are designed to prevent traffic accidents for the most possible predicted trajectory, but ignore other possibilities and cannot guarantee safety in those cases [30], [31]. There are other planning strategies that consider all possible predicted trajectories of a surrounding vehicle. However, it would be over-conservative if the most cautious action is always selected in considering all possibilities [32]. Directly taking a 'weighted' action across all possibilities can also be risky [7], [10] – for example, when the traffic signal in an intersection just turns yellow, it may be safe for a vehicle to maintain its velocity and pass the intersection before the traffic signal turns red *or* decelerate and stop before the intersection, but unsafe for it to take a weighted action like hesitating and entering the intersection at a low speed.

As for the security challenges, cyber and physical attacks targeting connected vehicle applica-

tions can severely impact system performance [33], [34], cause accidents or disrupt traffic flow. For example, an intelligent intersection management system can experience deadlock if messages have a long transmission delay or get lost under the denial of service attack [5], [35]. Merging in the highway will be more prone to accidents if vehicles get wrong position data of surrounding vehicles under spoofing and false message attacks [36]. As the impact of cyber threats [37] on CV operations can be so destructive, it is essential to develop security solutions against them.

A central issue in CV security is to build trust among vehicles. The authors in [38] reviewed methods to evaluate message trustworthiness in the vehicular network, including entity-oriented, data-centric, and collaborative trust models. The entity-oriented trust models evaluate messages' trustworthiness based on the trustworthiness of their senders. A Certification Authority is often leveraged to record vehicles' behavior and provide estimations of trust; otherwise, a vehicle needs to collect information and make evaluation by itself. The data-centric trust models evaluate the content of messages to estimate the trust for them. In this category, Bayesian inference, and Dempster-Shafer theory are popular methods to estimate the plausibility and trustworthiness of messages, however they may result in false positives for detecting valid messages [39]. The collaborative trust models are based on integrating trust estimates from other peer vehicles, which could be time-consuming and attacked by malicious players. The Security Credential Management System (SCMS) is a proof-of-concept message security solution that is supported and developed by USDOT. Instead of evaluating message trustworthiness by each vehicle, message senders with credentials can be trusted in the system. However, since a certified vehicle can be attacked later, it is challenging for certificate authorities to track and update vehicles' status quickly. Thus, SCMS on its own does not prevent application level attacks [36].

Blockchain [40] technology has natural strength in recording transactions/events and reaching a secure consensus among all users. It provides a promising direction for building trust in CV

applications, as shown in [41]–[45]. However, the required secure consensus operations in those earlier works introduce high overhead that makes it difficult for applying them in practical CV applications.

1.2 Related Works

In this section, we summarize related works. We first introduce traditional and machine learning-based planner designs in autonomous driving, then present the state-of-the-art approaches to ensure system safety, discuss the prediction-planning interface design, and finally present recent studies in applying blockchain to transportation systems.

1.2.1 Planner Designs in Autonomous Driving

There are a number of varied planner designs, including classical rule-based [46], optimization-based [47] and game theory-based [48] planners, as well as emerging neural network-based planners. [49] reviews the planning and control techniques in an urban environment, [50] reviews motion planning techniques for highway driving, [51], [52] emphasize the real-time performance of planning techniques, and [53]–[55] focus on the performance of the proposed methods. [56], [57] attempt to balance computational efficiency and solution quality. [58], [59] consider communication delay and reaction time in designing motion planners, and [60]–[62] address the uncertainty in the driving behavior of surrounding vehicles. [30] proposes a dynamic lane changing planner that updates its reference trajectory periodically. If necessary, it can plan a trajectory back to the original lane to eliminate collision. Similarly, [63] can plan a safe evasion trajectory. However, [30], [63] assume that the leading and following vehicles in the target lane will remain their velocities when the ego vehicle changes lanes, which may not hold due to the fluctuation of traffic stream and the interactions between vehicles. [64] leverages dynamic programming to compute decisions

for safely changing lanes, but the computation efficiency will be significantly reduced with higher discretization precision.

Both search-based and sampling-based methods are developed in a discretized space (either state space or action space), thus the computation efficiency will be significantly influenced if we increase the discretization precision. Besides this point, search-based methods are limited in spatiotemporal planning while sampling-based methods may lead to jerky paths. Optimization-based methods, e.g., Model Predictive Control (MPC) [65]–[68], can handle static and dynamic obstacles as constraints and output a smooth trajectory intrinsically. However, these constraints from obstacles and road boundaries are usually non-convex. It considerably increases the computation time for the optimization problem, which is hard to be in real-time. Recent proposed algorithms, Differential Dynamic Programming (DDP) [69], Iterative Linear Quadratic Regulator (ILQR) [70], and most recently, CILQR [71], [72], can solve real-time optimization with non-convex constraints within 200 milliseconds by transforming the hard constraints into cost terms.

To improve performance and task success rate, especially in dense traffic, prior works [73]–[78] have emphasized the importance of modeling inter-vehicle interactions. [73] proposes a game theory-based lane changing model for connected vehicles, and calibrates different parameters in the utility functions for mandatory and discretionary lane changing scenarios. [79] assumes that all vehicles are connected and cooperative, which is not the case during the transition period. [80] leverages partially observable Markov decision process to model the level of cooperation of other drivers, and incorporates this belief into reinforcement learning-based planner for higher merging success rate. [81] uses recurrent neural network (RNN) to model interaction between vehicles, and then incorporates the prediction results in safety constraints for a MPC planner. Besides RNNs, transformers have been regarded as an alternative architecture for modeling time-series trajectories and have achieved superior performance [82]. [74] obtains a probability distribution of different

intentions for each surrounding vehicle via Bayesian estimation, and then plans trajectories of the ego vehicle considering its uncertain interactions with surrounding vehicles. [83] leverages game theory to model cooperative lane changing scenarios, in which vehicles aim to minimize the joint cost function. [16] defines a parameter called aggressiveness to represent driver's personality, estimates it in real time and incorporates it in the utility function. Different extent of aggressiveness represents a different preference for travel time, headway and etc. It models vehicle's interaction in lane changing process in a Stackelberg game. Machine learning methods have natural strengths in interaction modeling and prediction [1]. [84] makes discrete behavior decisions for mandatory lane changing based on Bayes classifier and decision trees. [85] leverages Partially Observable Markov Decision Process (POMDP) to include surrounding vehicles in the state space, thus planning interactive behavior. [86] acquires a strategic level k planner for merging in dense traffic by reinforcement learning and iterative reasoning.

A number of studies focus on the interaction between human drivers and robot drivers [87]. [88] assumes that humans presume robots to behave rationally. As such, robots can predict human behavior and take advantage of it in their motion planning. [89] uses microscopic traffic simulations to show that the average travel time decreases by a factor of 4 if altruistic AVs are introduced to traffic streams. [90] models surrounding vehicles based on level-k game theory. Optimal decisions for an AV at roundabouts are computed after estimating the driver type of the opponent vehicle. [91] and [92] model the interaction between AVs and human drivers using dynamic game theory. [93] utilizes inverse reinforcement learning to model human drivers, assuming they are perfectly rational. AVs can thus purposefully elicit desired changes in the human state. [94] claims that by interacting with humans, robots can learn the humans' internal states and thus optimize their operations. However, this information might lead to the robot taking advantage of the humans.

Machine learning-based techniques are increasingly popular in planning and decision-making

for autonomous driving [81], [95]–[97], for their potential in improving average system performance under complex scenarios. Some of those learn a single neural network for planning via reinforcement learning [98], imitation learning [99], supervised learning [100], etc., while others employ a hierarchical planner design [7], [101], which usually consists of low-level planners for different modes and a high-level planner that is responsible for selecting the mode. The work in [102] proposes a concept of social perception, which inferences surrounding environment from other vehicles' reactions. It then leverages inverse reinforcement learning (IRL) to acquire cost function of human driving, and uses Markov Decision Process (MDP) to get probabilistically optimal solutions. [103] formulates the lane changing planning problem as a partially observable Markov Decision Process (POMDP), in which the cooperativeness of other traffic participants is an unobservable state. It predicts future actions of human cars via logistic regression classifier, and solves the POMDP by Monta-Carlo Tree Search. [104] leverages reinforcement learning and considers the possible action of aborting lane changing and returning back to original lane. [105], [106] are also based on reinforcement learning. [107] develops hierarchical reinforcement learning-based planners to address both the timing and the specific maneuver for lane changing. [7] proposes a hierarchical reinforcement and imitation learning (H-REIL) approach that consists of low-level policies learned by imitation learning under different driving modes and a high-level policy learned by reinforcement learning for switching between driving modes.

Although these methods demonstrate great performance improvement, it is still quite challenging to verify the safety for learning-enabled systems. System efficiency is also restricted by uncertainties from perception [108] and prediction results of individual vehicles. And connectivity can enhance the transportation system by reducing such uncertainties [109]–[113]. For instance, the work in [19] proposes an intersection management scheme, in which the central manager assigns arriving speed and arriving time to vehicles. Vehicles track optimal trajectories according

to assignments under proportional–integral–derivative (PID) controllers, which can compensate bounded model mismatch and external disturbances. It assumes that all vehicles are connected and autonomous. The approach in [20] leverages Dynamic Bayesian Networks (DBNs) to model vehicle state evolution. The central manager can send out warning messages to vehicles for collision avoidance. It assumes that all vehicles are connected and human-driven, and collision rate depends on velocity and driver reaction time. The work in [114] assumes that all vehicles are connected, and leverages deep reinforcement learning (DRL) for behavior-level decision making in lane changing. In particular, it gets performance improvement by incorporating traffic status in the downstream with vehicle-to-vehicle communication.

There are several works developed for mixed traffic [115]–[120] of connected and non-connected vehicles. The work in [121] presents an RL-based multi-agent longitudinal planner for connected and autonomous vehicles, which adjusts speeds in upstream traffic to mitigate traffic shock-waves downstream. The results suggest that even for a penetration rate of 10%, connected and autonomous vehicles can significantly mitigate bottlenecks in highway traffic. The approach in [122] leverages RL for trajectory recommendation to the connected vehicles in highway merging scenarios. It assumes that not all vehicles are connected and uses camera in roadside for data fusion, in order to map all vehicles. The work in [123] proposes an RL-based method for connected and autonomous vehicles to decide actions such as whether to change lane or keep lane based on the observation and shared information from neighbors. The system is modeled by hybrid partially observable Markov Decision Process (HPOMDP) as not all vehicles are connected. However, it does not explicitly model inter-vehicle interaction, and the safety highly depends on accurate modeling of the surrounding vehicles, especially non-connected vehicles.

1.2.2 Safety-assured Approaches

Even though safety improvement is often considered and demonstrated empirically through experiments in those works [7], [124]–[130], formal system safety verification remains a challenging problem. Reachability analysis and certification are popular formal techniques for verifying system safety, with various recent methods for LE-CPSs [131]–[140]. However, these methods are usually applied to relatively simple scenarios, such as adaptive cruise control and emergency braking [131], [141]. Regarding the verification of neural network controlled systems in the literature [132]–[134], [142], [143], a single planner is typically considered. Verification for a hierarchical planner and hybrid system is not well studied. Moreover, these methods are challenging to scale to complex scenarios due to the disturbances and uncertainties from both the physical world and neural networks [144] and often result in conservative conclusions.

There are several works that try to provide formal safety guarantees. For instance, [145] analyzes the distance between vehicles to ensure safety in lane changing scenarios, however the distance is derived only based on braking behavior. Without considering steering, the calculated safe distance is often over-conservative and hard to meet in practice, especially in dense traffic. Moreover, it does not explicitly analyze the intention of the following vehicle in the target lane. And such limitation also exists in [22], [146]. The work in [147] analyzes the minimum critical distance around surrounding vehicles by considering both braking and steering behavior, and assumes that the worst case occurs when the leading vehicle in the target lane has a full stop suddenly or the following vehicle in the target lane remains its acceleration to close the gap. However, [147] neglects the fact that the ego vehicle can steer and brake at the same time to avoid collision, and that the worst case for the leading and the following vehicles can occur at the same time. Moreover, it does not consider inter-vehicle interactions.

There is an urgent need to prevent overly conservative design while also provide safety guaran-

tees when considering environment uncertainty and complex inter-vehicle interactions [24], [25], [32], [148], [149]. The work in [21] proposes a concept of legal safety, which means that autonomous vehicles will not be the cause of accidents and there is no collision if surrounding vehicles obey traffic rules. This is realized by proving the existence of the fail-safe trajectory under the planner all the time. Similarly, the concept of responsibility-sensitive safety is proposed in [22], which assumes that other participants behave according to common-sense rules and defines appropriate responses of autonomous vehicles in near-accident scenarios. However, safety can be compromised if other vehicles' behaviors violate the assumptions. The works in [23], [32] develop a non-conservatively defensive driving strategy, which leverages sampling-based or optimization-based methods. Planned trajectory is executed after the safety evaluation. The work in [150] presents a MPC-based planner for trajectory tracking, which ensures safety through maintaining the availability of a collision-free escape maneuver.

1.2.3 Prediction-planning Interface

With the wide adoption of machine learning-based techniques, the performance of trajectory prediction [151], [152] has significantly improved over the last several years. Most works predict the trajectories of traffic participants and evaluate their accuracy [27], [153]–[155]. There are also some works that consider both high-level behaviors and low-level trajectories in the prediction algorithms. For instance, [156] proposes an integrated lane change prediction model to predict the lane change decisions and lane change trajectories. [157] develops a domain generalization method for prediction in unseen scenarios, and mainly works on behavior prediction. [158] performs both standard forecasting and the novel task of conditional forecasting, which reasons about how all agents will likely respond to the goal of a controlled agent. It points out that goal/intent-conditioned trajectory forecasting can improve joint-agent and per-agent predictions, compared to

unconditional forecast. [159] proposes a framework that first explicitly predicts the distribution of an agent's endpoint over a discretized goal set and then completes the trajectories conditioned on the selected goal points. The goal set is designed with domain knowledge. [12], [153] propose an architecture to encode driving behaviors such as lateral intetion and aggressiveness into latent space, which can enhance the explainability and robustness of trajectory prediction modules.

Various planner designs are reviewed in [49], [50], [160], [161]. These planners can be classified according to the inputs and assumptions. Most of these planners such as [30], [71] take the latest system states and environmental information as inputs, e.g., position and velocity of surrounding vehicles, road geometry, and so on. Based on the position and velocity of surrounding vehicles, it is usually assumed that vehicles keep the same velocity in the next few seconds, thus the trajectory is acquired. Thus it is appropriate and convenient to take more accurately predicted trajectories as inputs to these planners [30], [47]. There are also many planners with embedding models for surrounding vehicles, which describe the reward function with a set of parameters [16], [48] or neural networks [162], [163]. These methods can model the interaction between the ego vehicle and surrounding vehicles, and the future trajectories of these agents are derived at the same time.

Significant progress has been made recently to reduce the uncertainty and safety risks during interactions among vehicles. Some works propose that it is safer and more effective to conduct motion planning with pre-determined or predicted behavior of surrounding vehicles [4], [7], [10]. Confidence-based methods [148], [164] are also promising in human-robot interaction, where robots are designed to use confidence-aware game theoretic models of human behavior when assessing the safety [164]. Confidence gets updated after comparing real human behavior and its predictions, and safety is ensured by switching to a safe planner when necessary. [31] integrates a probabilistic prediction model into the design of reachability-based safety controllers to achieve

more efficient two-car collision avoidance. However, it cannot ensure safety because it only reacts to the most possible predicted trajectories of surrounding vehicles. [165] introduces a Bayesian Long Short-term Memory (BLSTM) model to predict the probability distribution of surrounding vehicles' positions, which are used to estimate dynamic conflict risks. MPC is then incorporated to navigate vehicles through safe paths with the least predicted conflict risk. However, it only considers trajectory-level uncertainty, and it could be over-conservative because collision risk is the only term for optimization.

Some prediction methods provide more information to quantify the uncertainty, which can improve the performance of planners in uncertain and dynamic environments [129], [130]. For instance, [166] uses Gaussian Mixture Model to describe the predicted intentions, detailed waypoints, and corresponding covariance for each waypoint. [26], [27] work on multimodal trajectory predictions, and produce every trajectory's probability by a prediction network. [167] predicts multimodal potential trajectories with corresponding probabilities based on a set of anchors. [28], [29] first predict possible goals for surrounding vehicles and then generate multiple detailed future trajectories. [168] can learn effective representations for detecting potential anomalous or unsafe predictions of surrounding vehicles.

1.2.4 Blockchain in Transportation

Blockchain is literally a chain of blocks (called a ledger), in which a block stores a set of transactions and the hash value of the previous block. Transactions are public to every user, and thus blocks can be verified by every user. Blockchain is updated based on consensus algorithms to guarantee that the state of the ledger is identical at all nodes. The most well-known blockchain, bitcoin, is based on proof of work. It assumes that a group of miners will build up new blocks by consuming CPU resources to find the right cryptographic nonce. Once a new block is proposed

by a miner, other miners will verify it and start working on the next block. Because the block generation process is not deterministic, a hacker may successfully build a new block with a slight possibility. In this way, a block is considered to be confirmed if it is followed by several other blocks. For bitcoin, the block generating period is about 10 minutes, and the confirmation time can be 1 hour. Moreover, with proof of work, miners may compete to earn profits and dominate the blockchain with more CPU resources, which could lead to huge computation cost and energy consumption. Several alternative consensus mechanisms are proposed to prevent exhausting computation resources [40], [169].

Algorand is a recently proposed blockchain design that enables efficient block confirmation time. It is based on proof of stake, where users do not need to exhaust computation resources. Instead, users need to deposit money as the stake. The more money a user owns, the higher chance the user can be selected as a block proposer or verifier. Its security is guaranteed theoretically as long as the majority of stake is owned by honest users [170]. Furthermore, its scalability is demonstrated with simulations of up to 500,000 users, in which transaction confirmation time can be shortened to be within one minute [171], making it feasible for some systems in real-time. More specifically, for each round r, a leader l^r and a small set of verifiers SV^r that are randomly selected and publicized by cryptographic self-selection participate. The leader will build and propagate a new block, and then selected verifiers will reach the Byzantine agreement on it within a finite number of steps. Upon termination, the agreed block B^r is added to the ledger. Since the block has already reached consensus and has been confirmed, the round period is exactly the transaction confirmation time.

Blockchain has been leveraged by many researchers to advance applications in transportation systems in the literature. [172] presents a blockchain-based negotiation process to select the most convenient electric vehicle (EV) charging station. [173] leverages blockchain to record EV charg-

ing activities, which enables information sharing while securing sensitive user information. [174] presents a seven-layer conceptual model for blockchain in transportation and discusses ride sharing in the application layer.

There are also several works that consider using blockchain in vehicular networks. [41] reviews different models for calculating reputations of vehicles, discusses challenges of previously centralized and distributed methods, and indicates new directions on blockchain and fog computing. [42] proposes to secure vehicular communication by recording each message in blockchain, which is not scalable. Its proof of work scheme also leads to heavy computation burden. [43] proposes a blockchain-based trust management scheme for vehicular networks, in which road side units (RSUs) update and maintain the blockchain. Its consensus is partially based on proof of work, which still leads to high computation cost for every RSU, and its security in one region cannot be well protected once the RSU is compromised. [175] leverages Bitcoin-NG and hierarchical design to reduce transmission latency of blocks and increases the throughput capacity. However, this hierarchical design assumes that RSUs will maintain a higher-layer blockchain, which processes all information from lower-layer blockchains. It also has high computation resource demand and is not resilient to attacks. [176] proposes a similar idea that may also suffer from the failure of one node. [177] attempts to mitigate the resource demands by proposing concepts of sub-blockchains for different node types, e.g., vehicles, RSUs, and stations. However, it lacks implementation details such as communication across different sub-blockchains.

1.3 Overview of Our Approaches

In this section, we briefly present our approaches to address the challenges discussed above.

1.3.1 Safety-assured Hierarchical Neural Network-based Planner

We first observe that for systems that may evolve into different physical scenarios[1] under a single neural network-based planner, it is often difficult to verify their safety or the planner is indeed unsafe. And we conduct theoretical analysis to show the reason. Based on such observation and the fact that many safety-critical systems may evolve into multiple different physical scenarios and thus require dramatically different behaviors to ensure their safety and improve performance, we propose a *hierarchical neural network-based planner* that consists of a system-level behavior planner and multiple scenario-specific motion planners. We then develop an *efficient verification method* that incorporates novel partition and union techniques and an approach for overapproximating system state reachable set to formally verify the system safety under our hierarchical planner.

1.3.2 Safety-driven Interactive Neural Network-based Planner

We propose a *safety-driven interactive planning framework with neural network-based planners* in dense traffic scenarios. In the example of lane changing, we have two neural network planners for longitudinal and lateral motions, respectively. The two planners take motion status of surrounding vehicles and the ego vehicle as input, and output planned accelerations for the ego vehicle. In order to enhance safety while improving performance, the ego vehicle can make lane changing attempt under the neural network planners only if it has a safe evasion trajectory even in the predicted worst case; and if such safe evasion trajectory does not exist, the planned trajectory from neural networks will be adjusted according to safety analysis of all involved vehicles. To prevent an overly conservative planner design, we leverage another neural network to assess the aggressiveness of the following vehicle in the target lane and predict whether it is willing to let the ego vehicle complete the lane change. In the case that the following vehicle is cautious (intuitively meaning that it is

[1] For instance, in unprotected left turn, a vehicle may make the turn, yield or stop based on the situation.

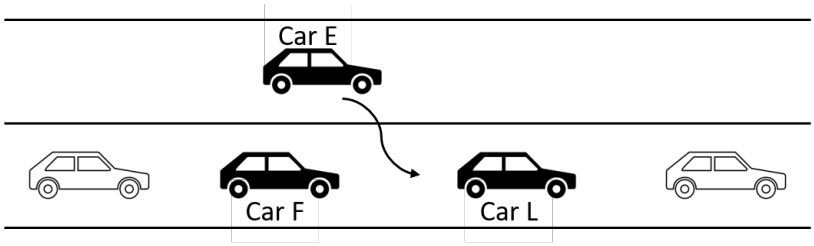

Figure 1.1: Lane changing scenario. The ego vehicle E, an autonomous vehicle, intends to change lanes and insert itself downstream of vehicle F, a human-driven or an autonomous vehicle in the target lane.

willing to let the ego vehicle get in front of it), the ego vehicle can complete the lane changing confidently; otherwise, the following vehicle is aggressive and the ego vehicle needs to be more conservative.

Fig. 2.1 shows the specific scenario we consider. The ego vehicle E, an autonomous vehicle, intends to change lanes and insert itself downstream of vehicle F, a human-driven or an autonomous vehicle in the target lane. We assume that the worst case occurs when the leading vehicle L in the target lane decelerates abruptly and at the same time the following vehicle F has the highest acceleration under its predicted cautious/aggressive mode. A safe evasion trajectory exists if the ego vehicle can take that trajectory and return to the original lane without colliding with other vehicles.

1.3.3 Safety-assured Speculative Planning with Adaptive Prediction

This work focuses on addressing complex driving scenarios where the surrounding vehicles' intentions and planned trajectories have multiple possibilities in prediction. For example, we consider the lane changing in a multi-lane highway as a representative application (while our proposed approach can be extended to other driving tasks and scenarios). As shown in Fig. 1.2, the system includes an ego vehicle E going straight and a surrounding vehicle S indicating a right turn to change lanes. However, the intention of the surrounding vehicle is ambiguous. It can change lane

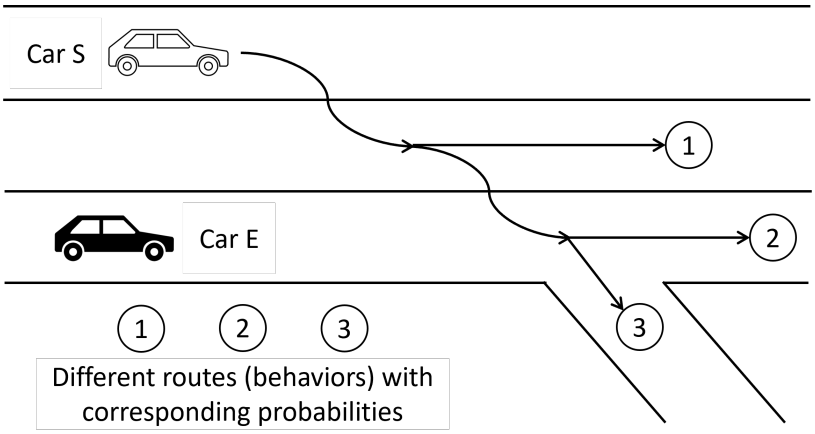

Figure 1.2: Representative case study: In a multi-lane highway, an ego vehicle E goes straight and a surrounding vehicle S indicates a right turn to change lanes. However, the intention of S is ambiguous. It can change lane once and follow the route 1, or change lanes twice and follow the route 2, or exit the highway after lane changing and follow the route 3.

once and follow the route 1, or change lanes twice and follow the route 2, or exit the highway after lane changing and follow the route 3. Each possible route may be associated with a probability, i.e., p_1, p_2 and p_3. Moreover, besides the behavior-level uncertainty on taking which route, the exact parameters for defining each trajectory are also uncertain.

We propose a safety-assured speculative planning framework with adaptive prediction. The framework leverages prediction algorithms that can provide one or more possible behaviors and future trajectories of surrounding vehicles [28], [154], [159]. During planning, the framework considers all those possibilities and first rules out the actions that may be unsafe in the worst case. Within the remaining actions, it selects the one that maximizes the expected reward (representing system performance) of all possible intentions and trajectories in prediction results, with larger weights assigned to the more likely ones. We consider such planning *speculative* because the prediction results are likely to change over time. Thus, our framework also checks the updated prediction results over time and *adapts* them based on system states and traffic environment, to

filter out those impossible behaviors and trajectories of the surrounding vehicles for more effective planning. Moreover, we incorporate the prediction of the aggressiveness level of the surrounding vehicles into our prediction-planning interface, which may further reduce prediction uncertainty and improve system performance.

1.3.4 Securing Connected Vehicle Applications with Blockchain

Our framework leverages Algorand, combining its protocol design with our ideas of designing dual cyber-physical blockchains at different time scales, using vehicles' physical sensing capabilities for verifying messages, and introducing the concept of proof-of-travel credits in stake computation. Moreover, to scale Algorand to CV applications in a large traffic network, we leverage the sharding technique [40] for record transferring and sharing between blockchains in different regions. By choosing an appropriate region size, our framework design can have short round latency and low overhead, while preventing vehicles from transferring records frequently. In addition, our proposed summary step further reduces the overhead for new vehicles.

The design of our framework is shown in Fig. 1.3. It builds and updates two blockchains for recording vehicles' communication activities and establishing trust among vehicles. The first blockchain, named *trust points blockchain*, is used to quickly identify and record malicious misbehavior. Intuitively, telling the truth and getting acknowledged by most neighbors will earn trust points, while telling a lie will lose trust points. The second blockchain, named *proof-of-travel blockchain*, accumulates and records each vehicle's long-term contribution to the CV community. Intuitively, the more traffic information it shares with others, the more contribution to the CV community it gets and the higher proof-of-travel credits it can receive from others. This blockchain serves as a low-cost conceptually-centralized tracker of vehicle trust, enabling two vehicles to establish trust when they do not have extended experience with each other, in a way that is secure

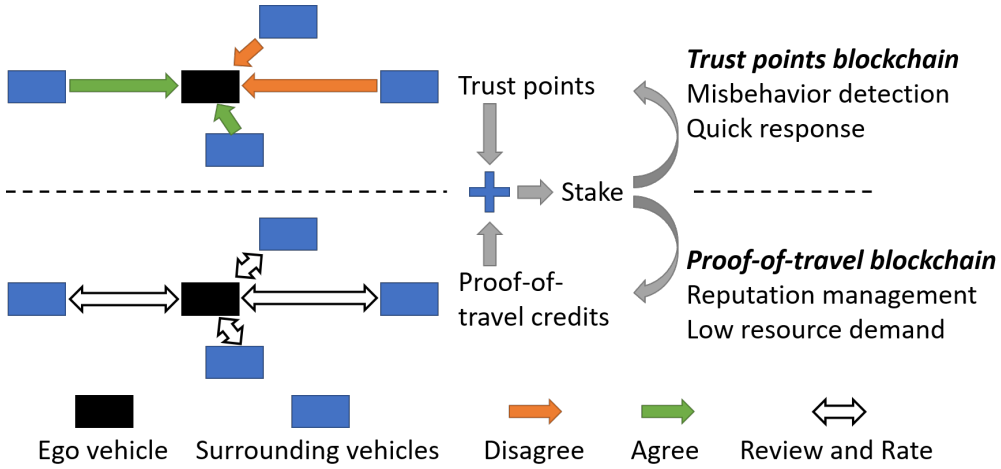

Figure 1.3: Overview of our proposed dual cyber-physical blockchains framework. The ego vehicle is traveling and sharing traffic information with surrounding vehicles. In the trust points blockchain, surrounding vehicles leverage their physical sensing capability to verify the messages sent from the ego vehicle. If a falsified message is detected and reported, the voting contract will instantly start collecting surrounding vehicles' opinions and adjust trust points of relevant vehicles. In the proof-of-travel blockchain, ego vehicle and surrounding vehicles record the number of received messages from different vehicles, which can reflect the travel activities and their contributions. To reduce resource overhead, the credits are updated in a longer period than that of trust points blockchain for reflecting long-term reputation. The stake of each vehicle is computed from both its trust points and proof-of-travel credits. Our dual blockchain design is secure as long as more than 2/3 of the stake is held by honest vehicles.

and dependent on consensus.

Our proposed framework makes it difficult to launch attacks, and facilitates quick detection and reaction to misbehavior. Specifically, once a suspicious message is reported, surrounding vehicles can adapt to more cautious and conservative actions within just hundreds of milliseconds. Within one minute, surrounding vehicles can leverage their sensing capabilities to verify the message and reach consensus on it. Then, the trust points of the vehicle that sent the suspicious message will get updated in the trust points blockchain. On the other hand, as travel history is recorded in the proof-of-travel blockchain, vehicles with poor travel history cannot easily start certain attacks such as Sybil attack or flooding attack (or other attacks that require repeated transmission). Moreover, to

improve the efficiency, we propose the concepts of permanent address and current active address, which can partition all vehicles into different regions with the sharding technique. Our framework design enables record transfer when vehicles move across different regions. Each region can have its blockchains updated and maintained with intra-region and inter-region communication to promote scalability.

1.4 Contributions

In summary, our work makes the following contributions:

- With empirical study and theoretical analysis, we show that for those systems that may evolve into multiple physical scenarios, single neural network based planner is either unsafe, or extremely difficult to verify. We design a novel hierarchical neural network-based planner with assured safety and better verifiability, based on the underlying physical scenarios of the system. We develop novel partition and union techniques to improve efficiency and accuracy of reachability analysis, and propose an overapproximation method for the system under our hierarchical planner. We demonstrate the safety enhancement from our hierarchical design through case studies of unprotected left turn and highway merging, compared with single neural network-based planners.
- We propose a safety-driven interactive planning framework. The framework includes a safety-driven behavior adjustment module that takes the outputs from two neural network-based planners and decides whether to proceed, abort, or hesitate. This is based on analyzing whether a safe evasion trajectory exists, considering the aggressiveness of surrounding vehicles. We demonstrate the advantages of our framework through extensive simulations on synthetic examples and real-world scenarios, when compared with a traditional optimization-based planner and an end-to-end neural network planner. In particular, our framework is safe in all simulations.

Moreover, our framework is *guaranteed* to be safe *if* the aggressive assessment is accurate or if we choose to always treat surrounding vehicles as aggressive. Based on the framework, we propose a connectivity-enhanced planner in mixed traffic environment with human-driven and autonomous vehicles, and connected and non-connected vehicles. We demonstrate significant system performance improvements by leveraging connectivity in dynamic environment. We also analyze system robustness when coordination between connected vehicles is not perfect.

- We propose a speculative planning method to address the challenges in ambiguous scenarios where multiple behaviors and trajectories of surrounding vehicles exist. Our method considers all possible predicted behaviors and trajectories, ensures system safety by ruling out actions that may be unsafe in the worst case, and improves system performance by sampling all possibilities and choosing the action that maximizes the expected reward. Our planner leverages a prediction-planning interface that incorporates uncertainty on both behavior level and trajectory level, including the probability distribution of relevant parameters. It reacts to the prediction changes in real-time and adapts the prediction results based on the system states and traffic environment, to filter out impossible behaviors and trajectories of surrounding vehicles as time goes by for further improving system performance. We demonstrate the advantages of our approach over baseline methods in improving system performance and ensuring system safety. Note that our approach is guaranteed to be safe if the prediction results are conservative and there exists a safe planning decision at the initial state.
- We develop a novel dual blockchain framework that leverages cyber-security techniques, physical sensing capabilities of vehicles, and their travel histories to build trust and secure communication in a large-scale vehicular network, in which every vehicle can get updates timely with low resource overhead. We use a stake-based consensus mechanism across the faster trust points blockchain and the slower proof-of-travel blockchain, a sharding technique for partitioning ve-

hicles into regions, and a dedicated summary step to reduce computation, communication and storage costs of our framework so that it is practical for CV applications. We demonstrate the effectiveness of our framework against a few prevalent attacks, including message spoofing attack, bad mouthing attack, and Sybil attack. We illustrate the performance and timing efficiency of our defense system via simulations in SUMO simulator, and also analyze its resource overhead and trade-offs.

The rest of the dissertation is organized as follows. Chapter 2 presents our design of physics-aware safety-assured hierarchical neural network-based planner. In Chapter 3, we present the safety-driven interactive planning framework, which considers inter-vehicle interactions in dense traffic and can incorporate connectivity in mixed traffic. Chapter 4 presents the safety-assured speculative planning framework with adaptive prediction. In Chapter 5, we present the efficient dual cyber-physical blockchains framework for securing connected vehicle applications. Chapter 6 concludes the dissertation.

CHAPTER 2
SAFETY-ASSURED HIERARCHICAL NEURAL NETWORK-BASED PLANNER

A major challenge for the neural network-based planners is to ensure system safety [24], especially for safety-critical applications [178], [179] and in near-accident scenarios [7], [9], such as unprotected left turn and highway merging in autonomous driving. Our work focuses on addressing such complex scenarios in safety-critical systems. This chapter is based on the work published at [10]. It is organized as follows. Section 2.1 introduces an illustrating example and defines the problem formulation. Section 2.2 presents our planner design and verification approach. Section 2.3 shows the case studies.

2.1 Problem Formulation

We consider the unprotected left turn as a representative application where different planning decisions and system states can eventually lead to different physical scenarios. The system includes a left turn vehicle $C1$ and another vehicle $C2$ going straight from the opposite direction in an

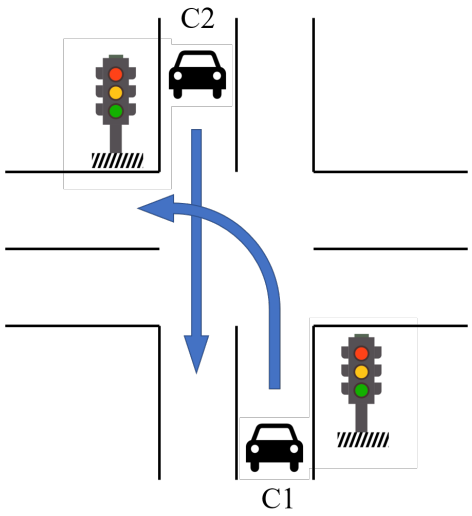

Figure 2.1: The unprotected left turn system.

intersection, as shown in Fig. 2.1. We model it as a 5-dimensional system:

$$\begin{cases} \dot{p}_1(t) = v_1(t) \\ \dot{v}_1(t) = u(t) \\ \dot{\tau}_{min}(t) = f_1(t) \\ \dot{\tau}_{max}(t) = f_2(t) \\ \dot{t} = 1 \end{cases} \tag{2.1}$$

where $p_1(t)$ and $v_1(t)$ are the position and velocity of vehicle $C1$. Vehicle $C2$ is predicted to pass the conflicting area (where the two vehicles may potentially collide) in this intersection within the time window $[\tau_{min}(t), \tau_{max}(t)]$. $u(t)$ is the control input, representing the acceleration of vehicle $C1$. We assume that vehicle $C1$ follows a given path in the intersection to turn left, thus its trajectory can be derived with $u(t)$. $\tau_{min}(t)$ and $\tau_{max}(t)$ may change over time as vehicle $C1$ can

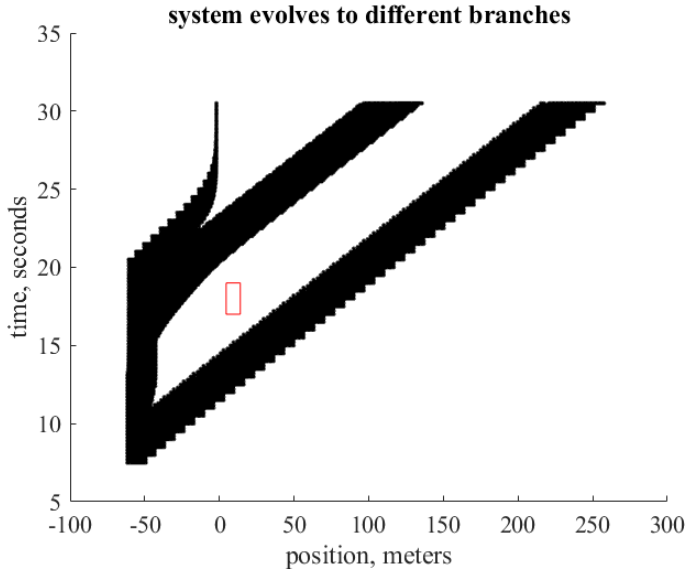

Figure 2.2: System in Fig. 2.1 evolves to multiple physical scenarios. The horizontal axis denotes the position of vehicle $C1$ along the planned path, and a negative value means that $C1$ has not entered the intersection. The vertical axis denotes time. The black region represents simulated trajectories of the system. From left to right, the three branches correspond to the scenarios where vehicle $C1$ stops before the intersection, yields to $C2$, and proceeds, respectively. The red rectangle is the unsafe region as vehicle $C2$ is expected to passing the intersection at time interval $[17, 19]$.

update its prediction for $C2$. We assume that this time window will become tighter as two vehicles get closer to each other, i.e., $f_1(t) >= 0$ and $f_2(t) <= 0$. We also assume that the traffic signal follows a fixed pattern. First it is green for t_g seconds, then it turns yellow for t_y seconds, and then it turns red for t_r seconds. After that, it turns back to green and repeats this turning pattern.

Fig. 2.2 shows the simulated trajectories based on human driving norm. The horizontal axis denotes the position of vehicle $C1$ along the planned path, where a negative value represents that $C1$ has not entered the intersection. When $p_1 = 4.5$ meters, $C1$ enters the region where two vehicles' paths may intersect. When $p_1 = 14$ meters, it leaves that conflicting region. There are three obvious branches as time goes on. The leftmost branch corresponds to the behavior of $C1$ stopping before the intersection, when it cannot pass the intersection before the signal turns red.

The middle branch corresponds to the behavior of $C1$ yielding to vehicle $C2$ that goes straight, when there is potential danger for collision and $C2$ has the right of way. The rightmost branch corresponds to the behavior of $C1$ proceeding, when it is safe to pass the intersection before $C2$. The red rectangle marks the unsafe region, as vehicle $C2$ is expected to pass the conflicting region within the time window $[\tau_{min}(t), \tau_{max}(t)] \equiv [17, 19]$ seconds.

To react safely and efficiently, depending on the initial system state and changes in the surrounding traffic, vehicle $C1$ may take different actions. Note that although there may exist some planner $u'(t)$ that is safe and can lead to only one branch of system trajectories, i.e., braking and then stopping before the intersection in any case, it is not efficient and ideal in real life.

In some simpler systems such as adaptive cruise control and emergency braking [131], [141], the system state may converge to a constant distance gap or gradually slows down to full stop. Here, the potential reachable states of the unprotected left turn system do not converge to a single scenario, but evolve to multiple different scenarios. This presents significant challenges to safety verification and assurance.

Thus, in this work, we are interested in the following questions: Can vehicle $C1$ turn left safely and efficiently under our designed planner when facing different traffic scenarios, i.e., turning left without hesitance when it is safe and decelerating when facing potential collision? If so, can we formally verify the system safety under our designed planner? To answer these, we will first generally formulate the above-mentioned system where different planning decisions and system states can eventually lead to different physical scenarios.

General Formulation. We consider a dynamic system:

$$\begin{cases} \dot{x}(t) = f(x(t), u(t)), \forall t \geq 0 \\ x(t) \in X - S_u, x(0) \in I, u(t) \in U \end{cases} \tag{2.2}$$

where $x(t)$ is the state variable, and $u(t)$ is the control input variable. We assume that f is Lipschitz continuous in x and continuous in u to ensure the uniqueness of solution. By including time t in state variable $x(t)$, system function f can be time-variant. $X = \{x \in \mathbb{R}^n\}$ is the state space. $S_u = \{x \in \mathbb{R}^n | \bigwedge_i h_i(x) \leq 0\}$ is the unsafe state space, h denotes the linear constraint function and state x is unsafe if $\bigwedge_i h_i(x) \leq 0$ is satisfied. $X - S_u$ is the set difference of X and S_u. $I \subseteq X - S_u$ is the initial set of system state. $U = \{u \in \mathbb{R}^m\}$ is the control input space.

Let δ_c denote the control time stepsize. At time $t = i * \delta_c$, $i = 0, 1, 2, \ldots$, the system controller κ takes current state $x(i * \delta_c)$ and computes control input $u(i * \delta_c) = \kappa(x(i * \delta_c))$ for the next time step, the system becomes $\dot{x}(t) = f(x(t), u(i * \delta_c))$ in the time interval $t \in [i * \delta_c, (i + 1) * \delta_c]$.

The trajectory $\varphi_{x(0)}$ to the system (2.2) starting from an initial state $x(0)$ can be formulated as:

$$\dot{\varphi}_{x(0)}(t) = f(\varphi_{x(0)}(t), u(t)), x(0) \in I \tag{2.3}$$

where $\varphi_{x(0)}(0) = x(0)$.

With a well-designed controller $u(t)$, the system trajectories will evolve to disjoint subsets at step i (and possibly at the following steps as well) to avoid the unsafe set. That is:

$$\begin{cases} \varphi_{x(0)}(t) \in \bigcup_k S_k, \forall t \in [(i-1) * \delta_c, i * \delta_c], \forall x(0) \in I \\ ||x - x'|| > \epsilon_x, \exists \epsilon_x > 0, \forall x \in S_k, \forall x' \in S_j, \forall k \neq j \end{cases} \tag{2.4}$$

where δ_c is the control time stepsize, S_k is a subset of system states in time interval $[(i-1) * \delta_c, i * \delta_c]$, $\bigcup_k S_k$ is the union of all subsets. The distance between any two elements $x \in S_k$ and $x' \in S_j$ is always strictly greater than a positive real number ϵ_x, for any two different subsets S_k and S_j.

It becomes more challenging to design a safety-assured planner due to the properties of the system as in (2.2) and (2.4). Since the safe state space $X - S_u$ is non-convex, the system reach-

able set needs to evolve to multiple branches to avoid the unsafe set. However, a planner $\kappa(x)$ that is Lipschitz continuous in x intuitively cannot output significant different control signal $u(x)$ under only minor changes in system state x. For these complex systems, accidents cannot be prevented in experiments with previous neural network-based planner designs, including hierarchical planners [7], [124]–[128]. Thus, we try to answer: is there a planner design that can enable the change of system trajectory under changing scenarios? If so, can we verify its safety as the system reachable states evolve to multiple possible scenarios? Formally, we try to solve:

Problem 1. *For a dynamical system defined by* (2.2) *and* (2.3)*, is there a planner design* κ *that can satisfy* (2.4)*?*

Problem 2. *If there exists a planner* κ *that can satisfy* (2.4)*, the safety verification problem is to determine whether the controlled trajectory* $\varphi_{x(0)}(t) \in X - S_u$*,* $\forall t \geq 0$*,* $\forall x(0) \in I$*.*

2.2 Planner Design and Safety Verification

In this section we first conduct formal analysis for systems under a single neural network-based planner. With theoretical analysis, we show that single neural network planners cannot handle well the systems that may evolve into different scenarios, and are hard to verify. To overcome these challenges, we present our design of a hierarchical neural network-based planner, and then introduce the partition and union algorithm we developed for the verification of our hierarchical planner.

2.2.1 Formal Analysis of Single Planner Design

Using a single neural network for planner design is well-studied [180]–[182]. Deep neural networks provide better performance for complex systems than many traditional methods [114], [183],

[184]. However, single neural network-based planner has its limitations, especially for safety-critical systems [7]. Below we formally analyze the system under a single neural network-based planner with reachability analysis of a neural network controlled system [132]–[134], [143], and we leverage the Bernstein polynomial-based reachability analysis [132], as it can handle neural networks with general and heterogeneous activation functions. Let us start with introducing reachable set and Bernstein polynomial.

Definition 2.2.1. *A system state x is reachable at time $t \geq 0$ on a system defined by* (2.2) *and* (2.3)*, if and only if there exists $x(0) \in I$ such that $x = \varphi_{x(0)}(t)$. The reachable set R of the system is defined as the set of all reachable states $R = \{x | x = \varphi_{x(0)}(t), \forall t \geq 0, \forall x(0) \in I\}$.*

The system is considered to be safe if its reachable set R has no overlap with the unsafe set S_u. However, it is proven that computing the exact reachable set for most nonlinear systems is an undecidable problem [185], not to mention systems with neural network planners. Thus, recent works mainly consider overapproximation of the reachable set. Safety can still be guaranteed if the overapproximated reachable set has no overlap with the unsafe set. Note that in this paper, for simplificy, we use the same notation for both the reachable set and its overapproximation.

For a controller/planner κ_s defined over a n-dimensional state x, its Bernstein polynomials $B_{\kappa_s,d}(x)$ under degree $d = (d_1, d_2, \ldots, d_n)$ is:

$$B_{\kappa_s,d}(x) = \sum_{\substack{0 \leq a_j \leq d_j \\ j \in \{1,2,\ldots,n\}}} \kappa_s\left(\frac{a_1}{d_1}, \frac{a_2}{d_2}, \ldots, \frac{a_n}{d_n}\right) \prod_{j=1}^{n} \left(\binom{d_j}{a_j} x_j^{a_j} (1 - x_j)^{d_j - a_j} \right) \tag{2.5}$$

where $\binom{d_j}{a_j}$ is a binomial coefficient.

To obtain an overapproximation of the reachable set for a system with a neural network-based controller κ_s, we compute an overapproximation of the controller κ_s using Bernstein polynomials

similarly as in [132]. Note that the reachable set is computed step by step and it is sufficient to perform the overapproximation of κ_s at step i on the latest computed reachable set R_{i-1}. That is, κ_s is overly approximated by a Bernstein polynomial with bounded error ϵ on set R_{i-1} as:

$$\kappa_s(x) \in B_{\kappa_s,d}(x) + [-\epsilon, \epsilon], \forall x \in R_{i-1} \tag{2.6}$$

where $R_0 = I$ when performing overapproximation in the first step. In the rest of the paper, $B_{\kappa_s}(x)$ is short for $B_{\kappa_s,d}(x)$, as it is not necessary to have the same d for the overapproximation of different controllers.

With the above approach, the dynamic system with a single neural network-based planner κ_s is transformed into a polynomial system for computing the overapproximation of the reachable set. This enables our following analysis.

Challenge on correctness. Intuitively it is unlikely that Lipschitz continuous planners can output significantly different control signal $u(x)$ under only minor changes in the system state x, and enable system trajectory go into several disjoint subsets under different scenarios. Next, let us formally introduce Lipschitz constant and explain that a large number of single neural network planners may indeed be unsafe for the system.

Definition 2.2.2. *A real-valued function $f : X \to \mathbb{R}$ is called Lipschitz continuous over $X \subseteq \mathbb{R}^n$, if there exists a non-negative real L, such that $||f(x) - f(x')|| \leq L||x - x'||$ for $\forall x, x' \in X$. Any such L is called a Lipschitz constant of f over X.*

Proposition 2.2.1. *For a dynamical system defined by* (2.2) *and* (2.3) *with single neural network-based planner κ_s, if κ_s is a convolutional or fully connected neural network with ReLU, sigmoid or hyperbolic tangent (tanh) activation functions, then the controlled trajectory $\varphi_{x'(0)}(t')$ will not evolve to several branches as formulated in* (2.4).

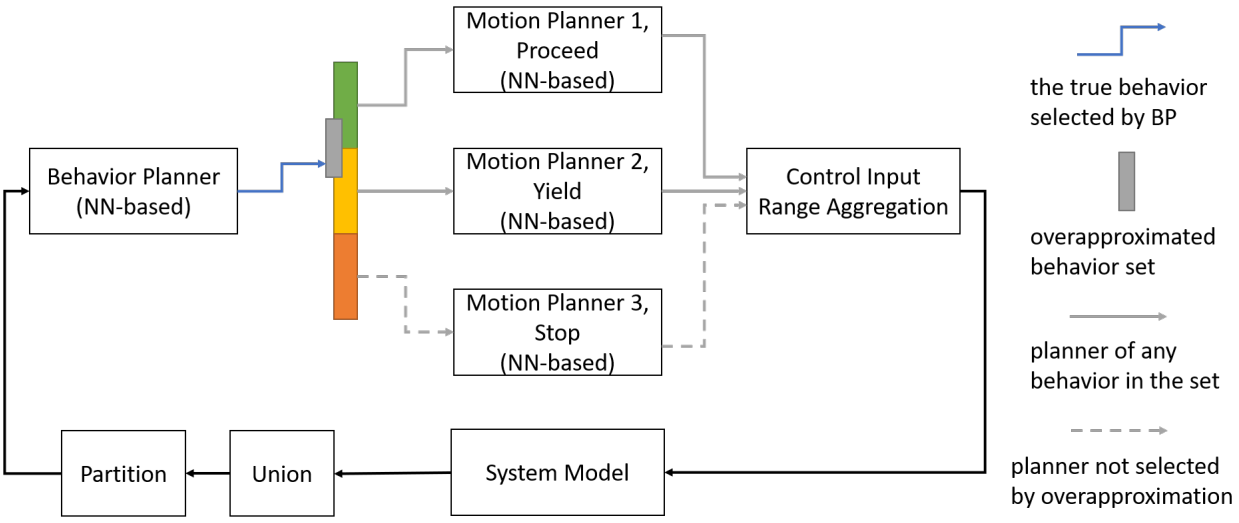

Figure 2.3: Our design of the hierarchical neural network-based planner consists of one behavior planner μ and N motion planners $\{\kappa_1, \kappa_2, \ldots, \kappa_N\}$. Take the unprotected left turn system as an example, there are three underlying physical scenarios: vehicle $C1$ may stop before the intersection, yield to vehicle $C2$, or proceed, and they correspond to three motion planners shown here in the figure. The behavior planner decides the most appropriate behavior for vehicle $C1$ given the system state x, and then the corresponding motion planner is enabled to control the system. To compute an overapproximation of the reachable set of the system under such hierarchical planner, we first compute an overapproximated behavior set with Bernstein polynomial approximation as in Eq. (2.5) and (2.6), which is illustrated by the grey rectangle in the figure. Then for each behavior in the overapproximated behavior set, the corresponding motion planner's output range can be aggregated as the possible control input range, thus computing an overapproximation of the system state reachable set under all possible behaviors.

Proof. We will first prove that if a neural network planner κ_s can ensure that the system evolves to several branches as defined in (2.4), κ_s is not Lipschitz continuous. We assume it is at step i that the reachable set R_i can be represented as $R_i = \bigcup_k S_k$ as in (2.4) for the first time. Then there exists $x_{i-1} \in R_{i-1}$ and $x'_{i-1} \in R_{i-1}$ such that $||x_{i-1} - x'_{i-1}|| \to 0$, $x_{i-1} + f(x_{i-1}, \kappa_s(x_{i-1})) * \delta t = x_i \in R_k$, $x'_{i-1} + f(x'_{i-1}, \kappa_s(x'_{i-1})) * \delta t = x'_i \in R_j$, $k \neq j$ and $\delta t \to 0$. According to (2.4), there exists $\epsilon_x > 0$ such that $||x_i - x'_i|| > \epsilon_x$, and we have $||f(x_{i-1}, \kappa_s(x_{i-1})) - f(x'_{i-1}, \kappa_s(x'_{i-1}))|| > \epsilon_f$ and $\epsilon_f > 0$. Because f is Lipschitz continuous, then κ_s is not Lipschitz continuous. However, since κ_s is

a convolutional or fully connected neural network with ReLU, sigmoid and hyperbolic tangent (tanh) activation functions, it should be Lipschitz continuous [132]. From this contradiction, we know that the controlled trajectory $\varphi_{x'(0)}(t')$ will not evolve to several branches as formulated in (2.4). □

Based on this proposition, for most single neural network-based planners, the system reachable set will cover the unsafe region S_u and there is actually no disjoint subsets.

Challenge on verifiability. Even if κ_s is a neural network-based planner that can satisfy (2.4), it will have infinitely large Lipschitz constant according to Proposition 2.2.1. This typically makes the safety verification extremely hard due to the importance of Lipschitz constant in the construction of reachable sets. As observed in [134], [137], [142] and shown in our case studies in Section 2.3, the reachable set expands more quickly when the Lipschitz constant of neural network-based planner is larger. In which case, the verification process may terminate due to uncontrollable approximation error or excessively long computation time. These challenges can be overcome in our hierarchical planner design as introduced below.

2.2.2 Hierarchical Planner Design and Reachability Analysis

Hierarchical planner design. The drawbacks of a single neural network-based planner motivate us to propose a hierarchical planner, as shown in Fig. 2.3. The main idea is to learn different motion planners for different physical scenarios and a system-level behavior planner for changing between scenarios. Specifically, our hierarchical planner consists of a behavior planner μ and N motion planners $\{\kappa_1, \kappa_2, \ldots, \kappa_N\}$, assuming that the system may evolve into N different physical scenarios. These planners are all neural network-based and take system states as inputs. The behavior planner's output $\mu(x)$ can be mapped to the discrete behavior choice by mapping function f_m and we have $f_m(\mu(x)) \in \{\kappa_1, \kappa_2, \ldots, \kappa_N\}$, while motion planners output control variable $u(t)$.

To illustrate the idea, we still use the unprotected left turn system as an example: The behavior planner will decide the most appropriate behavior for vehicle $C1$ given the system state x. Then the corresponding motion planner, e.g., motion planner 2 in Fig. 2.3, will be enabled to control the system. The system will use the same motion planner before the next triggering of the behavior planner. Note that it is flexible to set the trigger conditions for the behavior planner, e.g., behavior planner may be triggered every t_{bp} seconds when the system state has significant changes.

The challenges under a single neural network planner κ_s can be overcome in our hierarchical planner design as the reachable sets computed under different motion planners correspond to different physical scenarios and are disjoint. For each underlying physical scenario, the corresponding motion planner can generate system trajectory to avoid the unsafe region. Given that all motion planners are safe in their corresponding scenarios, system safety can be guaranteed with the computation of an overapproximated behavior set for the system-level behavior planner.

Reachability verification. Next we first present our partition and union algorithm for general reachable set computation to improve efficiency and accuracy, and then we introduce our method to overapproximate the reachable set for a system under a hierarchical planner $\mu(\kappa_1, \kappa_2, \ldots, \kappa_N)$.

We develop a partition and union method that can improve the efficiency and accuracy of the overapproximated reachable set at every computation step, as shown in Algorithm 1. Specifically, when the system state has not fully reached the goal set S_g and the system is currently safe (line 2), we will keep computing the reachable set as follows. We first partition the initial set I into grids of size δ (line 3), and then compute the reachable set for each grid I_γ in the next n steps (line 5). The computation process may terminate without returning R_γ (line 6) due to memory limitation, low accuracy or no result within certain time. In that case, we will further partition the initial set I_γ (line 7) and compute the reachable set (line 9). Once the reachable set R_γ is computed for each grid, we union them as the system reachable set for this round, and reset the initial set $I = \underset{\gamma}{\cup} R_\gamma$

Algorithm 1: Partition and Union for Reachable Set Computation

Result: Reachable set R and verification result
Input: Initial set I, unsafe set S_u and goal set S_g

1 $R \leftarrow I$;
2 **while** $I - S_g \neq \emptyset$ *and* $R \cap S_u = \emptyset$ **do**
3 $\quad \{I_\gamma\} \leftarrow Partition(I, \delta)$;
4 $\quad$ **for** *each grid* I_γ **do**
5 $\quad\quad R_\gamma \leftarrow Reach\_Comp(I_\gamma, n)$;
6 $\quad\quad$ **while** $Reach\_Comp$ *terminates without returning* R_γ **do**
7 $\quad\quad\quad \{I_{\gamma'}\} \leftarrow Partition(I_\gamma, \delta/2)$;
8 $\quad\quad\quad$ **for** *each grid* $I_{\gamma'}$ **do**
9 $\quad\quad\quad\quad R_{\gamma'} \leftarrow Reach\_Comp(I_{\gamma'}, n)$;
10 $\quad\quad\quad$ **end**
11 $\quad\quad$ **end**
12 $\quad$ **end**
13 $\quad I \leftarrow \underset{\gamma}{\cup} R_\gamma$;
14 $\quad R \leftarrow R \cup I$;
15 **end**
16 **if** $R \cap S_u = \emptyset$ **then**
17 $\quad$ Verification result is safe.
18 **else**
19 $\quad$ Verification result is uncertain.
20 **end**

(line 13). This process repeats until the system state reaches the goal set S_g and is verified to be safe, or the overapproximation of the reachable set has overlap with the unsafe set S_u (which presents an uncertain verification results given the nature of overapproximation).

To compute an overapproximated reachable set for a system with planner $\mu(\kappa_1, \kappa_2, \ldots, \kappa_N)$, we first overapproximate $\mu(x)$ in a similar way as in (2.6) as:

$$\mu(x) \in B_\mu(x) + [-\epsilon_\mu, \epsilon_\mu], \forall x \in R_{i-1} \tag{2.7}$$

where R_{i-1} is the overapproximated reachable set of the system in the $(i-1)$-th step. We then

compute the overapproximated output range of $\mu(x)$, R_i^{μ}, on the set R_{i-1}. By the mapping function f_m, we have the overapproximated set of the selected planner $S_{ctrl} \subseteq \{\kappa_1, \kappa_2, \ldots, \kappa_N\}$. For any planner $\kappa_a(x) \in S_{ctrl}$, we can compute its overapproximated system state reachable set $R_i^{\kappa_a}$ on R_{i-1}. Then, the overapproximated reachable set of the system at step i is $R_i = \bigcup_{\kappa_a \in S_{ctrl}} R_i^{\kappa_a}$. The soundness of our approach can be proven below.

Proposition 2.2.2. *(Soundness). For a dynamical system defined by* (2.2)*,* (2.3) *and* (2.4) *with a hierarchical neural network-based planner* $\mu(\kappa_1, \kappa_2, \ldots, \kappa_N)$*, the controlled trajectory* $\varphi_{x(0)}(t)$ *satisfies* $\varphi_{x(0)}(t) \in \bigcup_{\kappa_a \in S_{ctrl}} R_i^{\kappa_a}$, $\forall t \in [(i-1) * \delta_c, i * \delta_c]$, $\forall i \in \mathbb{N}^+$, $\forall x(0) \in I$.

Proof. Let us prove by contradiction. We assume that it is at step i that we compute the wrong reachable set $R_i = \bigcup_{\kappa_a \in S_{ctrl}} R_i^{\kappa_a}$ for the first time. Thus $\exists x'(0) \in I$ and $\exists \kappa' \in \{\kappa_1, \kappa_2, \ldots, \kappa_N\}$ such that $\varphi_{x'(0)}((i-1) * \delta_c) = x' \in R_{i-1}$, $\varphi_{x'(0)}(t) \notin \bigcup_{\kappa_a \in S_{ctrl}} R_i^{\kappa_a}$ and $f_m(\mu(x')) = \kappa'$. Since $f_m(\mu(x')) = \kappa'$, we have $\kappa' \in S_{ctrl}$ and $R_i^{\kappa'} \subseteq \bigcup_{\kappa_a \in S_{ctrl}} R_i^{\kappa_a}$. Finally $\varphi_{x'(0)}(t) \in R_i^{\kappa'}$ contradicts that $\varphi_{x'(0)}(t) \notin \bigcup_{\kappa_a \in S_{ctrl}} R_i^{\kappa_a}$.

□

2.3 Case Studies

Our hierarchical neural network-based planner design and safety verification method can be applied to many safety critical systems defined by (2.2), (2.3) and (2.4). In this section, we demonstrate its effectiveness in two case studies of unprotected left turn and highway merging in autonomous driving.

For both applications, we generate the trajectory dataset based on human driving norm and use the same dataset to train all the planning neural networks (κ_s, μ, κ_1, κ_2, …, κ_N). These neural networks all have two hidden layers, with each layer having ten neurons. We select ReLU and tanh

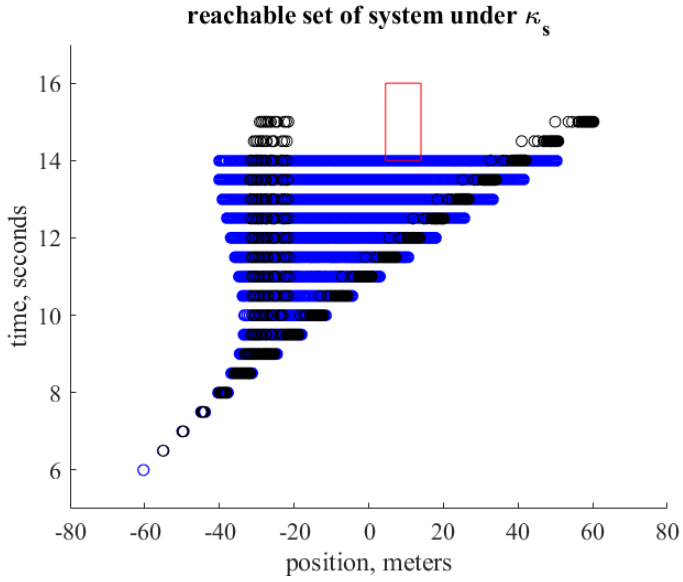

Figure 2.4: Reachable set and sampled trajectories for the unprotected left turn system with a single neural network planner κ_s. The blue region is the overapproximated reachable set and the black region is 100 sampled trajectories from the same initial set. Initial set I is set as $I = \{x \in \mathbb{R}^5 | p_1 \in [-60.4, -60.3], v_1 \in [10.5, 10.51], \tau_{min} \equiv 14, \tau_{max} \equiv 16, t = 6\}$. Unsafe set is $S_u = \{x \in \mathbb{R}^5 | p_1 \in [4.5, 14], t \in [14, 16]\}$, and it is marked with a red rectangle.

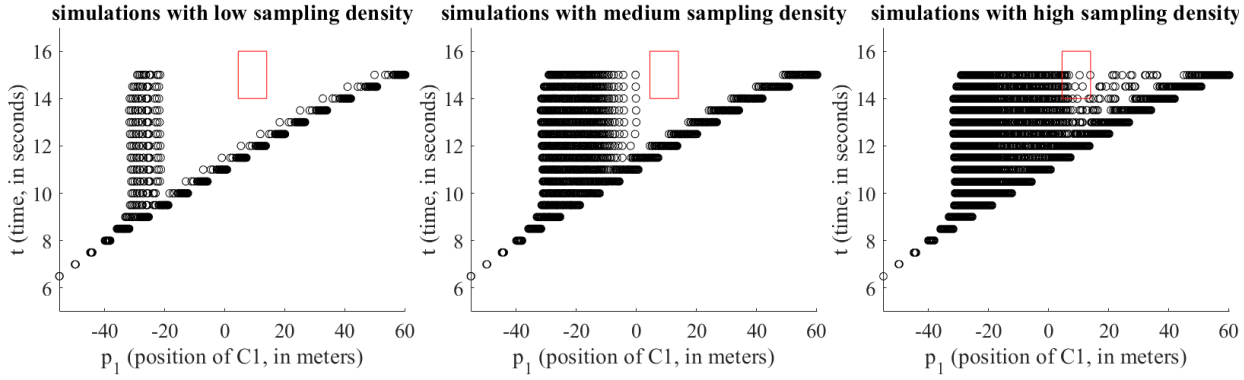

Figure 2.5: Simulated trajectories with different sampling densities from the initial set I, for the unprotected left turn system with a single neural network-based planner κ_s. The black region represents simulated trajectories and the red rectangle is the unsafe set S_u. I and S_u are the same as in Fig. 2.4. From left to right, these three subplots correspond to the trajectories of one hundred, ten thousand and one million samples from the initial set I, respectively.

as the activation functions for the hidden layer and the output layer, respectively. We set control time stepsize $\delta_c = 0.5$ seconds for all experiments in this work. Based on the verification tool ReachNN [132] and POLAR [186], we implement Algorithm 1 to compute the overapproximation of reachable set for the system under the single neural network-based planner and the hierarchical neural network-based planner, respectively.

2.3.1 Unprotected Left Turn

We conduct experiments for the unprotected left turn system as described earlier in Section 2.1.

Fig. 2.4 shows the overapproximated reachable set under a single neural network-based planner κ_s, which is consistent with our analysis in Proposition 2.2.1. We select the initial set $I = \{x \in \mathbb{R}^5 | p_1 \in [-60.4, -60.3], v_1 \in [10.5, 10.51], \tau_{min} \equiv 14, \tau_{max} \equiv 16, t = 6\}$ to cover different behaviors (proceed and yield) of vehicle $C1$. The unsafe set S_u is determined by τ_{min} and τ_{max}, i.e., $S_u = \{x \in \mathbb{R}^5 | p_1 \in [4.5, 14], t \in [14, 16]\}$. In this figure, the blue region represents the reachable set and the black region is the simulated trajectories of 100 sampling states from the same initial set. The reachable set cannot be represented with several disjoint subsets as described in Eq. (2.4) and it overlaps with the unsafe set S_u.

By increasing the number of sampling states from the same initial set I, we actually find counterexamples that prove κ_s is indeed unsafe. Fig. 2.5 shows the simulated trajectories with different sampling density from I. The three subplots from left to right correspond to the trajectories of one hundred, ten thousand and one million samples in the initial set I, respectively. Based on our analysis in Proposition 2.2.1, for any other neural network planner κ_s' that is a convolutional or fully connected neural network with ReLU, sigmoid and tanh activation functions, we can always find counterexamples by increasing the sampling density.

For the system with our design of a hierarchical planner $\mu(\kappa_1, \kappa_2, \kappa_3)$, we compute the overap-

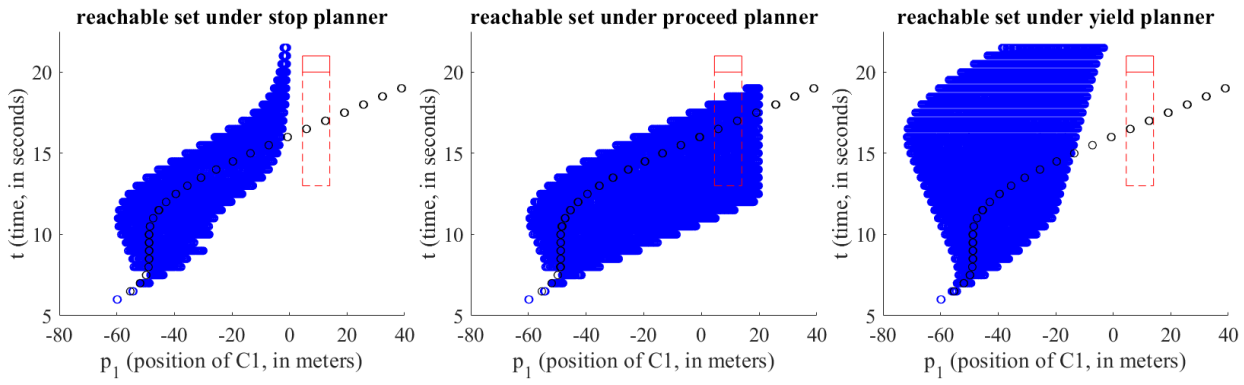

Figure 2.6: Reachable set and sampled trajectories for the unprotected left turn system under hierarchical neural network planner with changing time window $[\tau_{min}, \tau_{max}]$. From left to right, the three subplots present the reachable system states under the stop motion planner, proceed motion planner, and yield motion planner respectively. The blue region is the overapproximated reachable set and the black region is 100 sampled trajectories from the same initial set. Initial set I is set as $I = \{x \in \mathbb{R}^5 | p_1 \in [-60, -59.7], v_1 \in [10.5, 10.51], \tau_{min} = 13, \tau_{max} = 21, t = 6\}$. The time window $[\tau_{min}, \tau_{max}]$ is initially $[13, 21]$ at time $6 \leq t < 7$, then $[15, 21]$ at time $7 \leq t < 8$, $[17, 21]$ at time $8 \leq t < 9$, $[19, 21]$ at time $9 \leq t < 10$, and finally $[20, 21]$ at time $t \geq 10$. The unsafe set S_u changes with the time window $[\tau_{min}, \tau_{max}]$ as time goes on. It is initially the red rectangle with the dashed line and finally the red rectangle with the solid line.

proximation of the reachable set with the method introduced in Section 2.2.2, and the results are shown in Figs. 2.7 and 2.6.

First, we assume that τ_{min} and τ_{max} do not change over time. We set the initial set $I = \{x \in \mathbb{R}^5 | p_1 \in [-64.35, -64.05], v_1 \in [10.5, 10.51], \tau_{min} \equiv 14, \tau_{max} \equiv 16, t = 6\}$ to cover different behaviors (proceed and yield) of vehicle $C1$. Then the unsafe set is $S_u = \{x \in \mathbb{R}^5 | p_1 \in [4.5, 14], t \in [14, 16]\}$. As shown in Fig. 2.7, due to overapproximation, all three behaviors are included in the reachable set S_{ctrl} by the behavior planner μ at the beginning. Since the motion estimation for the other vehicle $C2$ remains the same, we assume that the behavior planner is triggered only once. From the figure, we can observe three disjoint reachable subsets in the time interval $t \in [10, 16]$ and the reachable set has no overlap with the red unsafe region. Thus the planner $\mu(\kappa_1, \kappa_2, \kappa_3)$ is verified to be safe in this example.

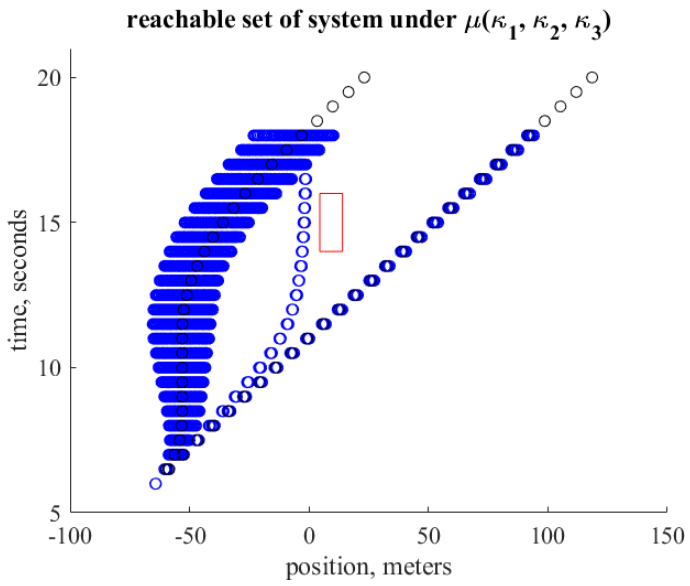

Figure 2.7: Reachable set and sampled trajectories for the unprotected left turn system with hierarchical neural network planner $\mu(\kappa_1, \kappa_2, \kappa_3)$. The blue region is the overapproximated reachable set and the black region is 100 sampled trajectories from the same initial set. Initial set I is set as $I = \{x \in \mathbb{R}^5 | p_1 \in [-64.35, -64.05], v_1 \in [10.5, 10.51], \tau_{min} \equiv 14, \tau_{max} \equiv 16, t = 6\}$. Unsafe set is $S_u = \{x \in \mathbb{R}^5 | p_1 \in [4.5, 14], t \in [14, 16]\}$, and it is marked with a red rectangle. States in the initial set I is assessed by the behavior planner μ and in this example S_{ctrl} includes all three behaviors (proceed, yield and stop). At the following time steps, vehicle $C1$ will not adjust its behavior and the reachable set of different behaviors are computed independently.

We then consider the case where vehicle $C1$ updates the motion estimation for vehicle $C2$ as the two vehicles get closer to the intersection (which is often the case in practice), and we verify the system safety with the same hierarchical planner. Fig. 2.6 presents the system state reachable sets under different motion planners $\{R^{\kappa_1}, R^{\kappa_2}, R^{\kappa_3}\}$, which all together form the reachable set R under our hierarchical planner. We set the initial set $I = \{x \in \mathbb{R}^5 | p_1 \in [-60, -59.7], v_1 \in [10.5, 10.51], \tau_{min} = 13, \tau_{max} = 21, t = 6\}$. The time window $[\tau_{min}, \tau_{max}]$ is initially $[13, 21]$ at time $6 \leq t < 7$, then $[15, 21]$ at time $7 \leq t < 8$, $[17, 21]$ at time $8 \leq t < 9$, $[19, 21]$ at time $9 \leq t < 10$, and finally $[20, 21]$ at time $t \geq 10$. The unsafe set S_u changes with the time window $[\tau_{min}, \tau_{max}]$ as time goes on. In Fig. 2.6, the red rectangle with dashed line is the initial

unsafe region, and the red rectangle with solid line is the final unsafe region. As there is no overlap between the reachable set with unsafe region, the planner $\mu(\kappa_1, \kappa_2, \kappa_3)$ is verified to be safe in this case where $C1$ updates its estimation on $C2$ and the time window $[\tau_{min}, \tau_{max}]$ is updated over time.

Since the behavior planner is triggered multiple times in the case in Fig. 2.6, the eventual reachable set R includes system trajectories under switched motion planners and is significantly larger than the reachable set in Fig. 2.7. Intuitively, if the behavior planner is triggered more frequently, the vehicle $C1$ can adapt to a more appropriate behavior sooner, but this will also results in a larger reachable set and more verification effort.

2.3.2 Highway Merging

Another common and challenging task for autonomous driving is highway merging. As shown in Fig. 2.8, vehicle $C1$ intends to merge onto the highway from on-ramp while another vehicle $C2$ stays on the highway. The system can be modeled as follows:

$$\begin{cases} \dot{p}_1(t) = v_1(t) \\ \dot{v}_1(t) = u(t) \\ \dot{p}_2(t) = v_2(t) \\ \dot{v}_2(t) = 0 \end{cases} \tag{2.8}$$

where $p_1(t)$, $v_1(t)$, $p_2(t)$ and $v_2(t)$ are the longitudinal position and the velocity of vehicle $C1$ and $C2$, respectively. $u(t)$ is the control input, which is the acceleration of vehicle $C1$. In this example, we assume that vehicle $C2$ is a heavy truck and will maintain its speed.

To simplify this problem, we only discuss longitudinal motion of vehicle $C1$. Merging is

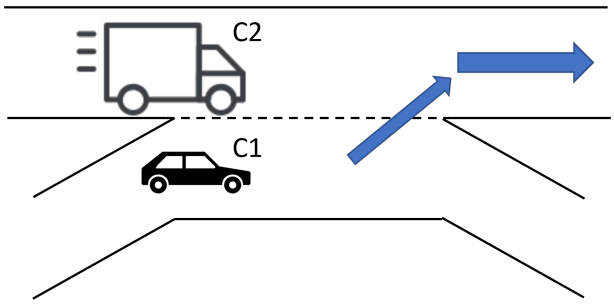

Figure 2.8: The highway merging system. Vehicle $C1$ is merging onto the highway and vehicle $C2$ stays on the highway. Depending on the positions and velocities of vehicle $C1$ and $C2$, vehicle $C1$ may yield to vehicle $C2$, or proceed.

considered to be feasible and safe if the longitudinal distance $|p_1(t_x) - p_2(t_x)|$ is larger than a threshold d_{th} at some time point t_x when vehicle $C1$ has not reached the end of the side road, i.e, $p_1(t_x) < p_{end} = 150$ meters. Different from the unprotected left turn case, the unsafe set S_u is not fixed here. The system is safe as long as there exists a position window $[p_{1,min}, p_{1,max}]$ for vehicle $C1$ and unsafe set $S_u = \{x \in \mathbb{R}^4 || p_1(t) - p_2(t)| \leq d_{th}, p_1(t) \in [p_{1,min}, p_{1,max}]\}$ such that $S_u \cap R = \emptyset$ and $0 \leq p_{1,min} < p_{1,max} \leq p_{end}$. We select the initial set $I = \{x \in \mathbb{R}^4 | p_1 = 0, v_1 = 25, p_2 \in [-24.5, -23.5], v_2 \in [24.5, 25.5]\}$ such that vehicle $C1$ may need to choose different behaviors according to the system state.

Similarly as the unprotected left turn case study, we first consider the highway merging system under a single neural network planner κ_s, and Fig. 2.9 shows the overapproximated reachable set in this case. The reachability analysis is interrupted due to increasing approximation error, and thus the blue region only shows the a subset of the reachable set where the position of truck p_2 is within 40 meters. This is consistent with our analysis that extremely large Lipschitz constant in this case may greatly increase the difficulty in verification. The black region is the sampled trajectories, which should be strictly covered by the reachable set (if it were computed). From this figure, we cannot find an unsafe set S_u that has no overlap with the reachable set, and the system is analyzed

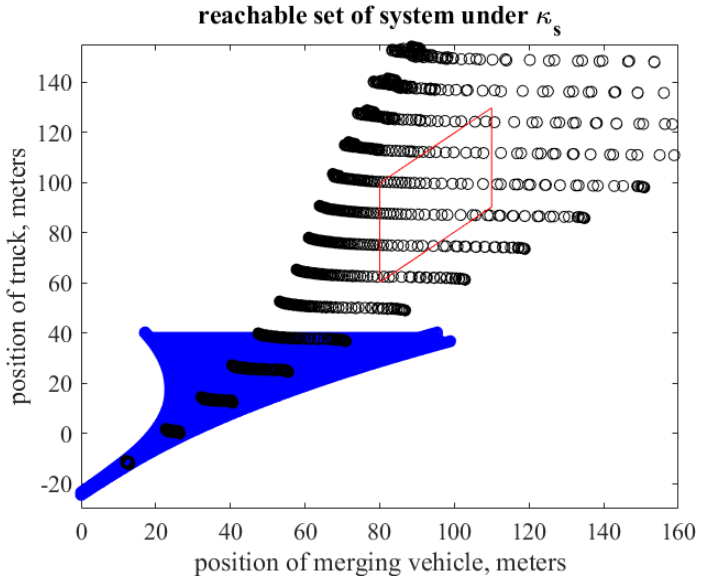

Figure 2.9: Reachable set and sampled trajectories for the highway merging system under a single neural network planner κ_s. Due to the increasing approximation error, reachability analysis is interrupted and the blue region only shows an overapproximated subset of the reachable set where the position of truck p_2 is within 40 meters. The black region is 100 sampled trajectories from the same initial set $I = \{x \in \mathbb{R}^4 | p_1 = 0, v_1 = 25, p_2 \in [-24.5, -23.5], v_2 \in [24.5, 25.5]\}$. An example of unsafe set is $S_u = \{x \in \mathbb{R}^4 | |p_1(t) - p_2(t)| \leq 19.75, p_1(t) \in [80, 110]\}$, and it is marked with a red parallelogram.

to be unsafe under the planner κ_s.

We then consider the highway merging system under our design of a hierarchical neural network-based planner $\mu(\kappa_1, \kappa_2)$. We use the same method in Section 2.2.2 to compute the overapproximation of the reachable set in this case, as shown in Fig. 2.10. We assume that the behavior planner μ is triggered only once at the beginning. The left and right branch correspond to yield and proceed behavior of vehicle $C1$, respectively. We can find an unsafe region marked by a red parallelogram, $S_u = \{x \in \mathbb{R}^4 | |p_1(t) - p_2(t)| \leq 19.75, p_1(t) \in [80, 110]\}$. Since this S_u has no overlap with the overapproximated reachable set, vehicle $C1$ can safely merge into the highway when $p_1(t) \in [80, 110]$. This case study once again demonstrates the verifiability and safety of our

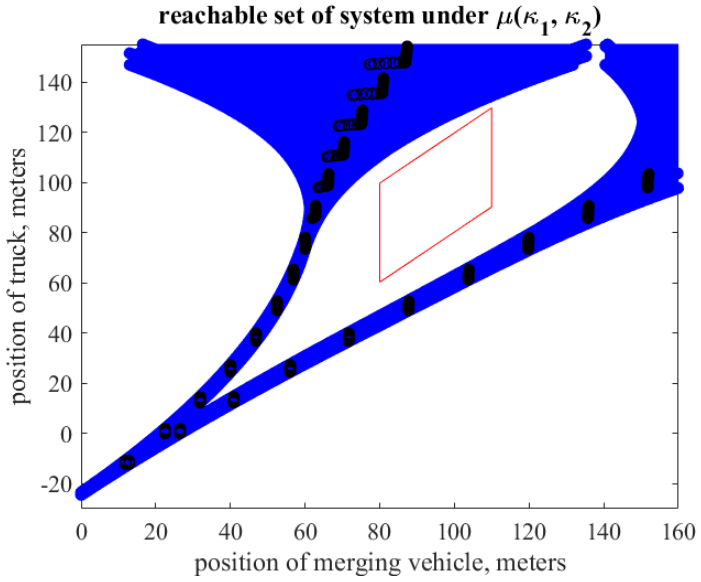

Figure 2.10: Reachable set and sampled trajectories for the highway merging system under a hierarchical neural network planner $\mu(\kappa_1, \kappa_2)$. The blue region is the overapproximated reachable set and the black region is 100 sampled trajectories from the same initial set $I = \{x \in \mathbb{R}^4 | p_1 = 0, v_1 = 25, p_2 \in [-24.5, -23.5], v_2 \in [24.5, 25.5]\}$. In this case, we can find an unsafe set $S_u = \{x \in \mathbb{R}^4 | |p_1(t) - p_2(t)| \leq 19.75, p_1(t) \in [80, 110]\}$, and it is marked with a red parallelogram. Since $S_u \cap R = \emptyset$, vehicle $C1$ can safely merge onto the highway when $p_1(t) \in [80, 110]$.

hierarchical neural network planner.

CHAPTER 3
SAFETY-DRIVEN INTERACTIVE NEURAL NETWORK-BASED PLANNER

In this chapter, we present our safety-driven neural network-based interactive planning framework to prevent over-conservative planning while ensuring safety, which is based on the work published at [25], [119]. In Section 3.1, we present our safety-driven interactive planning framework without leveraging connectivity. Section 3.2 shows the experimental results. In Section 3.3, we present our connectivity-enhanced planning framework, which can be employed in mixed traffic scenarios. Section 3.4 shows the experimental results. Although Planner designs in this chapter are demonstrated in the example of lane changing in dense traffic, they can be widely applied in many challenging scenarios.

3.1 Planner Design with Safety Guarantee

Our proposed framework design is shown in Fig. 3.1. Let us take lane changing in dense traffic as a challenging example. In this framework, we consider neural network-based planners for longitudinal and lateral motion planning, with more details in Section 3.1.1. To improve lane changing success rate in dense traffic, we leverage another neural network to assess the aggressiveness of the following vehicle F in the target lane, which is discussed in Section 3.1.2. Then, based on the predicted behavior of vehicle F (aggressive or cautious), we conduct safety analysis and compute the critical region to avoid collision in the predicted worst case. The trajectory planned under neural networks is adjusted in advance if there is any possibility of collision during the lane changing process. This is detailed in Section 3.1.3.

Table 3.1: Input and output of neural network-based planners.

notation	definition
inputs	
p_x	longitudinal position of the ego vehicle
p_y	lateral position of the ego vehicle
v_x	longitudinal velocity of the ego vehicle
v_y	lateral velocity of the ego vehicle
$p_{x,l}$	longitudinal position of the leading vehicle L
$v_{x,l}$	longitudinal velocity of the leading vehicle L
$p_{x,f}$	longitudinal position of the following vehicle L
$v_{x,f}$	longitudinal velocity of the following vehicle L
outputs	
a_x	longitudinal acceleration of the ego vehicle
a_y	lateral acceleration of the ego vehicle

3.1.1 Longitudinal and Lateral Planners

The two neural networks for longitudinal and lateral planning each has seven independent input variables[1] and one of the two output variables, as summarized in Table 3.1. In order to cover as many traffic scenarios as possible, we synthesize the dataset by simulations according to human driving norm [10]. The ego vehicle initially is at the center of the original lane and has no lateral speed. Then it changes lanes under an MPC controller [47] with step size $\delta t = 0.1$ second. The optimization goal is to minimize fuel consumption and lane changing time while satisfying safety and comfort constraints.

The dataset collects system states and accelerations of the ego vehicle at every step, which is composed of about 36 million entries. The neural network planner is trained to minimize mean squared error with the Adam optimizer. Note that using more comprehensive datasets with good performance (either from human driving trajectories or from synthesized trajectories), the neural network-based planners can be further improved.

[1]There are eight input variables in Table 3.1. However, among p_x, $p_{x,l}$ and $p_{x,f}$, only two of them are independent.

3.1.2 Aggressiveness Assessment and Behavior Prediction

According to [103], the driving behavior of the following vehicle in the target lane follows one model when it is cautious and another model when it is aggressive. This assumption is validated by real-world human driving data. In this work, we use similar assumptions. We assume that the following vehicle F follows the ego vehicle E when it is cautious and follows the leading vehicle L when it is aggressive.

For both two cases, we leverage a neural network to predict the accelerations of the following vehicle. Let a_1 and a_0 denote the accelerations when it is cautious or aggressive, respectively. Different from the motion planner, we do not need p_y and v_y as input variables of the neural network.

For this prediction task, we also synthesize the dataset via simulations. We generate various types of traffic states for the three vehicles and compute the accelerations of the following vehicle F with the Intelligent Driver Model (IDM) [187]. The parameters in the IDM model are uniformly sampled by following $a_{x,a} = 4\ \mathrm{m/s^2}$, $v_m = \dot{h} + v_{x,f}$, $5\ \mathrm{m} \leq h_s \leq 8\ \mathrm{m}$, $1\ \mathrm{s} \leq t_g \leq 2\ \mathrm{s}$, $a_{x,d} = 6\ \mathrm{m/s^2}$. With the dataset of one million entries, we train the neural network to minimize mean squared error with the Adam optimizer.

The following vehicle's behavior is predicted by comparing its true acceleration $a^*_{x,f}$ with the predicted a_1 and a_0. When $a^*_{x,f}$ is closer to a_1, the following vehicle F is predicted as cautious and follows the ego vehicle E; when $a^*_{x,f}$ is closer to a_0, it is predicted as aggressive and follows the

leading vehicle L:

$$\begin{cases} |a^*_{x,f} - a_1| < |a^*_{x,f} - a_0| - a_{th} \rightarrow \text{vehicle } F \text{ is cautious} \\ |a^*_{x,f} - a_0| < |a^*_{x,f} - a_1| - a_{th} \rightarrow \text{vehicle } F \text{ is aggressive} \\ -a_{th} \leq |a^*_{x,f} - a_0| - |a^*_{x,f} - a_1| \leq a_{th} \rightarrow \text{uncertain} \end{cases} \tag{3.1}$$

Here a_{th} is a threshold. Larger a_{th} means higher confidence on the behavior prediction. For those uncertain scenarios, we assume that the vehicle F is aggressive so that the planned trajectory for ego vehicle is conservative and safe.

Based on the predicted behavior of the following vehicle, we conduct safety analysis. We assume that (1) if the following vehicle is cautious and willing to create gap for ego vehicle, it can at least decelerate with $a_{x,f,d} = 6$ m/s^2; and (2) if the following vehicle is aggressive, in the worst case, it can accelerate with $a_{x,f,a} = 4$ m/s^2 to prevent ego vehicle from cutting in.

3.1.3 Safety Analysis and Motion Adjustment

We assume that the ego vehicle is safe when it is in the original lane at the very beginning. At every step during the lane changing, the ego vehicle has a safe evasion trajectory computed at the last step. As shown in Fig. 3.2, it has three options with decreasing preference: proceed to change lanes, hesitate around the current lateral position, or abort the lane changing and return back to the original lane. It analyzes the state after executing selected behavior for one time step. If it has a safe evasion trajectory following that, it can go ahead with the selected behavior; otherwise, it has to attempt a less preferred behavior. In summary, safety of the ego vehicle is ensured by only selecting the strategy with a following safe evasion trajectory, assuming that the aggressiveness assessment for the following vehicle is correct (if not, safety can only be ensured if the following

vehicle is always treated as aggressive).

The behavior of proceeding to change lanes is to directly follow the longitudinal and lateral accelerations computed using the neural network planners. For hesitating, the lateral acceleration is adjusted to diminish the velocity:

$$a_y = min(max(-v_y/\delta t, -a_{y,m}), a_{y,m}), \tag{3.2}$$

where $a_{y,m}$ is the absolute value of the maximal lateral acceleration.

Next we will analyze the safe evasion trajectory. The optimal lateral motion is to return back to the original lane as soon as possible. We define time $t = 0$ when the ego vehicle just starts taking the evasion trajectory. The centers of the original and the target lane are $y = 0$ and $y = w_l$, respectively. The width of a vehicle is denoted as w_v. The ego vehicle is completely in the original lane when $p_y \leq \frac{w_l - w_v}{2}$. The fastest lateral motion is that the ego vehicle has lateral acceleration $a_y = -a_{y,m}$ when $t \in [0, t_1]$ and then $a_y = a_{y,m}$ when $t \in [t_1, t_{y,f}]$, and finally it reaches the position $p_y = \frac{w_l - w_v}{2}$ with $v_y = 0$:

$$\begin{cases} p_{y,t_0} + v_{y,t_0} t_1 - \frac{a_{y,m} t_1^2}{2} + (v_{y,t_0} - a_{y,m} t_1)(t_{y,f} - t_1) + \frac{a_{y,m}(t_{y,f} - t_1)^2}{2} = \frac{w_l - w_v}{2} \\ v_{y,t_0} - a_{y,m} t_1 + a_{y,m}(t_{y,f} - t_1) = 0 \end{cases} \tag{3.3}$$

Here p_{y,t_0} and v_{y,t_0} are the lateral position and velocity of the ego vehicle when $t = 0$.

We assume that the ego vehicle E, leading vehicle L and following vehicle F all have the same maximal longitudinal acceleration $a_{x,a} = a_{x,l,a} = a_{x,f,a}$ and braking deceleration $a_{x,d} = a_{x,l,d} = a_{x,f,d}$. Let p_{x,t_0} and v_{x,t_0} represent the longitudinal position and velocity of the ego vehicle when $t = 0$, respectively. For simplicity of notation, we omit the subscript of t_0, and use $p_{x,l}$, $v_{x,l}$, $p_{x,f}$ and $v_{x,f}$ to denote the longitudinal position and velocity of the leading and following vehicles

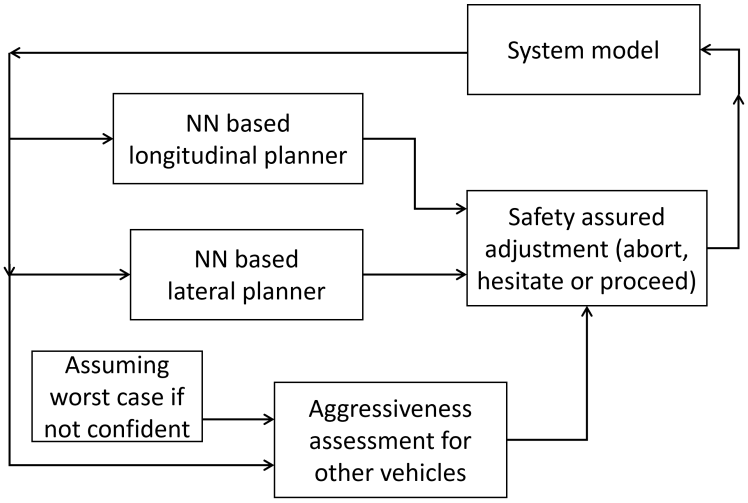

Figure 3.1: Design of our safety-driven neural network-based interactive planning framework. Safety-driven behavior adjustment module will adjust risky motions based on analyzing whether a safe evasion trajectory exists, considering the aggressiveness assessment of other vehicles.

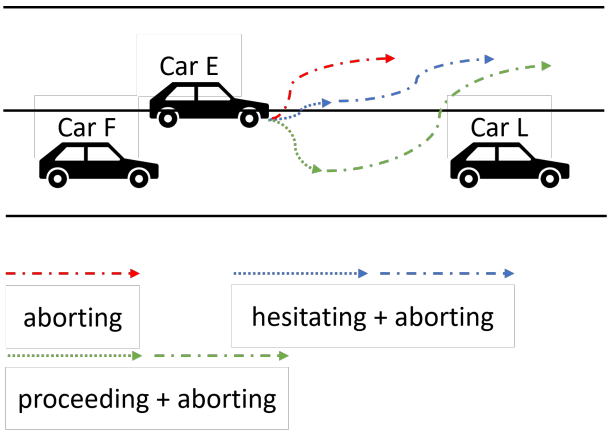

Figure 3.2: At every step in the lane changing process, the ego vehicle has three strategy choices. The first is proceeding to change lanes, in which case the short-term proceeding trajectory (green dotted line) and following complete aborting trajectory (green dash-dotted line) should be verified with safety guarantee. If the first strategy is not safe, the ego vehicle can hesitate around the current lateral position, and we need to verify safety for the short-term hesitating trajectory (blue dotted line) and following complete aborting trajectory (blue dash-dotted line). If both strategies do not work, the ego vehicle can directly abort lane changing behavior and go back to the original lane (red dash-dotted line), which is already verified to be safe in the last planning step.

when $t = 0$, respectively. Next we will temporarily neglect the following vehicle F, and analyze the longitudinal motion when the leading vehicle L decelerates with $a_{x,l,d}$ abruptly.

If $v_{x,l} \geq v_{x,t_0}$, then $\frac{v_{x,l}^2}{2a_{x,l,d}} \geq \frac{v_{x,t_0}^2}{2a_{x,d}}$, and the ego vehicle can prevent collisions with the leading vehicle if it decelerates with $a_{x,d}$. If $v_{x,l} < v_{x,t_0}$, the ego vehicle takes at least $t_{x,f} = \frac{v_{x,t_0}}{a_{x,d}}$ to fully stop. If $t_{x,f} \geq t_{y,f}$, ego vehicle only needs to keep enough headway when $t \in [0, t_{y,f}]$ because it is already not in the target lane when $t \in [t_{y,f}, t_{x,f}]$. We define C_1 to reflect the minimum headway[2]:

$$C_1 = \begin{cases} p_{x,t_0} - p_{x,l} + v_{x,t_0} t_{y,f} - \frac{a_{x,d} t_{y,f}^2}{2} - \frac{v_{x,l}^2}{2a_{x,l,d}} + p_m \\ \qquad \text{if } \frac{v_{x,l}}{a_{x,l,d}} < t_{y,f} \\ p_{x,t_0} - p_{x,l} + (v_{x,t_0} - v_{x,l}) t_{y,f} - \frac{(a_{x,d} - a_{x,l,d}) t_{y,f}^2}{2} + p_m \\ \qquad \text{otherwise} \end{cases} \tag{3.4}$$

Here p_m is the minimum gap between vehicles to avoid collisions. We need $C_1 < 0$ to ensure safety. If $t_{x,f} < t_{y,f}$, ego vehicle needs to keep enough headway when $t \in [0, t_{x,f}]$. Similarly, we define C_2 and need $C_2 < 0$ to ensure safety:

$$C_2 = p_{x,t_0} - p_{x,l} + \frac{v_{x,t_0}^2}{2a_{x,d}} - \frac{v_{x,l}^2}{2a_{x,l,d}} + p_m \tag{3.5}$$

Next we assume that at the same time, the following vehicle F accelerates to prevent the ego vehicle from cutting in. In such case, the ego vehicle can even accelerate and get closer to the leading vehicle, thus acquiring more time for lateral evasion before the following vehicle catches up. It is indeed the fastest longitudinal motion to prevent collision with the following vehicle by first accelerating with $a_{x,a}$, and then decelerating with $a_{x,d}$.

We assume that the ego vehicle accelerates with $a_x = a_{x,a}$ when $t \in [0, t_2]$ and then decelerates

[2]It is indeed a constant minus the minimum headway.

Table 3.2: Safety and performance evaluation for our proposed framework. From top to bottom, experimental settings correspond to more challenging lane changing scenarios. Our 'SafIn NN' planner results in zero collision rate in all simulations.

experimental settings	methods	lane changing time	final lateral position	success rate	collision rate
$-6 \leq a_{x,l} \leq 4$, $7 \leq \delta p \leq 37$	MPC	1.90 s	**3.44 m**	**92.61%**	7.39%
	only NN	**1.70 s**	3.25 m	89.59%	10.41%
	SafIn NN	1.90 s	2.73 m	80.31%	**0%**
$-6 \leq a_{x,l} \leq 0$, $7 \leq \delta p \leq 37$	MPC	1.90 s	**3.46 m**	**87.46%**	12.54%
	only NN	**1.68 s**	3.30 m	82.37%	17.63%
	SafIn NN	2.08 s	2.44 m	67.89%	**0%**
$-6 \leq a_{x,l} \leq 4$, $7 \leq \delta p \leq 17$	MPC	1.90 s	**3.44 m**	83.06%	16.94%
	only NN	**1.73 s**	3.24 m	**84.53%**	15.47%
	SafIn NN	1.97 s	2.23 m	61.51%	**0%**
$-6 \leq a_{x,l} \leq 0$, $7 \leq \delta p \leq 17$	MPC	1.90 s	**3.46 m**	71.82%	28.18%
	only NN	**1.71 s**	3.29 m	**74.32%**	25.68%
	SafIn NN	2.34 s	1.66 m	38.76%	**0%**

with $a_x = -a_{x,d}$ until it stops when $t \in [t_2, t_{y,f}]$. By letting the minimum distance between ego vehicle and leading vehicle be exactly p_m to remain safe, t_2 can be represented as a function of C_1 and C_2.

To prevent collisions with the following vehicle accelerating with $a_{x,f,a}$, we have

$$\begin{cases} p_{x,t_0} + v_{x,t_0} t_2 + \frac{a_{x,a} t_2^2}{2} + \frac{(v_{x,t_0} + a_{x,a} t_2)^2}{2 a_{x,d}} - p_{x,f} - v_{x,f} t_{y,f} - \frac{a_{x,f,a} t_{y,f}^2}{2} - p_m > 0 \\ \qquad \text{if } t_2 + \frac{v_{x,t_0} + a_{x,a} t_2}{a_{x,d}} < t_{y,f} \\ p_{x,t_0} + v_{x,t_0} t_{y,f} - \frac{a_{x,a} t_2^2}{2} + a_{x,a} t_2 t_{y,f} - \frac{a_{x,d} (t_{y,f} - t_2)^2}{2} - p_{x,f} - v_{x,f} t_{y,f} - \frac{a_{x,f,a} t_{y,f}^2}{2} - p_m > 0 \\ \qquad \text{otherwise} \end{cases} \tag{3.6}$$

Similar to Eq. 3.6, we can derive the inequality constraint for ensuring system safety when the following vehicle is cautious, willing to decelerate and create gap for ego vehicle.

In summary, if the constraints can be satisfied, the ego vehicle is verified to have safe evasion trajectory after taking planned trajectory; otherwise, it has to adjust the driving behavior to prevent

possible collisions.

3.2 Experimental Results

In this section, we first demonstrate the effectiveness of our proposed framework via simulations on synthetic and real-world examples. We compare our approach (denoted as 'SafIn NN') with a neural network-based planner without safety consideration or interactive planning (denoted as 'only NN') and an MPC-based planner from [47]. We then further evaluate the performance of our aggressiveness assessment module for the following vehicle in the target lane.

3.2.1 Evaluation with Synthetic Examples

We first evaluate the performance of our proposed framework through extensive simulations on synthetic examples, and the results are shown in Table 3.2. We have four classes with different experimental settings, which indicate the ranges that $a_{x,l}$ and δp will uniformly sample from. δp is the initial longitudinal distance between leading vehicle and ego vehicle. Thus $-6 \leq a_{x,l} \leq 4$ and $7 \leq \delta p \leq 37$ correspond to easier lane changing scenarios, while $-6 \leq a_{x,l} \leq 0$ and $7 \leq \delta p \leq 17$ correspond to more congested and dangerous scenarios. For each class, we conduct 200,000 simulations with randomly generated relative positions, velocities and IDM parameters. The following vehicle has 50% probability of being aggressive, and another 50% probability of being cautious.

For every round simulation with a horizon of 10 seconds, the ego vehicle attempts to change lanes until it collides with other vehicles. It is successful if the ego vehicle finally crosses the two lanes within the simulation horizon without any collision. For all successful lane changing simulations, we compute the average time that it takes to cross the lanes. For all safe simulations, we compute the average of final lateral positions. The lateral position $y = 3.5$ meters represents

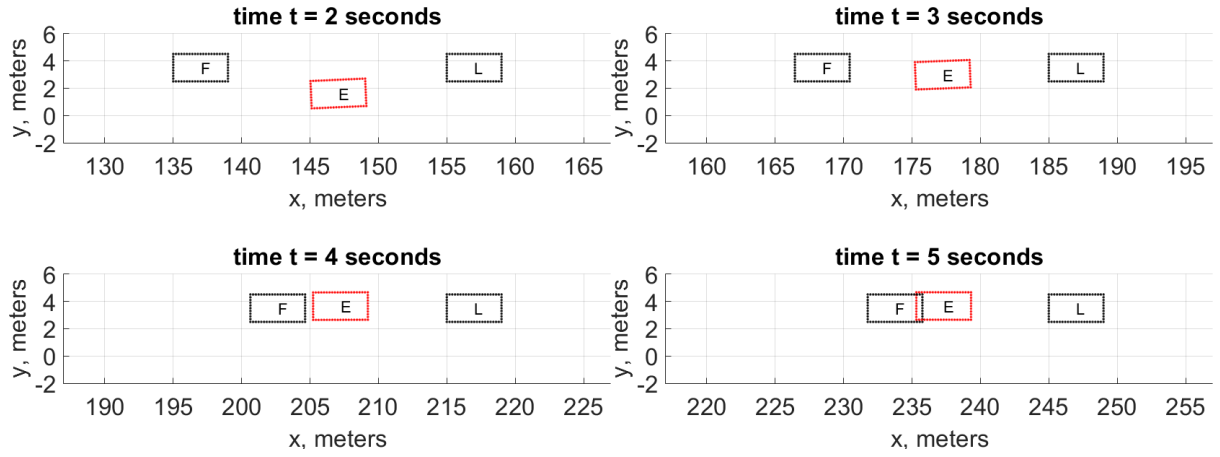

Figure 3.3: Illustrating Example. The x and y axes show the longitudinal and lateral positions of vehicles. Four subplots show the positions at different times. Red rectangle represents the ego vehicle, and black rectangles are surrounding vehicles. It corresponds to the scenario that initial velocity is 30 m/s for all vehicles, and the distance between the leading vehicle and the following vehicle is 20 m. The following vehicle accelerates at $t = 2$ seconds to prevent the ego vehicle from cutting in. The ego vehicle is controlled by the 'only NN' planner.

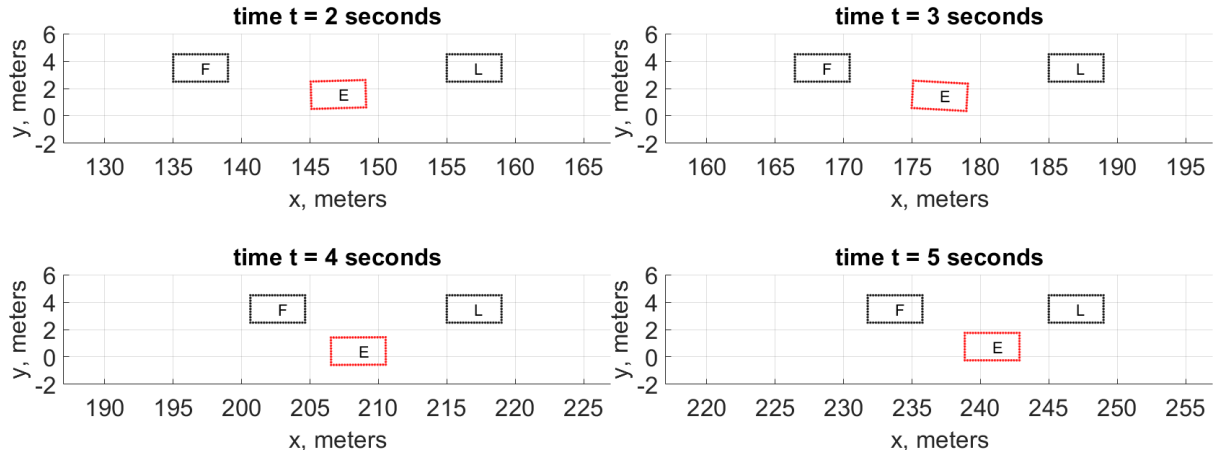

Figure 3.4: Illustrating Example. The x and y axes show the longitudinal and lateral positions of vehicles. Four subplots show the positions at different times. Red rectangle represents the ego vehicle, and black rectangles are surrounding vehicles. It corresponds to the same scenario as in Fig. 3.3, except that the ego vehicle is controlled by our 'SafIn NN' planner.

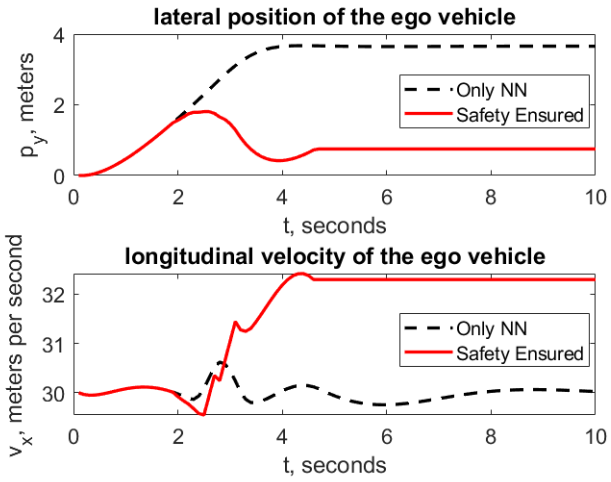

Figure 3.5: Lateral position and longitudinal velocity of the ego vehicle in the same scenario as in Fig. 3.3.

the center of target lane, and $y = 1.75$ meters is the border of two lanes.

Table 3.2 shows that: (1) Our approach 'SafIn NN' results in **zero collision rate in all simulations regardless whether the following vehicle is aggressive or not**, while 'MPC' and 'only NN' both lead to significant collision rate, especially in more challenging scenarios. (2) Under 'MPC' and 'only NN', the ego vehicle has a higher lane changing success rate, less lane changing time and larger final lateral position. In easier scenarios, these advantages are relatively small. When it becomes more congested and challenging, risky behaviors are restricted under our 'SafIn NN' planner while lead to higher collision rate for 'MPC' and 'only NN'.

We further demonstrate the strength of our 'SafIn NN' planner with a concrete example. The initial longitudinal velocities of all three vehicles are set to 30 meters per second. The distance between leading and following vehicle is 20 meters. Then we let the following vehicle start accelerating with $a_{x,f} = 4$ meters per second squared until $p_{x,l} - p_{x_f} \leq 17$ meters at $t = 2$ seconds. Then it adjusts velocity and attempts to maintain the distance to the leading vehicle. Fig. 3.3 and 3.4 present the relative position changes of vehicles when the ego vehicle is controlled by 'only NN'

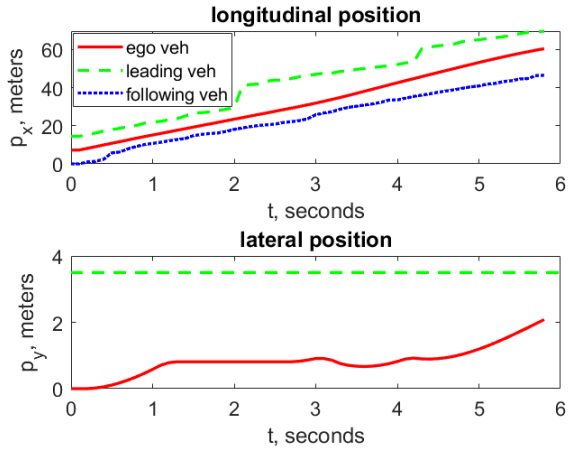

Figure 3.6: Longitudinal and lateral position of the ego vehicle, leading vehicle and following vehicle. It shows an example of challenging scenario in dataset collected by Pony.ai and the ego vehicle is controlled by the 'SafIn NN' planner.

and 'SafIn NN' planners, respectively. Under the 'only NN' planner, the ego vehicle completes lane changing at $t = 3$ seconds, but collides with the following vehicle at $t = 5$ seconds. Our 'SafIn NN' planner prevents the collision proactively. As shown in Fig. 3.4, the ego vehicle aborts changing lanes at around $t = 3$ seconds, and then hesitates around $y = 1$ meters and looks for the next chance to change lanes. Fig. 3.5 shows the lateral position and longitudinal velocity of the ego vehicle in the simulation horizon.

3.2.2 Evaluation with Real-world Challenging Dataset

We further evaluate our approach in challenging scenarios with real-world dataset collected by Pony.ai, an autonomous vehicle company. The dataset provides road geometry and motion information of surrounding traffic participants in congested scenarios with industry-level accuracy. Under our designed planner, the ego vehicle is controlled to change lanes. **In all tested 48 real-world challenging scenarios, the ego vehicle can always remain safe under our planner** during the lane change process, despite that our planner is never trained or optimized with the dataset. Under

Table 3.3: Performance of aggressiveness assessment.

	$a_{th} = 0$	$a_{th} = 0.15$	$a_{th} = 0.25$	$a_{th} = 0.5$	$a_{th} = 1$
easy					
uncertain rate	0%	2.28%	4.05%	9.16%	19.3%
error rate	4.61%	3.6%	3.05%	2.01%	0.91%
medium					
uncertain rate	0%	18.38%	31.33%	56.32%	79.65%
error rate	36.73%	28.82%	24.53%	15.87%	7.16%
hard					
uncertain rate	0%	65.26%	74.16%	85.57%	94.39%
error rate	49.76%	17.18%	12.76%	7.1%	2.74%

'only NN' planner, there are 12 scenarios in which the ego vehicle collides with other vehicles.

Fig. 3.6 shows a concrete example of the challenging scenario that the gap between the leading vehicle and the following vehicle is decreasing initially. The ego vehicle under our designed planner attempts to change lane at the beginning and then hesitates around $p_y = 0.8$ meters from $t = 1.1$ seconds, and finally continues changing lane from $t = 4.5$ seconds. While under the 'only NN' planner, the ego vehicle will collide with the following vehicle at $t = 1.6$ seconds in this challenging scenario.

3.2.3 Evaluation of Aggressiveness Assessment

We conduct experiments to evaluate the performance of our aggressive assessment module, and Table 3.3 shows the results. We classify all simulation entries in the dataset into three classes based on the difference of accelerations under different behaviors, $\delta a^*_{x,f} = |a^*_{x,f,1} - a^*_{x,f,2}|$. It is classified as easy, medium or hard if $\delta a^*_{x,f} > 0.5$, $0.25 < \delta a^*_{x,f} \leq 0.5$, or $\delta a^*_{x,f} \leq 0.25$, respectively. We conduct sensitivity analysis over the threshold a_{th}. It shows that with larger a_{th}, the uncertain rate is higher and the error rate is lower for all three different difficulty levels. It meets our expectation because a larger a_{th} results in a more robust and conservative predictor, which is prone to be uncertain when it is less confident. It also presents that for easy cases, the performance

can be considerably greater because a larger $\delta a^*_{x,f}$ means that it is more distinguishable.

For simulations conducted in Section 3.2.1, we use $a_{th} = 0.5$. Although it has a positive error rate, our overall approach is quite robust and does not result in collisions in all experiments. We think that there are two reasons: (1) aggressiveness assessment is conducted every 0.1 seconds along with other modules, and thus occasional mis-prediction is highly likely to be corrected later; and (2) it is more challenging to make correct assessment when the following vehicle is far away from the ego vehicle. However, in those cases, incorrect assessment is less critical because of the large gap between vehicles.

3.2.4 Discussion on MPC and Neural Network-based Planners

In this work, the neural network planners are learned from the synthesized data of the system under MPC, and thus 'only NN' has similar performance as MPC – albeit in more challenging scenarios, 'only NN' shows slight advantages in both success rate and collision rate. Moreover, MPC with the safety-driven behavior adjustment module also provide similar performance as our 'SafIn NN' planner. However, note that our safety-driven interactive planning framework can be incorporated with any state-of-the-art neural network-based planners to improve safety. We believe that with more high-quality training data, 'SafIn NN' can also significantly improve its performance in success rate and may perform better than MPC-based planners in all metrics, especially when system dynamics and interactions are hard to model (in this work the MPC is assumed to have perfect system model).

3.3 Connectivity-enhanced Planner Design

By leveraging the connectivity technology, our planner design can further improve system efficiency while ensuring system safety at the same time. In this general framework, we assume that

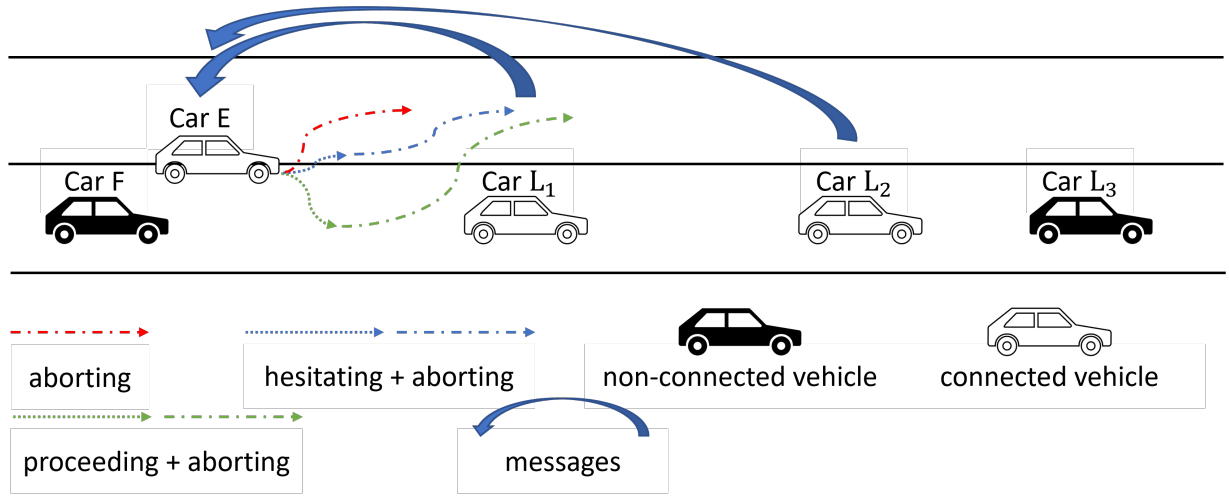

Figure 3.7: The ego vehicle E intends to change lane. With connectivity technology, vehicle E receives planned acceleration profiles and real-time motion states from connected leading vehicles L_i, $1 \leq i \leq N$, and then analyzes the maximum deceleration of vehicle L_1 during the lane changing process. By identifying the behavior of following vehicle F and analyzing system safety in the worst case, vehicle E may proceed to change lane, hesitate around the current lateral position, or abort the lane changing plan. This figure shows an example with $N = 2$ and the following vehicle F is non-connected.

the ego vehicle is connected and autonomous, while surrounding vehicles can be either connected or non-connected, and either autonomous or human-driven[3]. Figure 3.7 shows an example of the lane changing scenario. The connected ego vehicle E intends to change to the target lane. From the neighbor area to the traffic downstream, there are a set of connected leading vehicles L_1, L_2, $\cdots$, L_N and one non-connected leading vehicle L_{N+1} in the target lane. In this work, we are considering all possible scenarios with varied type and number of surrounding vehicles in our planner design. Depending on the dynamic mixed traffic environment, the number of connected leading vehicles N may vary[4]. The following vehicle F in the target lane can be either connected or non-connected.

[3]For connected human-driven vehicles, we assume that a larger inter-vehicle distance needs to be kept to ensure safety, as human-driven vehicles can have larger execution error compared with autonomous vehicles.

[4]By $N = 0$, it represents that the immediate leading vehicle is non-connected.

To ensure safety, we verify the existence of a safe evasion trajectory following the planned trajectory in the worst case. We analyze the worst case for the ego vehicle in dynamical environments. For the case without a connected leading vehicle, i.e., $N = 0$, we can directly assume that in the worst case, vehicle L_1 takes the maximum deceleration under the mechanical constraints. In other cases that $N > 0$, we can leverage the coordination from connected vehicles to prevent overly conservative planning. We evaluate the fastest evasion trajectory of the ego vehicle in emergency situations. For instance, when there is an emergency brake in the downstream, connected vehicles that first realize the event can collaboratively take a smaller deceleration, if it is safe for themselves, thus leaving more reaction time for the ego vehicle. If the evasion trajectory can prevent collisions, the ego vehicle can proceed to change lanes under the neural network-based planners, otherwise it has to hesitate around the boundary of the two lanes or return back to the original lane for safety.

The planning framework is presented in Figure 3.8. Based on the planned acceleration profiles in the planning horizon and the real-time motion states of surrounding vehicles, we can leverage neural networks for longitudinal and lateral trajectory planning. At the same time, we can derive the maximum deceleration of the leading vehicle L_1 (scenario as shown in Figure 3.7), and then perform system analysis for the worst case and adjust trajectory to ensure safety. Here we adopt the same aggressiveness assessment method for the following vehicle F as in Section 3.1 when vehicle F is non-connected. For the case that the following vehicle F is connected, we assume that it is collaborative.

Safety analysis and trajectory adjustment are conducted periodically. At every step during the lane changing, the ego vehicle has three behavior-level options with strictly decreasing preference: proceed changing lane, hesitate around current lateral position, or abort changing lane and return back to the original lane. It analyzes the state after executing the accelerations under neural network-based planners for one time step. If it has a safe evasion trajectory in the worst case, it

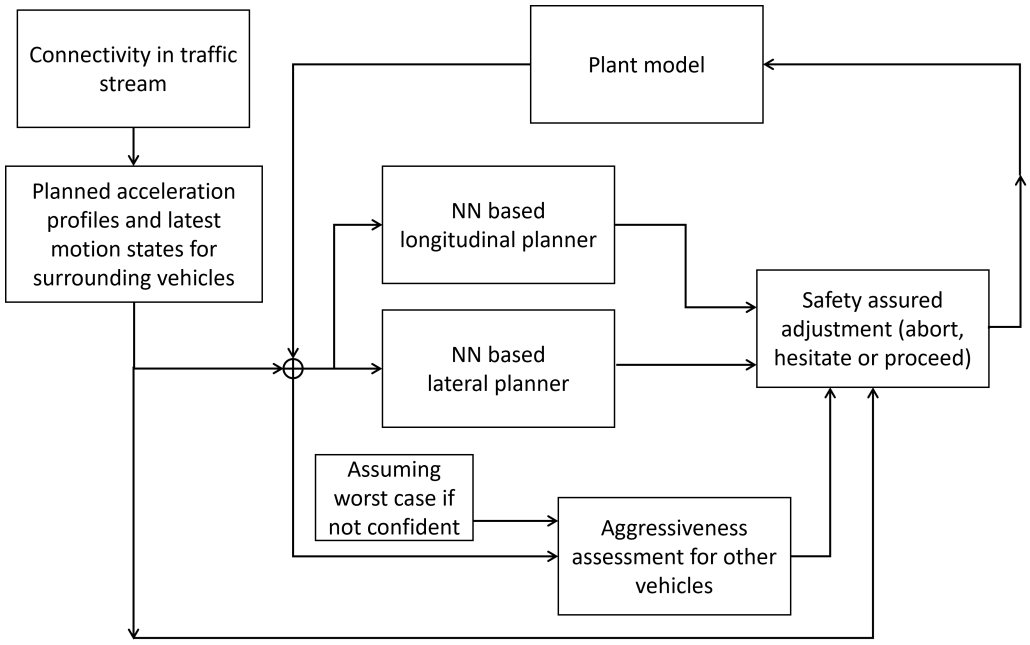

Figure 3.8: By incorporating the planned acceleration profiles and the real-time motion states of surrounding vehicles updated by connectivity technology, we can further improve the performance of neural network-based lane changing planner. At the same time, we can derive the maximum deceleration of the leading vehicle L_1 (scenario as shown in Figure 3.7), perform aggressiveness assessment and system analysis for the worst case, and adjust trajectory to ensure safety.

can go ahead and change lanes. Otherwise, it has to attempt a less preferred behavior until a safe evasion trajectory is found following that.

3.3.1 Connectivity Assumptions

In this work, we assume that the connected leading vehicles L_i, $1 \leq i \leq N$ (for example, $N = 2$ in the scenario from Figure 3.7) will collaboratively assist the lane changing process of the ego vehicle E and prevent collision with their own immediate leading vehicle L_{i+1}. Specifically, the connected leading vehicle L_i will keep its acceleration within the range $[-a_i^{m,d}, a_i^{m,a}]$ to execute its own driving task if that is sufficient for keeping a safe distance between L_i and L_{i+1}. It is noted that the values of $a_i^{m,d}$ and $a_i^{m,a}$ will be communicated to other connected vehicles. Only in

emerging scenarios, e.g., when the leading vehicle L_{i+1} decelerates suddenly, L_i can violate such *'promise'* and take a deceleration $a_{x,i,d} > a_i^{m,d}$ to ensure safety. Note that this is considered as an *expected violation* of the promise, and our planner framework can ensure safety in such cases (later in Section 3.4, we will demonstrate the impact when such promise is violated *unexpectedly*, e.g., when one surrounding vehicle is malfunctioning or influenced by other unknown obstacles).

To generalize the model, we assume that both connected human-driven vehicles and connected autonomous vehicles communicate their planned acceleration range to other vehicles in the same manner. However, human-driven vehicles could have larger acceleration range because there is typically larger uncertainty in human driver's behavior and execution process. For non-connected vehicles, we assume that it can take any acceleration value within their mechanical constraints.

3.3.2 Safety Analysis

Next we will analyze the evasion trajectory given the system states. Assuming that vehicle L_i decelerates with $a_{x,i,d}$, vehicle L_{i-1} needs to decelerate with $z_{x,i-1,d}$ to prevent collisions. By letting p_m denote the minimum safe distance between two vehicles, we have

$$\begin{cases} p_{x,i} + v_{x,i} t_{L_i} + \frac{a_{x,i,d} t_{L_i}^2}{2} - p_{x,i-1} - v_{x,i-1} t_{L_i} - \frac{z_{x,i-1,d} t_{L_i}^2}{2} - p_m = 0 \\ \qquad \text{if } v_{x,i} < v_{x,i-1} \text{ and } \frac{v_{x,i}}{a_{x,i,d}} \geq \frac{v_{x,i-1}}{z_{x,i-1,d}}, \\ p_{x,i} + \frac{v_{x,i}^2}{2a_{x,i,d}} - p_{x,i-1} - \frac{v_{x,i-1}^2}{2z_{x,i-1,d}} - p_m = 0 \\ \qquad \text{otherwise}, \end{cases} \tag{3.7}$$

where $p_{x,i}$, $v_{x,i}$, $p_{x,i-1}$ and $v_{x,i-1}$ denote the position and velocity of vehicle L_i and L_{i-1}, respectively. The first equation corresponds to the case that two vehicles can have the same velocity at $t_{L_i} = \frac{v_{x,i-1} - v_{x,i}}{z_{x,i-1,d} - a_{x,i,d}}$ and reach minimum distance gap at the same time. The second equation

corresponds to the case that L_{i-1} is the closest to L_i when it stops.

Then we can obtain the deceleration of vehicle L_{i-1} by solving Equation (3.7)

$$z_{x,i-1,d} = \begin{cases} \frac{3(v_{x,i-1}-v_{x,i})^2}{2(p_{x,i}-p_{x,i-1}-p_m)} + a_{x,i,d} \\ \qquad \text{if } v_{x,i} < v_{x,i-1} \text{ and } \frac{v_{x,i}}{a_{x,i,d}} \geq \frac{v_{x,i-1}}{z_{x,i-1,d}}, \\ \frac{v_{x,i-1}^2}{2(p_{x,i}-p_{x,i-1}-p_m)+\frac{v_{x,i}^2}{a_{x,i,d}}} \\ \qquad \text{otherwise.} \end{cases} \tag{3.8}$$

Considering that vehicle L_{i-1} has claimed in a message that it will take an acceleration in range $[-a_{i-1}^{m,d}, a_{i-1}^{m,a}]$ to execute its own driving task, we then have the deceleration of vehicle L_{i-1} in the worst case

$$a_{x,i-1,d} = max(a_{i-1}^{m,d}, z_{x,i-1,d}). \tag{3.9}$$

Let $a_{x,d}$ and $a_{x,a}$ denote the maximal longitudinal deceleration and acceleration of a vehicle under mechanical constraints. Let $a_{y,m}$ denote the absolute value of the maximal lateral acceleration under mechanical constraints. In the worst case, we assume that the non-connected leading vehicle L_{N+1} can decelerate with $a_{x,N+1,d} = a_{x,d}$, and then we can obtain $a_{x,N,d}$ with Equations (3.8) and (3.9). Similarly, we can compute the decelerations $a_{x,N-1,d}$, $a_{x,N-2,d}$, $\cdots$, $a_{x,1,d}$ for the other leading vehicles in the worst case.

Given that the leading vehicle L_1 decelerates with $a_{x,1,d}$, its position at time t is

$$p_{x,1,t} = \begin{cases} p_{x,1} + v_{x,1}t - \frac{a_{x,1,d}t^2}{2} \\ \qquad\qquad \text{if } t \leq \frac{v_{x,1}}{a_{x,1,d}}, \\ p_{x,1} + \frac{v_{x,1}^2}{2a_{x,1,d}} \\ \qquad\qquad \text{otherwise.} \end{cases} \tag{3.10}$$

Next we will analyze the evasion trajectory for the ego vehicle E in the worst case. If it is safe, the ego vehicle E can proceed to change lane under the neural network-based planners, otherwise it can adapt behavior and trajectory to ensure safety. For the lateral motion, the fastest evasion trajectory is to accelerate with $a_y = -a_{y,m}$ when $t \in [0, t_1]$ and then $a_y = a_{y,m}$ when $t \in [t_1, t_{y,f}]$. It can be obtained by solving the equation below.

$$\begin{cases} p_{y,t_0} + v_{y,t_0}t_1 - \frac{a_{y,m}t_1^2}{2} + (v_{y,t_0} - a_{y,m}t_1)(t_{y,f} - t_1) + \frac{a_{y,m}(t_{y,f}-t_1)^2}{2} = \frac{w_l - w_v}{2}, \\ v_{y,t_0} - a_{y,m}t_1 + a_{y,m}(t_{y,f} - t_1) = 0, \end{cases} \tag{3.11}$$

where p_{y,t_0} and v_{y,t_0} are the lateral position and velocity of the ego vehicle when $t = 0$. The centers of the original and target lane are $y = 0$ and $y = w_l$, respectively. The width of a vehicle is w_v.

As for the longitudinal motion, the optimal trajectory is analyzed intuitively when the immediate leading vehicle L_1 decelerates with $a_{x,d}$ in Section 3.1. It is that the ego vehicle first accelerates with $a_{x,a}$ and then decelerates with $a_{x,d}$, and keeps the distance gap no smaller than p_m when $t \in [0, t_{y,f}]$. It is the fastest longitudinal motion to get closer to the leading vehicle, thus obtaining more time for the lateral evasion before the following vehicle catches up.

However, in this section, it is a more general situation that the leading vehicle L_1 can have

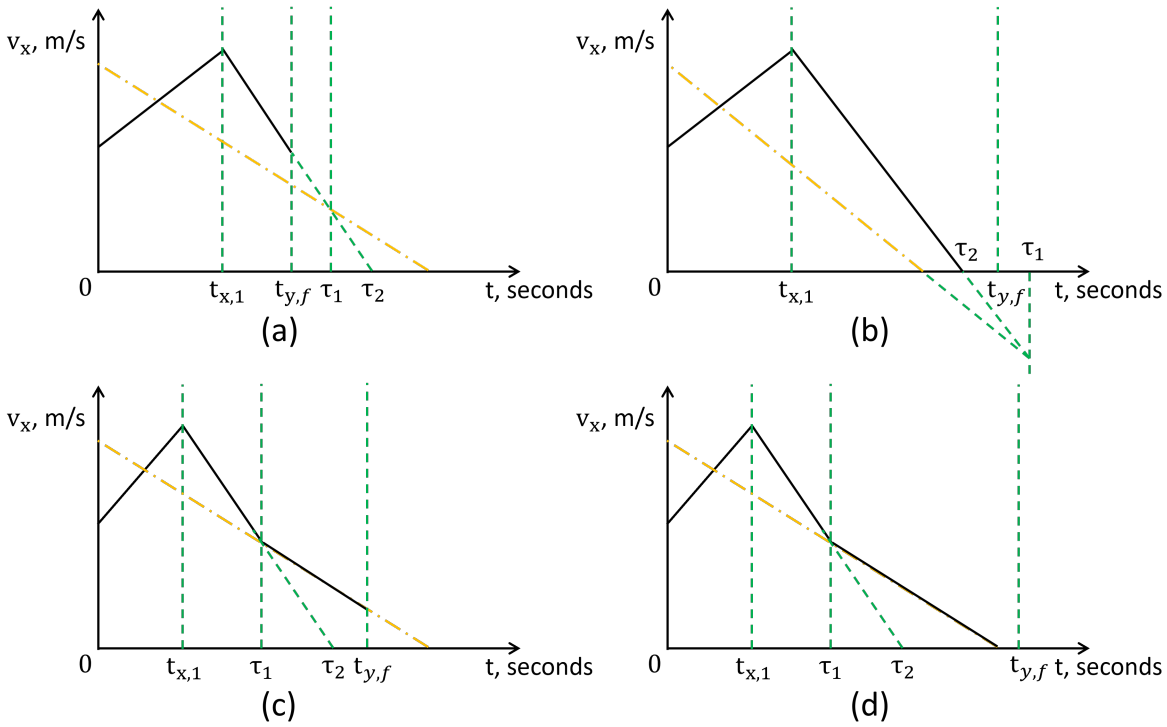

Figure 3.9: Depending on the initial states and $t_{y,f}$, there are four cases of longitudinal evasion trajectory for the ego vehicle E, which are presented in four sub-figures. The black solid line and yellow dot dash line represent velocity curves of the ego vehicle E and the leading vehicle L_1, respectively. (a) represents that the ego vehicle has already finished its lateral motion before it decelerates to the same velocity with vehicle L_1 and changes its deceleration. (b) is that the ego vehicle decelerates with $a_{x,d}$ until it stops, and then keeps $v_x = 0$ until $t = t_{y,f}$. (c) is that the ego vehicle has already finished its lateral motion before it reaches $v_x = 0$ with deceleration $a_{x,1,d}$. (d) represents that the ego vehicle decelerates with $a_{x,1,d}$ until it stops, and then keeps $v_x = 0$ until $t = t_{y,f}$.

deceleration $a_{x,1,d} \leq a_{x,d}$. Thus when the ego vehicle E decelerates to the same velocity with the leading vehicle L_1, its deceleration should change from $a_{x,d}$ to $a_{x,1,d}$. Otherwise the headway of the ego vehicle will increase when its velocity is smaller than that of the vehicle L_1. In that way, the ego vehicle is overreacting for collision avoidance, which cannot lead to an optimal trajectory.

There are three phases in the optimal longitudinal motion. The ego vehicle first accelerates with $a_x = a_{x,a}$ when $t \in [0, t_{x,1}]$, and then decelerates with $a_x = -a_{x,d}$ when $t \in [t_{x,1}, t_{x,2}]$,

and then decelerates with $a_x = -a_{x,1,d}$ until it stops when $t \in [t_{x,2}, t_{y,f}]$. The position of the ego vehicle when $t = t_{y,f}$ is formulated as

$$
p_{x,t_{y,f}} = \begin{cases}
p_{x,t_0} + v_{x,t_0} t_{x,1} + \frac{a_{x,a} t_{x,1}^2}{2} + (v_{x,t_0} + a_{x,a} t_{x,1})(t_{y,f} - t_{x,1}) - \frac{a_{x,d}(t_{y,f} - t_{x,1})^2}{2} \\
\qquad \text{if } v_{x,t_0} + a_{x,a} t_{x,1} - a_{x,d}(t_{y,f} - t_{x,1}) \geq max(0, v_{x,1} - a_{x,1,d} t_{y,f}), \\
p_{x,t_0} + v_{x,t_0} t_{x,1} + \frac{a_{x,a} t_{x,1}^2}{2} + \frac{(v_{x,t_0} + a_{x,a} t_{x,1})^2}{2 a_{x,d}} \\
\qquad \text{else if } \frac{v_{x,1}}{a_{x,1,d}} \leq \frac{v_{x,t_0} + a_{x,a} t_{x,1}}{a_{x,d}} + t_{x,1} \leq t_{y,f}, \\
p_{x,t_0} + v_{x,t_0} t_{x,2} + \frac{a_{x,a} t_{x,1}^2}{2} + a_{x,a} t_{x,1}(t_{x,2} - t_{x,1}) - \frac{a_{x,d}(t_{x,2} - t_{x,1})^2}{2} - \frac{a_{x,1,d}(t_{y,f} - t_{x,2})^2}{2} \\
\qquad + (v_{x,t_0} + a_{x,a} t_{x,1} - a_{x,d}(t_{x,2} - t_{x,1}))(t_{y,f} - t_{x,2}) \\
\qquad \text{else if } \frac{v_{x,1}}{a_{x,1,d}} \geq t_{y,f}, \\
p_{x,t_0} + v_{x,t_0} t_{x,2} + \frac{a_{x,a} t_{x,1}^2}{2} + a_{x,a} t_{x,1}(t_{x,2} - t_{x,1}) - \frac{a_{x,d}(t_{x,2} - t_{x,1})^2}{2} \\
\qquad + \frac{(v_{x,t_0} + a_{x,a} t_{x,1} - a_{x,d}(t_{x,2} - t_{x,1}))^2}{2 a_{x,1,d}} \\
\qquad \text{otherwise},
\end{cases}
\tag{3.12}
$$

where p_{x,t_0} and v_{x,t_0} are the longitudinal position and the velocity of the ego vehicle when $t = 0$, respectively.

Depending on the initial states and $t_{y,f}$, there are four cases in Equation (3.12), which correspond to the four velocity curves in Figure 3.9. The black solid line and yellow dot dash line represent velocity curves of the ego vehicle E and the leading vehicle L_1, respectively. The first case represents that the ego vehicle has already finished its lateral motion before it decelerates to the same velocity with vehicle L_1 and changes its deceleration. The second case is that the ego vehicle decelerates with $a_{x,d}$ until it stops, and then keeps $v_x = 0$ until $t = t_{y,f}$. The third case is that

the ego vehicle has already finished its lateral motion before it reaches $v_x = 0$ with deceleration $a_{x,1,d}$. The fourth case represents that the ego vehicle decelerates with $a_{x,1,d}$ until it stops, and then keeps $v_x = 0$ until $t = t_{y,f}$.

Given that the two vehicles reach the same velocity when $t = t_{x,2}$, we have $v_{x,t_0} + a_{x,a}t_{x,1} - a_{x,d}(t_{x,2} - t_{x,1}) = v_{x,1} - a_{x,1,d}t_{x,2}$. In all four cases, the distance gap between the ego vehicle E and the leading vehicle L_1 reaches the minimum when $t = t_{y,f}$. By letting $p_{x,1,t_{y,f}} - p_{x,t_{y,f}} - p_m = 0$, we can compute the value of $t_{x,1}$ as

$$
t_{x,1} = \begin{cases} t_{y,f} - \sqrt{t_{y,f}^2 + \frac{2v_{x,t_0}t_{y,f} - a_{x,d}t_{y,f}^2 + 2p_{x,t_0} + 2p_m - 2p_{x,1,t_{y,f}}}{a_{x,a} + a_{x,d}}} \\ \qquad \text{if } v_{x,t_0} + a_{x,a}t_{x,1} - a_{x,d}(t_{y,f} - t_{x,1}) \geq max(0, v_{x,1} - a_{x,1,d}t_{y,f}), \\ -\frac{v_{x,t_0}}{a_{x,a}} + \sqrt{(\frac{v_{x,t_0}}{a_{x,a}})^2 - \frac{2p_{x,t_0}a_{x,d} + v_{x,t_0}^2 + 2p_m a_{x,d} - 2p_{x,1,t_{y,f}}a_{x,d}}{(a_{x,a} + a_{x,d})a_{x,a}}} \\ \qquad \text{else if } \frac{v_{x,1}}{a_{x,1,d}} \leq \frac{v_{x,t_0} + a_{x,a}t_{x,1}}{a_{x,d}} + t_{x,1} \leq t_{y,f}, \\ \frac{-(v_{x,t_0} - v_{x,1}) + \sqrt{(v_{x,t_0} - v_{x,1})^2 - 2(a_{x,a} + a_{x,1,d})C_2}}{a_{x,a} + a_{x,1,d}} \\ \qquad \text{else if } \frac{v_{x,1}}{a_{x,1,d}} \geq t_{y,f}, \\ \frac{-(v_{x,t_0} - v_{x,1}) + \sqrt{(v_{x,t_0} - v_{x,1})^2 - 2(a_{x,a} + a_{x,1,d})C_2}}{a_{x,a} + a_{x,1,d}} \\ \qquad \text{otherwise}, \end{cases} \tag{3.13}
$$

where $C_2 = \frac{(v_{x,t_0} - v_{x,1})^2 + (2p_{x,t_0} + 2p_m - 2p_{x,1})(a_{x,d} - a_{x,1,d})}{2(a_{x,a} + a_{x,d})}$.

It is noted that if $t_{y,f}^2 + \frac{2v_{x,t_0}t_{y,f} - a_{x,d}t_{y,f}^2 + 2p_{x,t_0} + 2p_m - 2p_{x,1,t_{y,f}}}{a_{x,a} + a_{x,d}} \leq 0$, the ego vehicle can keep accelerating until $t = t_{y,f}$, i.e., $t_{x,1} = t_{y,f}$, and remain safe. If $2v_{x,t_0}t_{y,f} - a_{x,d}t_{y,f}^2 + 2p_{x,t_0} + 2p_m - 2p_{x,1,t_{y,f}} > 0$ in the first case, or $2p_{x,t_0}a_{x,d} + v_{x,t_0}^2 + 2p_m a_{x,d} - 2p_{x,1,t_{y,f}}a_{x,d} > 0$ in the second case, or $C_2 > 0$ in the third and fourth cases, we have $t_{x,1} < 0$, which means the ego vehicle cannot

prevent collisions even when it keeps decelerating with $a_{x,d}$ from $t = 0$, which means that the safe evasion trajectory does not exist.

Let τ_1 denote the time that the ego vehicle decelerates to the same velocity with the leading vehicle, and let τ_2 denote the time that the ego vehicle longitudinally decelerates to a velocity of zero. We present τ_1 and τ_2 in all four cases in Figure 3.9. Based on the value of $t_{x,1}$, we can obtain the position of the ego vehicle at time $t \in [0, t_{y,f}]$ as

$$p_{x,t} = \begin{cases} p_{x,t_0} + v_{x,t_0} t + \frac{a_{x,a} t^2}{2} \\ \qquad \text{if } t \in [0, t_{x,1}], \\ p_{x,t_0} + v_{x,t_0} t_{x,1} + \frac{a_{x,a} t_{x,1}^2}{2} + (v_{x,t_0} + a_{x,a} t_{x,1})(t - t_{x,1}) - \frac{a_{x,d}(t - t_{x,1})^2}{2} \\ \qquad \text{else if } t \in [t_{x,1}, min(\tau_1, \tau_2, t_{y,f})], \\ p_{x,1,t} - p_m \\ \qquad \text{otherwise.} \end{cases} \tag{3.14}$$

Assuming that the following vehicle accelerates with $a_{x,f} \in [-a_{x,d}, a_{x,a}]$, its position at t is

$$p_{x,f,t} = \begin{cases} p_{x,f} + v_{x,f} t + \frac{a_{x,f} t^2}{2} \\ \qquad \text{if } v_{x,f} + a_{x,f} t \geq 0, \\ p_{x,f} + \frac{v_{x,f}^2}{2|a_{x,f}|} \\ \qquad \text{otherwise.} \end{cases} \tag{3.15}$$

In this work, we make the same assumptions as in Section 3.1: if the following vehicle is collaborative and willing to create gap for ego vehicle, it can at least decelerate with $a_{x,f} = -a_{x,d}$; if

the following vehicle is aggressive, in the worst case, it can accelerate with $a_{x,f} = a_{x,a}$ to prevent the ego vehicle from cutting in. If $p_{x,t} - p_{x,f,t} \geq p_m$, $\forall t \in [0, t_{y,f}]$, the safe evasion trajectory exists, and is formulated in Equation (3.14). If it exists, the ego vehicle can go ahead and execute the planned trajectory, otherwise, it has to take a less preferred behavior, e.g., hesitate around the boundary of the two lanes or return back to the original lane for safety. It is noted that returning back to the original lane, which is the evasion trajectory computed in last control step, is already verified to be safe and feasible.

3.4 Experimental Results

3.4.1 Effectiveness of Our Framework Under Perfect Coordination

In this section, we first demonstrate the statistical performance of our connectivity-enhanced planner framework, and then present an example with detailed trajectories. We compare system performance under a few different planners to demonstrate the effectiveness of our approach: 'CV_all' represents the complete planner framework as in Figure 3.8, in which we leverage information shared by all connected vehicles; 'CV_follow' means that we only use information shared by the following connected vehicle F; 'CV_none' means that connectivity technology is not leveraged, which is the planner in Section 3.1 and can be viewed as our main baseline; 'No_agg_assess' means that we conservatively assume following vehicle is always aggressive and disable the aggressiveness assessment function.

In this work, we use synthesised data to train longitudinal and lateral planners, similarly as in Section 3.1. We conduct extensive simulations by uniformly sampling with $v_{x,t_0} \in [29, 31]$ meters per second and $\delta p \in [17, 22]$ meters, and setting $a_i^{m,d} = 0.5$ meters per second squared, $v_{x,i} = 30$ meters per second, $i = 1, \cdots, N$. Here, δp denotes the initial longitudinal inter-vehicle distance.

Figure 3.10 presents the average lane changing success rate under different planners when the

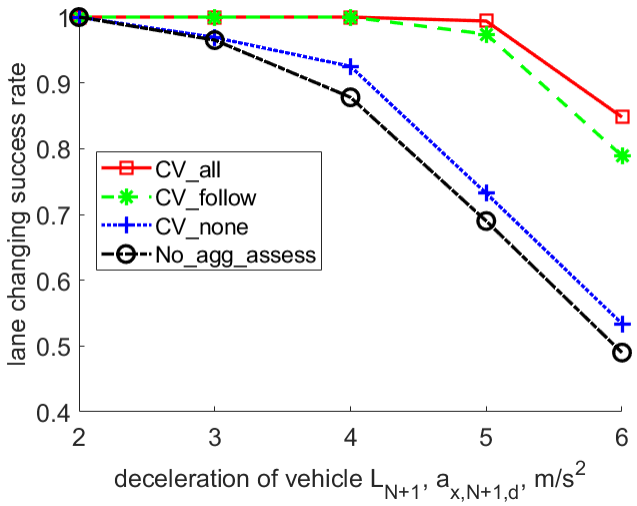

Figure 3.10: Lane changing success rate under different planners are compared when the number of connected leading vehicles is $N = 5$. The horizontal axes show the sudden deceleration of non-connected leading vehicle L_{N+1}.

number of connected leading vehicles is $N = 5$. The horizontal axes show the sudden deceleration of non-connected leading vehicle L_{N+1}. It is considered to be successful if the ego vehicle finally crosses the border of two lanes within the simulation horizon without any collision. Because safety is ensured by the trajectory adjustment function and there is indeed no collision in all simulations, we only present the results of lane changing success rate.

As we expected, lane changing success rate decreases when the deceleration $a_{x,N+1,d}$ gets larger. Under these planners, 'CV_all', 'CV_follow' and 'CV_none' correspond to full, partial and none utilization of connectivity, respectively. 'No_agg_assess' represents that when even aggressiveness assessment function is disabled, which leverages less information and is more conservative. It shows that 'CV_all' performs slightly better than 'CV_follow', and these two planners result in considerably larger success rate, when compared to 'CV_none' and 'No_agg_assess'. **This clearly shows the effectiveness of our approach in improving system performance.** It means that understanding the following vehicle's intention can greatly help the lane changing maneuver

Table 3.4: Lane changing success rate of different planners.

N	$a_{x,N+1,d}$	CV_all	CV_follow	CV_none	No_agg_assess
1	2	1	1	0.995	0.995
	3	1	1	0.875	0.83
	4	0.356	0.337	0.23	0.195
	5	0.001	0	0	0
	6	0	0	0	0
3	2	1	1	1	1
	3	1	1	0.928	0.894
	4	0.998	0.982	0.712	0.672
	5	0.534	0.489	0.288	0.256
	6	0.01	0.008	0.024	0.022
10	2	1	1	1	1
	3	1	1	1	1
	4	1	1	0.993	0.998
	5	1	1	0.969	0.956
	6	1	1	0.95	0.928

of the ego vehicle, and connectivity of leading vehicles can further improve that.

Table 3.4 shows the average lane changing success rate when N and $a_{x,N+1,d}$ change. It consistently presents that the more utilization of connectivity, system performance is better. Moreover, when N gets larger, it results in larger success rate because more connected leading vehicles can cooperate to leave larger space for the ego vehicle.

Figure 3.11 demonstrates the performance of our planner design with a specific example, in which $N = 5$ and $a_{x,N+1,d} = 5$ meters per second squared. It shows the lateral position and longitudinal velocity of the ego vehicle under different planners. Under 'CV_all' planner, the ego vehicle completes lane changing in about three seconds. Under other three planners, the ego vehicle first move laterally to the target lane, and then turn back to the original lane when $t = 3$ seconds. 'CV_follow' results in longer time of staying in the target lane, compared with 'CV_none' and 'No_agg_assess'. In this example, 'CV_none' and 'No_agg_assess' lead to the same trajectory of the ego vehicle.

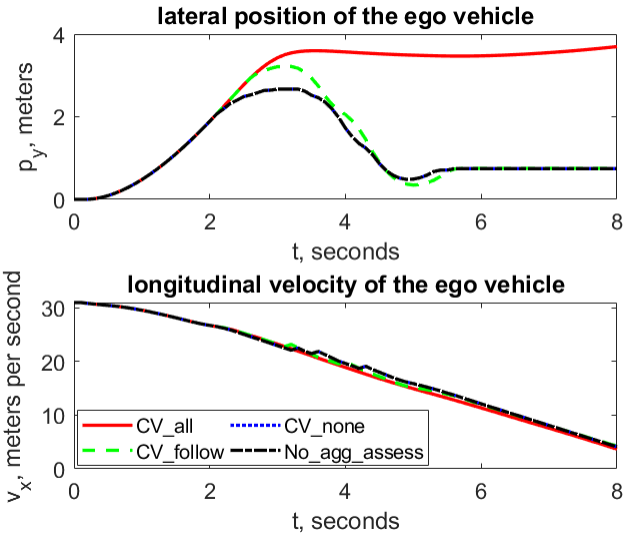

Figure 3.11: Lateral position and longitudinal velocity of the ego vehicle in an example scenario are plotted under different planners, when the number of connected leading vehicles is $N = 5$ and the deceleration of non-connected leading vehicle L_{N+1} is $a_{x,N+1,d} = 5$ meters per second squared.

3.4.2 Impact of Unexpected Promise Violation

According to the *promise* assumption introduced in Section 3.3, connected leading vehicle L_i will keep its acceleration in the range $[-a_i^{m,d}, a_i^{m,a}]$ as long as that does not hurt its own safety. Let us define the promise violation rate p_v, which denotes the probability that the promise is violated unexpectedly in every control period. We assume that promise violation is independent among all connected vehicles, and the vehicle can take any deceleration following the uniform distribution $[z_{x,i,d}, a_{x,N+1,d}]$ when violating the promise.

Figure 3.12 presents the collision rate and lane changing success rate under varied promise violation rate p_v when the first non-connected leading vehicle L_{N+1} takes different sudden deceleration $a_{x,N+1,d}$. Generally speaking, a larger promise violation rate and deceleration result in smaller lane changing success rate and higher collision risk. However, when promise violation rate

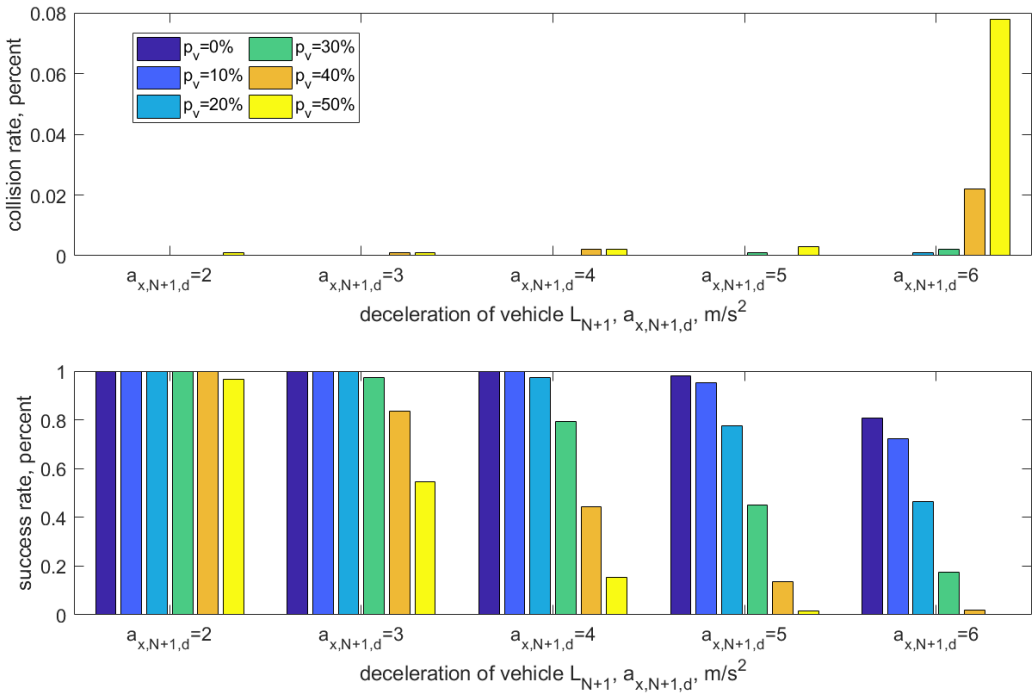

Figure 3.12: Collision rate and lane changing success rate under different promise violation rates are presented when the number of connected leading vehicles is $N = 10$. Promise violation rate p_v is the probability that the promise is violated unexpectedly every control period. We assume that promise violation is independent among all connected vehicles, and the vehicle can take any deceleration following the uniform distribution $[z_{x,i,d}, a_{x,N+1,d}]$ when violating the promise. The horizontal axes show the sudden deceleration of non-connected leading vehicle L_{N+1}.

is within 20% and the sudden deceleration is less than 5 meters per second squared, collision rate is 0% and success rate remains to be relatively high. **This shows the robustness of our approach.**

CHAPTER 4
SAFETY-ASSURED SPECULATIVE PLANNING WITH ADAPTIVE PREDICTION

This chapter focuses on addressing complex driving scenarios where the surrounding vehicles' intentions and planned trajectories have multiple possibilities in prediction, which is based on the work published at [188]. In Section 4.1, we present our method of speculative planning with adaptive prediction. Section 4.2 shows the experimental results.

4.1 Speculative Planning Framework

4.1.1 An Illustrating Example

We consider lane changing in multi-lane highway as a representative application. Fig. 4.1 shows an example scenario where the prediction for the surrounding vehicle gets adapted to the system states and the ego vehicle adjusts its action accordingly. Initially, as shown in the subplot (a), when the two vehicles are still far away from the highway off-ramp, the ego vehicle predicts that the probabilities of different routes for the surrounding vehicle are $p_1 = 0.8$, $p_2 = 0.02$ and $p_3 = 0.18$. A few seconds later, as shown in subplot (b), the surrounding vehicle is changing to the right-most lane before the off-ramp. The prediction gets adapted as the route 1 is no longer possible. Finally, as shown in subplot (c), the surrounding vehicle exits from the off-ramp, and route 3 becomes the only possible one.

We assume that the reward function measuring system performance is the average speed of the ego vehicle. Our planner behaves as follows. During (a), the ego vehicle may keep its speed or accelerate since route 1 has the highest probability for the surrounding vehicle, as long as it is

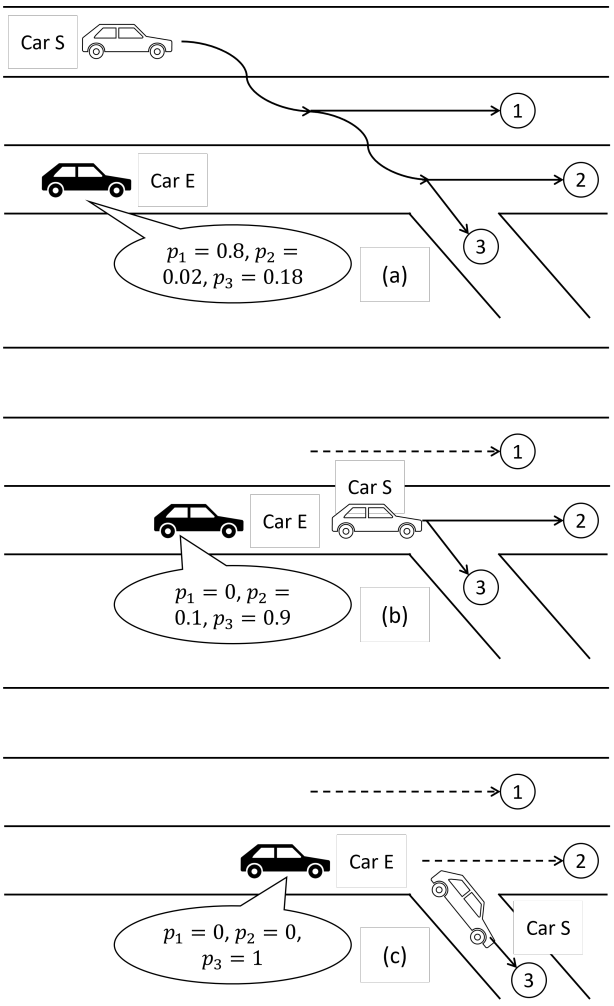

Figure 4.1: The subplots (a), (b) and (c) show that the system states and the prediction for the surrounding vehicle change as time goes on. Solid lines and dash lines represent possible and impossible routes at that time. Note that the surrounding vehicle's states in (b) do not match the route prediction with the highest probability in (a), but our planner can still ensure system safety as it considers all possible predicted behaviors and trajectories.

impossible to collide even in the worst case under all three possible routes. During (b), the ego vehicle remains safe because the less possible scenario (b) was considered in the planning process during (a). The ego vehicle will not necessarily decelerate harshly because route 3 is the most possible one. During (c), the ego vehicle finds that the surrounding vehicle indeed took the most possible route 3 in the prediction during (b), and can now accelerate without hesitance.

Our speculative planning method maximizes the expected reward, while leaving enough buffer space to ensure safety for those less possible predicted scenarios, as shown below.

4.1.2 Problem Formulation

We denote the system state with $\mathbb{S} = \{d_E, d_S, v_E, v_S, l_E, l_S\}$, where d, v and l denote the traveled distance, the velocity of the vehicle, and the lane that the vehicle is in, respectively. The subscript E and S of these variables correspond to ego vehicle and surrounding vehicle, respectively.

The probabilistic prediction of the surrounding vehicle's future trajectory can be represented as

$$\mathbb{P} = \{\{r_i, p_i, f_i(w_i)\}, i = 1, 2, \cdots, N\}, \tag{4.1}$$

where r_i denotes the discrete route choice, p_i denotes the probability of the route choice, w_i denotes the vector of related parameters to define a trajectory corresponding to the route r_i, and $f_i(w_i)$ defines the probability distribution of w_i under the route r_i. Assume that there are N different possible route choices, we have

$$\sum_{i=1}^{N} p_i = 1. \tag{4.2}$$

We assume that the prediction is conservative such that the real trajectory of the surrounding vehicle is always included and bounded by the prediction. The real trajectory is represented with

$\{\hat{r}, \hat{w}\}$, which are sampled from random variables r and w given $\mathbb{P}$. We have

$$\hat{r} = r_k,\, f_k(\hat{w}) > 0,\, \exists k \in \{1, 2, \cdots, N\}. \tag{4.3}$$

The system dynamics can be formulated as:

$$\begin{cases} \dot{d}_E(t) = v_E(t), \\ \dot{v}_E(t) = u(t), \\ l_E(t) \equiv l_E(t_0), \\ \dot{d}_S(t) = v_S(t), \\ \dot{v}_S(t) = \phi(\hat{w}), \\ l_S(t) = \psi(\hat{r}, \hat{w}, d_S(t)), \end{cases} \tag{4.4}$$

where $u(t)$ is the control input, representing the acceleration for the ego vehicle E. t_0 is the initial time, and $l_E(t) \equiv l_E(t_0)$ means that the ego vehicle will go straight and stay in the lane. $\phi(\hat{w})$ is a function to derive acceleration of surrounding vehicle S from the parameter vector $\hat{w}$. The function $\psi(\hat{r}, \hat{w}, d_S(t))$ can determine the lane that the surrounding vehicle is in. As shown in Fig. 1.2, we assign ids $\{0, 1, 2, 3\}$ to lanes from the leftmost to the rightmost, and lane 3 corresponds to the off-ramp.

To ensure safety, the system need to satisfy

$$l_E(t) \neq l_S(t) \;\vee\; |d_E(t) - d_S(t)| \geq d_m, \forall t \geq t_0, \forall \hat{r}, \hat{w} \text{ s.t. Eq. (4.3)}, \tag{4.5}$$

where d_m is the minimum distance gap between vehicles to prevent collisions. Let $J(t)$ denote

the reward function[1]. The sum of reward function over horizon $t \in [t_0, t_h]$, $\sum_{t=t_0}^{t_h} J(t)$, can be transferred to another function $Q(u(t_0), \bar{u}, \hat{r}, \hat{w})$[2], based on Eq. (4.4). Since $\hat{r}$ and $\hat{w}$ are unknown, our goal is to maximize the expectation of $Q(u(t_0), \bar{u}, r, w)$ given the probabilistic prediction $\mathbb{P}$. Then we have

$$u(t_0) = \arg\max_{u(t_0)} \mathbb{E}_{(r,w)\in\mathbb{P}} \left[\max_{\bar{u}} (Q(u(t_0), \bar{u}, r, w)) \right], \text{ s.t. Eq. (4.5).} \tag{4.6}$$

Similar to the receding horizon method in the MPC, we will use only $u(t_0)$ for the current time step, and run the optimization in Eq. (4.6) periodically for the following control inputs.

4.1.3 Speculative Planning with Adaptive Prediction

Next we present our speculative planning algorithm, as shown in Algorithm 2, which generates control input for maximizing the expected reward. First, we set the initial value of $u(t)$ to be 0, representing no acceleration (line 1). Similarly, we set the initial values for reward Ω, safety indicator Θ and the minimum distance gap between vehicles δd (lines 2-4). Both Θ and δd are acquired by the safety evaluation algorithm, as shown in Algorithm 3. Θ is a binary variable, $\Theta = 0$ denotes that the system is safe now and there exists a series of control inputs $\bar{u}$ to keep the system safe, $\Theta = 1$ denotes that the system is possible to be unsafe given $\mathbb{P}$. Then let the temporary variable a_t loop through the acceleration range $[a_{min}, a_{max}]$ (lines 5-6). We evaluate the system safety (line 7) and compute the expected reward given the probabilistic prediction $\mathbb{P}$ (line 8) for each a_t. If we get safety assurance with current action a_t (line 9), or we get a higher reward and a larger distance gap (line 14), we update the control input $u(t)$ and its corresponding reward, safety indicator and minimum distance gap. $\varphi(\Omega_t, \delta d_t)$ is a function to balance reward and

[1] $J(t)$ is the short form of $J(d_E(t), d_S(t), v_E(t), v_S(t), l_E(t), l_S(t), u(t))$.

[2] $\bar{u}$ represents a series of control inputs excluding $u(t_0)$, i.e., $u(t_0 + \delta t : \delta t : t_h)$.

Algorithm 2: Speculative planning

Result: Control input $u(t)$
Input: $\mathbb{S}(t) = \{d_E(t), d_S(t), v_E(t), v_S(t), l_E(t), l_S(t)\}$,
$\mathbb{P} = \{\{r_i, p_i, f_i(w_i)\}, i = 1, 2, \cdots, N\}$

1 $u(t) \leftarrow 0.0$;
2 Reward $\Omega \leftarrow 0$;
3 Safety indicator $\Theta \leftarrow 1$;
4 Minimum gap $\delta d \leftarrow 100$;
5 $a_t \leftarrow a_{min}$;
6 **while** $a_t \leq a_{max}$ **do**
7 $\quad \Theta_t, \delta d_t \leftarrow SafetyEval(\mathbb{S}(t), \mathbb{P}, a_t)$;
8 $\quad \Omega_t \leftarrow ExpectedReward(\mathbb{S}(t), \mathbb{P}, a_t)$;
9 $\quad$ **if** $\Theta == 1$ && $\Theta_t == 0$ **then**
10 $\quad\quad u(t) \leftarrow a_t$;
11 $\quad\quad \Omega \leftarrow \Omega_t$;
12 $\quad\quad \Theta \leftarrow \Theta_t$;
13 $\quad\quad \delta d \leftarrow \delta d_t$;
14 $\quad$ **else if** $\Theta == 0$ && $\Theta_t == 0$ && $\varphi(\Omega_t, \delta d_t) \geq \varphi(\Omega, \delta d)$ **then**
15 $\quad\quad u(t) \leftarrow a_t$;
16 $\quad\quad \Omega \leftarrow \Omega_t$;
17 $\quad\quad \delta d \leftarrow \delta d_t$;
18 $\quad$ **end**
19 $\quad a_t \leftarrow a_t + \delta a$;
20 **end**

minimum gap, and a larger value is preferred. We will prove the safety guarantee of our algorithm (under certain conditions) later in Section 4.1.4.

Safety evaluation is conducted as shown in Algorithm 3. We initially assume that it is safe (line 1) and set the minimum distance gap to be 100 (line 2). Then for every possible route (line 3), we assess whether it is still feasible according to the latest status of the surrounding vehicle S (line 4). For example, if the surrounding vehicle is already on the off-ramp to exit the highway, routes 1 and 2 become impossible. We compute the union set of all possible future spatial-temporal trajectories [189] of the surrounding vehicle, $\mathbb{T}_i$, even if the probability is small according to $f_i(w_i)$

Algorithm 3: *SafetyEval(): Safety evaluation*

Result: Safety indicator Θ_t, minimum gap δd_t
Input: $\mathbb{S}(t) = \{d_E(t), d_S(t), v_E(t), v_S(t), l_E(t), l_S(t)\}$,
$\mathbb{P} = \{\{r_i, p_i, f_i(w_i)\}, i = 1, 2, \cdots, N\}$, a_t

1 $\Theta_t \leftarrow 0$;
2 $\delta d_t \leftarrow 100$;
3 **for** $i \in \{1, 2, \cdots, N\}$ **do**
4 **if** *IsFeasible*$(r_i, d_S(t), l_S(t))$ **then**
5 $\mathbb{T}_i \leftarrow Traj(d_S(t), v_S(t), l_E(t), l_S(t), r_i, f_i(w_i))$;
6 $\delta d_{min} \leftarrow MinGap(\mathbb{T}_i, d_E(t), v_E(t), a_t)$;
7 $\delta d_t \leftarrow min(\delta d_t, \delta d_{min})$;
8 **end**
9 **end**
10 **if** $\delta d_t \leq \delta d_s$ **then**
11 $\Theta_t \leftarrow 1$;
12 **end**

(line 5). Assume that the ego vehicle adopts the control input a_t at the current time step, and it is allowed to take any acceleration in the range $[a_{min}, a_{max}]$ for the following steps to prevent overlap with $\mathbb{T}_i$. We compute the minimum distance gap between $\mathbb{T}_i$ and the future trajectory of ego vehicle, δd_{min} (line 6). We will update δd_t with δd_{min} acquired under route r_i (line 7). If the gap is even less than the threshold δd_s, it is unsafe (lines 10-12). As stated before, this result is used in Algorithm 3 (line 7).

Algorithm 4 presents the computation of the expected reward given $\mathbb{P}$. Let Ω_t and p_t denote the sum of weighted reward and the sum of weights (i.e., probabilities), respectively. We assign initial values for them (lines 1-2). Similarly to Algorithm 3, we loop through each feasible route r_i (lines 3-4) and adapt our prediction and probability distribution according to the latest status of the surrounding vehicle (line 5). For example, assume that there is an appropriate road segment for the lane changing, as the surrounding vehicle moves forward towards the end of the road segment, the range of possible lane-changing positions can be smaller, thus the weight (i.e., probability)

Algorithm 4: *ExpectedReward(): Computation of the expected reward*

Result: Reward Ω_t
Input: $\mathbb{S}(t) = \{d_E(t), d_S(t), v_E(t), v_S(t), l_E(t), l_S(t)\}$,
$\mathbb{P} = \{\{r_i, p_i, f_i(w_i)\}, i = 1, 2, \cdots, N\}$, a_t

1 $\Omega_t \leftarrow 0$;
2 $p_t \leftarrow 0$;
3 **for** $i \in \{1, 2, \cdots, N\}$ **do**
4 **if** *IsFeasible*$(r_i, d_S(t), l_S(t))$ **then**
5 $p_i^{'}, f_i^{'}(w_i) \leftarrow Adapt(d_S(t), l_S(t), r_i, p_i, f_i(w_i))$;
6 $\Omega_{t,i} \leftarrow 0$;
7 **for** $k \in \{1, 2, \cdots, N_s\}$ **do**
8 $w_{i,k} \leftarrow Sample(f_i^{'}(w_i))$;
9 $\Omega_{t,i} \leftarrow \Omega_{t,i} + \max_{\bar{u}}(Q(a_t, \bar{u}, r_i, w_{i,k}))$;
10 **end**
11 $\Omega_t \leftarrow \Omega_t + p_i^{'}\Omega_{t,i}/N_s$;
12 $p_t \leftarrow p_t + p_i^{'}$;
13 **end**
14 **end**
15 $\Omega_t \leftarrow \Omega_t/p_t$;

for the certain route needs to be adjusted. Under the specific route r_i, we sample the vector of parameters $w_{i,k}$ to generate the trajectory of the surrounding vehicle for N_s times (lines 7-8). We compute the reward of the ego vehicle for each sample $\{r_i, w_{i,k}\}$, and $\Omega_{t,i}$ is the sum of all rewards (line 9). As $\Omega_{t,i}/N_s$ is the averaged reward under route r_i, we update Ω_t and p_t accordingly (lines 11-12). Finally, Ω_t is scaled to be the expected reward (line 15), which as stated before, is used in Algorithm 2 (line 8).

The prediction adaptation is based on two assumptions: (1) the surrounding vehicle does not move backward; (2) after a complete and safe lane changing, the surrounding vehicle does not return back to the original lane. When the prediction is outdated, we filter out those impossible behaviors and trajectories of the surrounding vehicle based on its latest status, and scale the probabilities of the remaining such that the sum is still 1.

4.1.4 Safety Guarantee

Theorem 4.1.1. *For a dynamical system defined by Eq.* (4.4)*, our proposed planner (Algorithms 2, 3 and 4) will ensure system safety, if Eq.* (4.3) *holds and there exists a safe planning decision at the initial state of the system.*

Proof. Let us prove this by contradiction. We first assume that there exists a time t_s such that $\Theta(t_s) = 0$ and $\Theta(t_s + \delta t) = 1$. According to Algorithm 2, if there exists $u(t_s + \delta t)$ such that $\Theta_t(t_s + \delta t) = 0$, then $\Theta(t_s + \delta t) = 0$. So we have $\Theta_t(t_s + \delta t) = 1$, $\forall u(t_s + \delta t)$. According to Algorithm 3, there exists at least one feasible route r_i such that $\delta d_{min}(t_s + \delta t) \leq \delta d_s < \delta d_{min}(t_s)$, $\forall u(t_s+\delta t)$. Since $d_E(t_s+\delta t)$ and $v_E(t_s+\delta t)$ are acquired by substituting $u(t_s)$ into Eq. (4.4), there exists an action $a_t(t_s + \delta t)$ such that $\delta d_{min}(t_s + \delta t) = MinGap(\mathbb{T}_i(t_s + \delta t), d_E(t_s + \delta t), v_E(t_s + \delta t), a_t(t_s + \delta t)) = MinGap(\mathbb{T}_i(t_s + \delta t), d_E(t_s), v_E(t_s), a_t = u(t_s))$. Thus, $MinGap(\mathbb{T}_i(t_s + \delta t), d_E(t_s), v_E(t_s), a_t = u(t_s)) < \delta d_{min}(t_s) = MinGap(\mathbb{T}_i(t_s), d_E(t_s), v_E(t_s), a_t = u(t_s))$. It requires $\mathbb{T}_i(t_s + \delta t) \setminus \mathbb{T}_i(t_s) \neq \emptyset$. However, $\mathbb{T}_i$ is the union set of all possible future spatial-temporal trajectories of the surrounding vehicle, $\mathbb{T}_i(t_s + \delta t) \subseteq \mathbb{T}_i(t_s)$. From this contradiction, we know that $\Theta \equiv 0$ and the system is safe under our proposed planner. □

4.1.5 Surrounding Vehicle's Model

It is worth noting that the generality of our planning method will not be affected by the model of the surrounding vehicle.

For longitudinal motion, we assume that the surrounding vehicle is controlled to maintain the desired speed v_d under the acceleration function $\phi(\hat{w})$. For lateral motion, we assume that the surrounding vehicle indicates a right turn and intends to change lanes when $d_S = d_{lc}^0$. The route is randomly determined according to p_1, p_2 and p_3. Route 1 corresponds to change lane once, while

route 2 and 3 need to change lanes twice. Based on the selected route, the execution of first and second lane changing happen at d_{lc}^1 and d_{lc}^2, respectively. We assume that $d_{lc}^1 - d_{lc}^0$ and $d_{lc}^2 - d_{lc}^1$ are closely related to the personality of the driver, which is represented by aggressiveness [16], [48]. We then have

$$\delta d_{lc} = d_a q_a + d_c + d_n, \tag{4.7}$$

where q_a, the aggressiveness level, is a real variable satisfying $-1 \leq q_a \leq 1$. A larger q_a represents a more aggressive driver. $d_a < 0$ is the coefficient for the aggressiveness term, and d_c is a constant. With the noise term d_n, $d_{lc}^1 - d_{lc}^0$ and $d_{lc}^2 - d_{lc}^1$ are not necessarily the same. For a more aggressive surrounding vehicle, $d_{lc}^1 - d_{lc}^0$ and $d_{lc}^2 - d_{lc}^1$ are smaller, which leaves less time for the ego vehicle to react.

4.2 Experimental Results

4.2.1 Effectiveness of Our Approach

We compare system safety and performance under different planners to demonstrate the strength of our approach: baseline methods 'IDM_1', 'IDM_2' and 'IDM_3' are all based on Intelligent Driver Model [187], [190], which is a common car following model in the transportation domain. 'IDM_1' is the original version that the ego vehicle only follows the surrounding vehicle in the same lane, 'IDM_2' means that the ego vehicle will follow the surrounding vehicle that intends to change lanes from the adjacent lane as well, 'IDM_3' means that the ego vehicle follows the surrounding vehicle in any of the three lanes in the highway. A larger subscript corresponds to earlier reactions to the surrounding vehicle, which is expected to be safer and less efficient. Baseline method 'MPC' represents the Model Predictive Control approach [47], [191], [192], which can address the uncertainty in the behavior and trajectory of the surrounding vehicle. However, it does not consider

the underlying probability distribution. 'SPAP' is our proposed method, Speculative Planning with Adaptive Prediction. For the reward function, without losing generality, we assume that $J(t) = v_E(t)$. 'MPC+agg' and 'SPAP+agg' are extensions of MPC and our method with consideration of vehicle aggressiveness.

We set the horizon of each simulation to be 12 seconds, and the simulation step size and control step size δt are both at 0.1 second. The desired speed is $v_d = 25$ m/s, and the speed limit is 30 m/s in the highway. We randomly sample the values for p_1, p_2 and p_3 and evaluate the performance of different planners with or without the predicted aggressiveness of the surrounding vehicle, as shown in Table 4.1. Each row corresponds to the averaged results of $10,000$ simulations. We can see that **our SPAP and SPAP+agg planners provide significant improvement on system performance (larger average speed) over the baseline planners, while ensuring system safety**. SPAP+agg and MPC+agg provide improvement over SPAP and MPC, as we assume that the aggressive prediction is accurate and can help reduce the uncertainty in predicting surrounding vehicle behavior (if not, they may not provide such improvement).

Table 4.1: Safety and performance evaluation for different planners with or without predicted aggressiveness of the surrounding vehicle.

Planners	safety rate	average speed	final speed
IDM_1	94.73%	24.73 m/s	24.68 m/s
IDM_2	96.57%	23.76 m/s	23.38 m/s
IDM_3	100%	22.96 m/s	23.82 m/s
MPC	100%	25.85 m/s	29.21 m/s
MPC+agg	100%	26.45 m/s	29.26 m/s
SPAP	**100%**	**27.56 m/s**	**29.45 m/s**
SPAP+agg	**100%**	**27.90 m/s**	**29.53 m/s**

We conduct additional experiments to further study the impact of different route probabilities. Fig. 4.2 presents the safety rate and average speed of the ego vehicle when the probability of route 1, p_1, changes. We set $p_3 = 0.2$, and p_2 is determined such that $p_1 + p_2 + p_3 = 1$. As we expected,

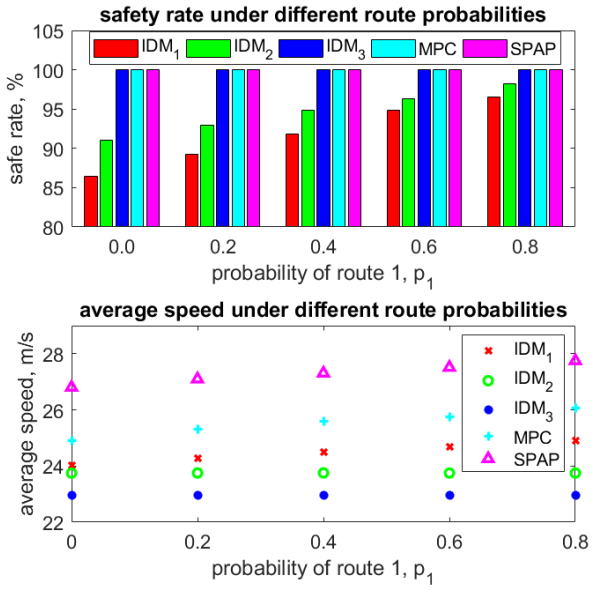

Figure 4.2: Safety rate and average speed of the ego vehicle under different planners are compared when the probability of route 1, p_1, changes. The probabilities of the other two routes are set as $p_2 = 0.8 - p_1$ and $p_3 = 0.2$.

since it is route 2 that has the most interference with the ego vehicle, the safety rate is the lowest when $p_1 = 0$ and $p_2 = 0.8$ for planners IDM_1 and IDM_2. The safety rate increases when p_1 increases gradually. For planners IDM_3, MPC and SPAP, there is no any collision. IDM_3 is over-conservative by its nature, MPC is formulated with safety constraints, and our proposed SPAP has safety guarantee, as presented in Section 4.1.4.

Fig. 4.2 also shows that the average speed increases slightly as p_1 increases. For the three IDM-based planners, it meets our expectation that a larger subscript corresponds to a less efficient planner, thus resulting in a lower average speed. MPC performs better than these IDM-based planners.

Table 4.2: Computation time, safety and performance evaluation under different sampling times N_s.

N_s	time cost	safety rate	average speed	final speed
10	0.03 s	100%	26.21 m/s	28.39 m/s
25	0.04 s	100%	26.82 m/s	28.47 m/s
50	0.07 s	100%	27.30 m/s	28.64 m/s
100	0.12 s	100%	27.31 m/s	28.63 m/s
200	0.22 s	100%	27.27 m/s	28.64 m/s

4.2.2 Real-time Computation Complexity

Since the control step size is set to 0.1 second, the planner could only be employed in real-time if the computation time is within 0.1 second. To evaluate the real-time computation complexity of our approach, we set $p_1 = 0.4$, $p_2 = 0.4$, $p_3 = 0.2$ and conduct experiments in a server with Intel(R) Xeon(R) Gold 6130 CPU @ 2.10GHz. We choose the number of sampling times N_s to be 10, 25, 50, 100 and 200. Intuitively, the larger the N_s, the more accurately the planner samples the prediction distribution and the better the performance, however at a higher computational cost.

From the results in Table 4.2, we can see that when N_s increases from 10, 25 to 50, there is a substantial increase of system performance (i.e., average speed the vehicle can safely achieve). Further increasing N_s to 100 and 200 will not lead to significant change in performance. Thus, $N_s = 50$ seems to be the sweet spot that balances performance and computational load for this application. Note that the real-time computation demand can be satisfied when $N_s = 50$, as the time cost of 0.07 second is smaller than 0.1 (moreover, the reported average computation time includes not only the time spent on planning, but also the time on the simulator; the real computation time is lower). We plan to conduct more evaluations on other driving tasks and computing platforms in the future work.

CHAPTER 5
SECURING CONNECTED VEHICLE APPLICATIONS WITH BLOCKCHAIN

In this chapter, we develop an efficient dual cyber-physical blockchains framework to build trust and secure communication for CV applications, which is based on the work published at [193]. The framework enables us to efficiently track and update a trust estimate for each vehicle in a large-scale traffic network, with low resource overhead. Section 5.1 introduces the security threat models we address in this work. Section 5.2 presents the design of our dual blockchain framework. Section 5.3 analyzes our framework's defense performance against message spoofing attack, bad mouthing attack, and Sybil and voting attack, and presents the simulation results in SUMO. Section 5.4 analyzes the computation, communication and storage overhead for each vehicle, and the trade-offs in adjusting the region size that one blockchain covers and the block generation period.

5.1 Threat Models to Connected Vehicles

In this work, we focus on the attacks from vehicle-side devices, e.g., those from the On-Board Units (OBUs) for CV applications [35]. While the techniques can be extended to protect against attacks on infrastructure units, it is reasonable to expect that it is typically harder to attack the infrastructure. In particular, we consider malicious vehicles that can generate falsified messages and broadcast them to other vehicles. It is important to note that we do not assume that the attacker can spoof the sender identities in the messages. We assume that such an identity is verifiable and non-forgeable with digital signatures technique (e.g., SCMS). Moreover, we assume that the communication infrastructure is mostly resilient and most surrounding vehicles within a known communication range should receive the same message within a known time bound. We also

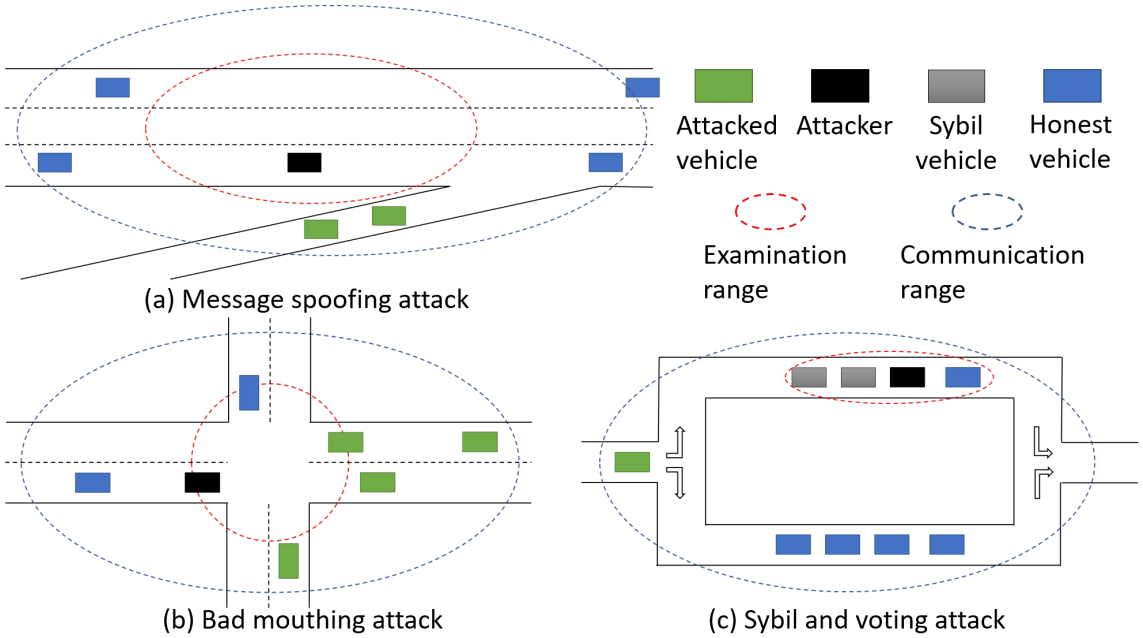

Figure 5.1: Threat models considered in our work. Subplot (a) displays that an attacker sends falsified messages with wrong traffic status on the highway to mislead the green merging vehicles. Subplot (b) displays that an attacker forges misbehavior of surrounding honest vehicles to hurt their reputation and traffic efficiency in the intelligent intersection. Subplot (c) displays that an attacker forges two pseudonymous identities by launching a Sybil attack. Together they report falsified traffic accidents in one of the routes, and may dominate the voice in voting among the surrounding vehicles so that the falsified messages would not be detected. This attack may lead to larger traffic density and travel time because more honest vehicles will choose the other route that is without the fake accident.

assume that most honest vehicles have loosely synchronized clocks (e.g., using Network Time Protocol (NTP)) for the liveness of blockchain. Finally, we assume that most vehicles in the traffic network are honest, such that most of the stake (more than 2/3) can be held by honest vehicles to ensure the safety of blockchain.

Under these assumptions, malicious attackers can compromise a small fraction of vehicles to broadcast falsified position and velocity data, forge traffic accidents and destroy others' reputation. Specifically, we consider three threat models in this paper with popular and representative CV

applications, as shown in Fig. 5.1. Among them, the message spoofing attack could target many other CV applications and be defended by trust frameworks such as ours. The bad mouthing attack and Sybil and voting attack in fact target the trust framework itself, and as shown later, our blockchain framework can effectively mitigate such attacks as well.

5.1.1 Message Spoofing Attack

In Fig. 5.1(a), two green vehicles communicate with the vehicles on the highway to adjust their speed for merging onto the highway. The attacker (shown in black) can send out falsified position and velocity data of itself, which may induce the green victim vehicles to accelerate/decelerate. This may damage traffic efficiency and even put vehicles near the on-ramp at the risk of collisions. A single malicious vehicle may start this type of attack when no honest vehicle can judge the truth of the message or the stake of attackers is overwhelming in the local area.

5.1.2 Bad Mouthing Attack

In Fig. 5.1(b), all vehicles send their current and destination lanes as well as estimated arriving time to a centralized intersection manager, which then decides the passing order and timing of the vehicles, similarly as in [5]. When a trust framework is deployed to secure the communication, an attacker may maliciously report misbehavior to degrade the trust for honest vehicles [194]. Without an accurate assessment of the trust, the intersection manager will not be able to decide the passing order and timing for vehicles, and they will have to adapt the traditional traffic rules as if there were stop signs in every direction.

5.1.3 Sybil and Voting Attack

In Fig. 5.1(c), there are two route choices in the region. An attacker can generate fake and pseudonymous vehicles by Sybil attack. Together they can broadcast a falsified traffic accident in one of the routes. In a trust framework based on consensus, while honest vehicles may disagree with the falsified accident, their voices could be dominated by the attacker and Sybil vehicles in the voting process for assessing the falsified accident, in which case, other vehicles coming to this region will believe the falsified accident and go through the other route, resulting in congestion and low traffic efficiency.

5.2 Efficient Dual Cyber-physical Blockchains Framework

To avoid confusion, in our work, *messages* carry traffic information that vehicles share with others, e.g., Basic Safety Message (BSM), Cooperative Awareness Message (CAM), Cooperative Perception Message (CPM), etc. To build trust among vehicles via blockchain, vehicles may generate *transactions* and broadcast them to other vehicles in the same region. The transactions generally include the report of malicious misbehavior, application of transferring records, etc. By aggregating all transactions within a period, one vehicle can then update all vehicles' records and build a block accordingly.

5.2.1 Framework Overview

Our blockchain design leverages the block generation and consensus mechanism from Algorand. Technically speaking, it is based on proof of stake. In our design, the stake is computed from trust points and proof-of-travel credits. The higher stake a vehicle holds, the higher chance it can be selected as a leader or verifier to maintain the blockchains. When driving, only traffic information

sent from vehicles with high trust points and proof-of-travel credits can be viewed as trustworthy. Such design sets up a relatively high threshold for malicious vehicles to manipulate the blockchains and start attacks.

To update blockchain on time, and alleviate resource demand for vehicles, we leverage the sharding method to partition vehicles into subsets according to their traveling region. Each vehicle has a permanent address and a current active address. For trust points blockchain, attackers' misbehavior should be exposed as soon as possible, and surrounding vehicles in the current active region can respond within a short time. Thus, trust points blockchain in one region is maintained by those vehicles that claim to be active in the current region and will only record those vehicles' trust points. A vehicle that moves across different regions can claim its new active region and have its trust points record transferred. The detailed mechanism is introduced in Section 5.2.2. For proof-of-travel blockchain, since it records a vehicle's historical information over a long period (e.g., 100 days), it is maintained by those vehicles that have their permanent address in this region. It will only record those vehicles' proof-of-travel credits. In this way, different regions will have their blockchains maintained and updated independently.

Other details, such as the transaction generation and physical verification processes for the two blockchains, are introduced in the following Sections 5.2.2 and 5.2.3, respectively.

5.2.2 Trust Points Blockchain

Trust points blockchain is mainly for identifying and exposing malicious attackers. When one vehicle sends out a message, surrounding vehicles within the communication range can receive and verify it based on information from their own on-board sensors or other sources (e.g., a message with falsified position data may be deemed as suspicious by surrounding vehicles using their own

sensors[1]). Then depending on the situation, various types of transactions can be generated and sent. Each transaction includes transaction ID (TID), smart contract type ID (SCID), senderID, debateID, regionID, location, time and payload. Transaction senders will encrypt the transactions with their private keys and broadcast them. Receivers can verify the identity of senders with the public keys. SCID identifies the transaction type and the corresponding smart contract for updating vehicles' points/credits. Location and time are the transaction generation location and time. DebateID is the ID of the vehicle that sends out the suspicious message. RegionID is the ID of the region that a vehicle is moving into. TID is the hash value of senderID, debateID, regionID, location and time, and is considered unique. Various transactions are sent in the following situations:

- If a vehicle disagrees with a message sent by another vehicle, it can report this disagreement in a transaction with SCID=0000.
- Upon receiving a disagreement transaction (SCID=0000), other vehicles can take a stand on agreeing or disagreeing in a new transaction with SCID=0000 if they have not done so.
- If a vehicle disagrees with the judgement made in the previous voting process (details of the process are introduced later in Algorithm 5), it can report the disagreement after enough evidence is gathered (i.e., when stake of honest vehicles gets larger) in a new transaction with SCID=0001.
- A vehicle moving to another region may apply to transfer its records in a transactions with SCID=0002.

The transactions received within a period (i.e, the round latency of trust points blockchain) are processed with our designed contracts in Algorithms 5, 6 and 7.

Transactions with SCID=0000 are to report attackers and falsified messages. They are handled by an instant voting contract, as shown in Algorithm 5.

[1]How vehicles may verify messages based on physical information from their own sensors or other sources is beyond the scope of this paper, which focuses on the design of the blockchain framework.

Let us use the highway merging application in Fig. 5.1(a) to help explain the algorithm. First, the attacker near the ramp sends out a falsified message with wrong traffic status to mislead vehicles on the ramp. If there are honest vehicles in both the examination and communication ranges, they can also receive the message and may deem it suspicious based on the information from their own on-board sensors. They can then generate transactions with SCID=0000 to report such findings and broadcast the transactions to other vehicles in the region. Within a period, surrounding vehicles within both the communication and examination ranges are all supposed to take a stand. Then other vehicles in the region collect all transactions and start the smart contract-based voting process as shown in Algorithm 5. The voting process for this contract instance will classify all transaction senders into two groups. One group of vehicles agree with the message's content sent by the vehicle with debateID, and the other group of vehicles disagrees with that. The contract will make a final judgment and update vehicles' trust points by comparing the two groups' accumulated stake. The group with the higher stake will be called majority and have their trust points increased by 1, while the other group will be called minority and have trust points of -1.

A group of attackers with high stake may dominate the voting process if the number of surrounding honest vehicles is initially limited. To avoid repeated attacks from the group of attackers, we design the redressing contract, which allows re-evaluation of vehicles' opinions on the message content from the vehicle with debateID. In particular, if a vehicle disagrees with the previous voting result obtained by Algorithm 5, it can send a transaction with SCID=0001, which triggers the redressing process in Algorithm 6. The redressing algorithm finds all the ended contracts that involve the debateID, and form groups G_s and G_o that represent all the vehicles agree and disagree with debateID, respectively. Let $N(G_s)$ and $N(G_o)$ denote the accumulated state for group G_s and G_o, respectively. If the stake difference between the two groups is larger than a threshold N_{th}, previous judgement can be redressed.

Algorithm 5: Instant Voting for SCID=0000 Transactions

Result: Updated trust points for involved vehicles
Input: senderID, debateID, location, time, opinion
if *time has not elapsed* $2tb_s$ *since receiving the suspicious message (* tb_s *denotes the time bound that most vehicles in the same region can receive the transaction)* **then**
    Find an ongoing contract with the same debateID, different senderID, close location and time;
    if *no such contract* **then**
        Create a new contract instance for debateID;
    end
    if *opinion is agree* **then**
        Add senderID to the agree group;
    else
        Add senderID to the disagree group;
    end
end
if *any contract has existed* $2tb_s$ **then**
    Compare the stakes between the agree group and the disagree group;
    Increase the trust points of the majority by 1;
    Set the trust points of the minority to be -1;
end

When a vehicle moves across different regions, it is required to update its current active region. This way the vehicles within the same active region can communicate efficiently and the real-time performance can be improved. To update the active region, a vehicle needs to transfer trust points and copy its proof-of-travel credits to the new region. This process is shown in Algorithm 7. Note that the trust points of this vehicle in the original region should be set to zero.

5.2.3 Proof-of-Travel Blockchain

Proof-of-travel blockchain is mainly to record vehicles' accumulated contributions to other vehicles. We assume that CVs will broadcast traffic information, e.g., BSM or CAM messages, in an average frequency of f_m Hz. Surrounding vehicles within the communication range will record

Algorithm 6: Redressing for SCID=0001 Transactions

Result: Updated trust points for involved vehicles
Input: senderTID, debateID, time
Find all ended contracts of type SCID=0000, in which the vehicle with debateID finally won;
From all those ended contracts, build a group G_s that includes all vehicles that agree with debatedID and G_o that includes all vehicles disagree with debated ID;
Let $N(G)$ denote the accumulated stake for vehicles in group G and N_{th} denote the threshold number;
if $N(G_o) - N(G_s) > N_{th}$ **then**
    Redress previous contract instances;
    Increase trust points for group G_o by 1;
    Set trust points for group G_s to be -1;
end

Algorithm 7: Transferring Records for SCID=0002 Transactions

Result: Updated trust points for senderID in two regions
Input: senderID, regionID, position
if *position is near the border of current region and the region of regionID* **then**
    Get proof-of-travel credits and trust points of senderID in current active region;
    Transfer these records to another region corresponding to regionID;
    if *another region updated records of senderID* **then**
        Set trust points of the vehicle to 0 in original region
    end
end

the public keys of message senders and the number of valid messages. After each travel period T_{pot} (which is much longer than the period for trust points blockchain), every vehicle v_i will have a list that records the public keys of message senders v_j and the number of messages n_{j2i} from every sender during this period. We propose a metric n_j^{pot}, named "proof-of-travel credits", to summarize all mages commenting on vehicle v_j from all other vehicles v_i, i.e.,

$$n_j^{pot} = \sum_i n_{j2i} \tag{5.1}$$

For a vehicle, its accumulated proof-of-travel credits in the latest N_{sum} periods is computed by:

$$N_j^{pot}[m] = \sum_{k=0}^{N_{sum}-1} \alpha^k \, n_j^{pot}[m-k] \tag{5.2}$$

where $n_j^{pot}[m-k]$ is the proof-of-travel credits in the $(m-k)_{th}$ period, α is a discounting factor, $N_j^{pot}[m]$ is the accumulated N_{sum} periods proof-of-travel credits by the m_{th} period. Vehicles that make a persistent contribution to others will have a high accumulated proof-of-travel credits.

We assume that most vehicles will report the number of messages they received honestly because honest vehicles generally do not know the identity of surrounding vehicles. Honest vehicles can also generate transactions to reveal falsified number report, in a similar way as in the trust points blockchain.

Fig. 5.2 presents the general process of updating the proof-of-travel blockchain. Different vehicles may get registered in different permanent regions, e.g., region A, region B, and so on. We assume that each vehicle is uniquely identified by the regionID and the vehicle index in that region, e.g., A1 is a vehicle with the index of 1 in region A. Since a vehicle may change lane, overtake, choose different routes, or even enter another region, it will encounter different vehicles in the same or different regions. It will record those vehicles and the number of messages they sent in a transaction and broadcast it in the region. As in the figure, each region will have a number of aggregated transactions. Each transaction is denoted by the vehicle senderID with a set including all vehicles it encountered, e.g., E1: $\{$E2, B3, F2, $\cdots\}$ is the transaction sent by vehicle E1. The selected block proposer will then arrange these transactions and build a pre-block by summarizing credits for all vehicles in the sets. After the consensus process, all vehicles in the region will agree with this pre-block. We name it pre-block because it may not include complete information for vehicles in the region, but record vehicles who are not in this region. Then vehicles in neighboring

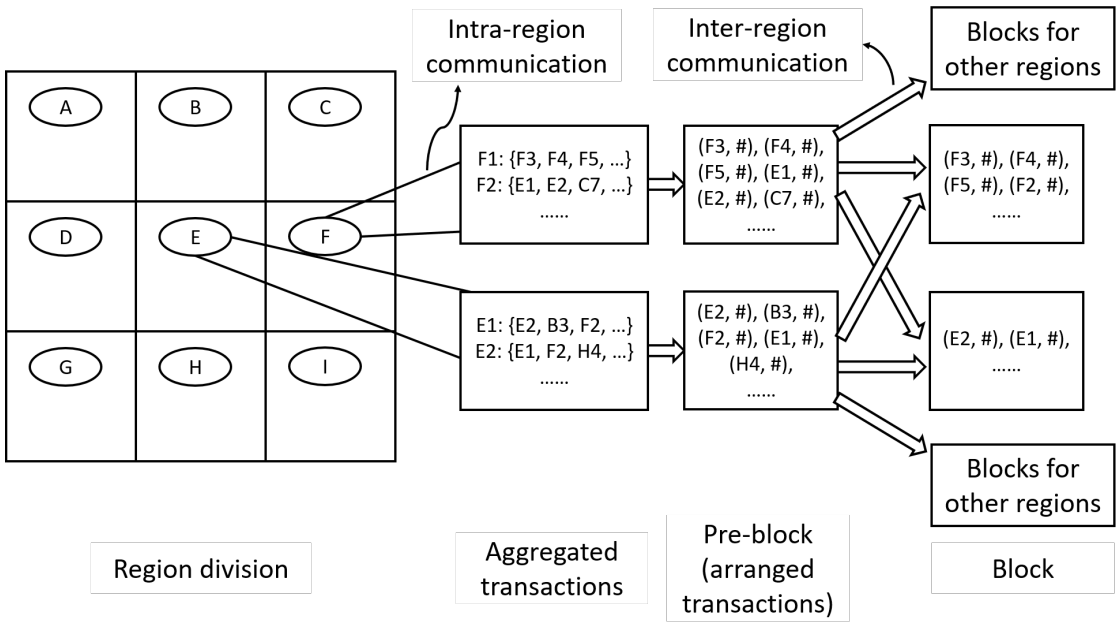

Figure 5.2: Proof-of-travel blockchain updating across multiple regions. With intra-region communication, each vehicle can aggregate transactions received from others in the same region. By computation and arrangement, vehicles can build a pre-block and reach consensus on it, which includes vehicle ID and proof-of-travel credits. Note that records in pre-blocks may not be complete as vehicles may move across multiple regions; but with inter-region communication, blocks for each region will be eventually built.

regions will communicate with each other regarding the pre-blocks. Finally block proposers will build a block containing complete information of vehicles in their region. In this way, to find the proof-of-travel credits of one vehicle, we can easily refer to the blockchain in that vehicle's permanent region.

5.2.4 Stake Computation and Region Size

In framework, the stake N^{stk} is computed from the accumulated proof-of-travel credits N^{pot} in the proof-of-travel blockchain and the trust points N^{tp} in the trust points blockchain:

$$N^{stk} = \frac{N^{pot}}{1 + M^{pot}} + \frac{N^{tp}}{1 + M^{tp}} \tag{5.3}$$

where M^{pot} and M^{tp} denote the mean value of N^{pot} and N^{tp} for all vehicles, respectively. Here we use $1 + M^{pot}$ and $1 + M^{tp}$ as denominator to avoid the near-zero case. This stake is the basis of consensus mechanism for both the trust points blockchain and the proof-of-travel blockchain.

The region size that one blockchain can cover is decided by various factors. First, it is limited by the round latency that we desire. A larger size may result in a larger latency to reach consensus, thus slow down system's response to attacks. Second, for trust points blockchain, the smaller the region is, the lower the resource demand is for all vehicles in the region to maintain and update the blockchain. However, a small region size typically means that vehicles will frequently move across different regions, thus frequently transferring their records. For proof-of-travel blockchain, a smaller region size leads to lower overhead on intra-region communication but higher overhead on inter-region communication. Such trade-offs are addressed in detail in Section 5.3.4 and 5.4.

5.3 Security Analysis

As discussed in the threat models in Section 5.1, a malicious vehicle can broadcast falsified messages, report falsified misbehavior of honest vehicles, or create other pseudonymous identities for manipulating the voting process.

We evaluate the effectiveness of our proposed framework in defending against the three threat models, based on simulations in SUMO. We also analyze the relationship between round latency of the trust points blockchain and the region size in our framework.

5.3.1 Against Message Spoofing Attack

For the threat model described in Fig. 5.1(a), under message spoofing attack and without our framework, the merging vehicle in on-ramp can be fooled to speed up when the highway is congested or to decelerate when there are few vehicles on the highway. When the merging vehicle is able to perceive the traffic environment, it has to adjust its velocity significantly, which leads to low efficiency and possible failure to merge onto the highway. With the protection of our framework, other honest vehicles on the highway may report this spoofing attack and launch the voting process to assess the message. Before the assessment is finished, the merging vehicle can start taking cautious actions in milliseconds, e.g., speeding up only a little bit for reacting to uncertain traffic status on the highway, preventing sudden velocity changes. When all vehicles reach consensus after the voting process, the identity of the attacker is exposed, and its messages are not trustworthy any more.

We compare the average speed, CO emission, and fuel consumption of the merging vehicles under three cases, i.e., without attack, under attack without our framework, and under attack with our framework. We perform 20 simulations for each case, and the average performance is recorded in Table 5.1, where 'ours' is short for our framework. During simulations, vehicle arriving rate on highway and on-ramp are both 0.2 vehicle per second. The performance is measured over the 200 meters on-ramp and the following 200 meters highway. The table shows that without the protection from our framework, the attack can lead to a 42.66% decrease in average speed and an 87.06% increase in CO emission. With our framework, the vehicles can take cautious actions, thus have a significantly higher travel speed and lower CO emission.

Table 5.1: Effectiveness of our framework in protecting against message spoofing attack in highway merging.

performance	without attack	under attack without ours	under attack with ours
average speed (m/s)	16.41	9.41	13.00
CO (mg)	741.06	1386.24	1006.36
fuel (mL)	41.17	41.91	42.96

5.3.2 Against Bad Mouthing Attack

Under the bad mouthing attack as shown in Fig. 5.1(b), all honest vehicles near the intersection are reported to have misbehavior by the attacker. Those honest vehicles may generate transactions to claim that they are innocent and being attacked. Before it is confirmed, all vehicles near the intersection will take cautious actions and choose not to trust others' messages; thus, the intelligent intersection may temporarily lose its functions. Vehicles will travel through the intersection as if there were all-way stop signs.

Fig. 5.3 shows the traffic in an intelligent intersection that is under attack. The vehicle arriving rate is 0.05 vehicle per second in the intersection. Around time 60 (seconds), the attacker starts the bad mouthing attack. However, the design of our framework ensures that the attack can be soon discovered and the system can return to normal shortly. Specifically, in one round of trust points blockchain, the surrounding vehicles have reached consensus on the existence of this attack. An honest vehicle will then trust messages from other honest vehicles, and the function of intelligent intersection gets recovered. From the figure, we can see that if our framework has a smaller round latency of $t_{lat} = 22$ seconds for the trust points blockchain, the system can recover before any significant increase of travel time. If the round latency reaches $t_{lat} = 60$ seconds, travel time starts increasing initially when the intelligent function is temporarily disabled, but starts decreasing around time 120 and fully recovers around time 140. More analysis of the impact of round latency

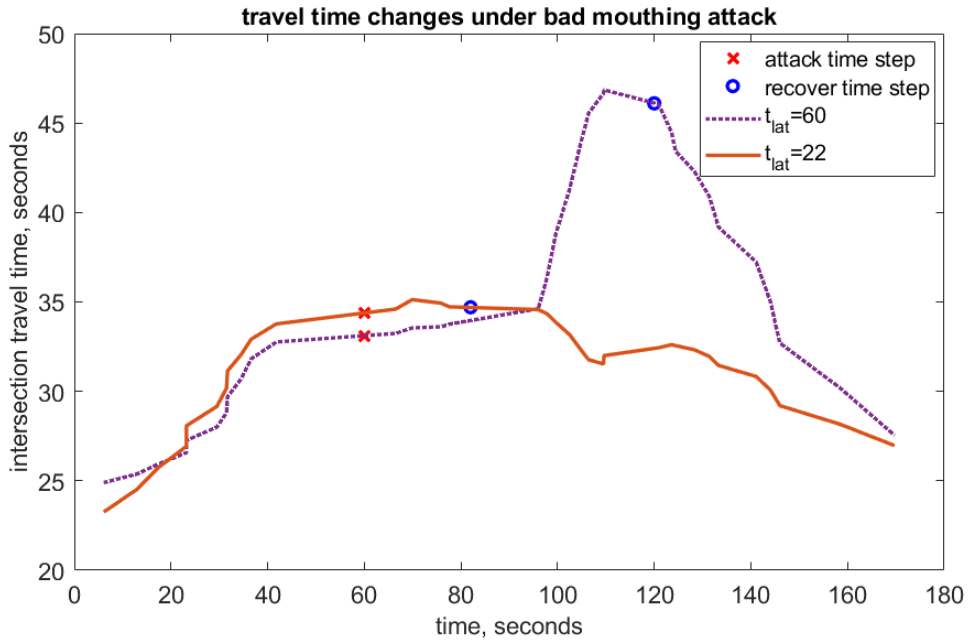

Figure 5.3: Effectiveness of our framework against bad mouthing attack in an intelligent intersection. The red stars denote the moment that the attack starts and the blue circles denote the moment that our framework detects the attack and the traffic system starts recovering from the attack. Travel time curves with different round latency ($t_{lat} = 22$ and 60 seconds) of the trust points blockchain are plotted.

is shown later in Section 5.3.4.

5.3.3 Against Sybil and Voting Attack

Fig. 5.1(c) shows the threat model where a traffic network with route choices is attacked with Sybil and voting attack. Specifically, an attack and the two Sybil vehicles it generates send transactions reporting a traffic accident in one route and win the voting process because they have more stake initially. Then other vehicles arriving at the area will have a higher probability (e.g., 0.9 in our simulations) to take another route, which leads to longer travel time. However, as several vehicles still choose the route with the fake accident and realize that it is a Sybil and voting attack, they

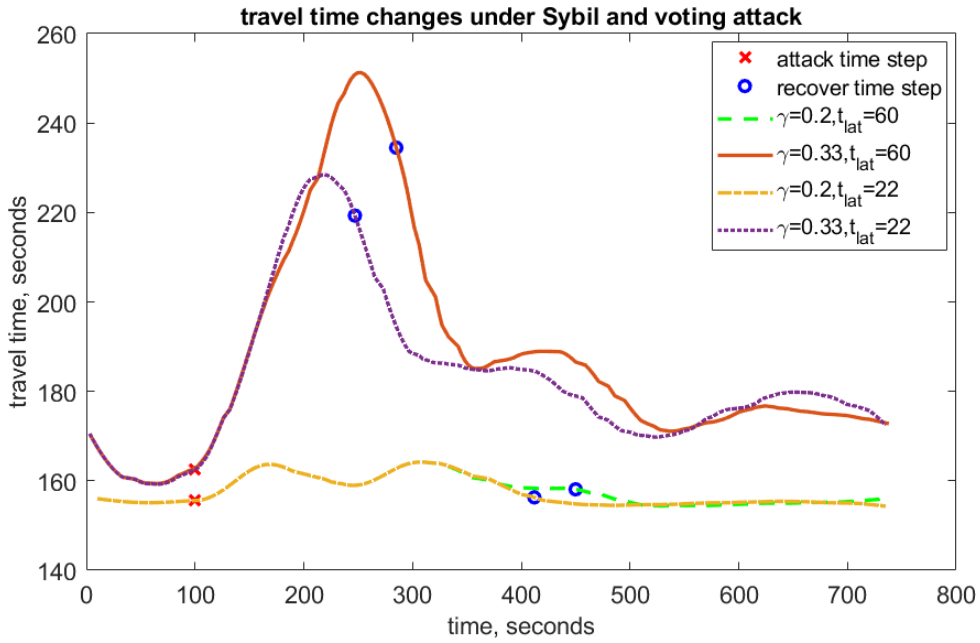

Figure 5.4: Effectiveness of our framework against Sybil and voting attack. Red stars denote the moment the attack starts, and the blue circles denote the moment the attack is detected and the recovery starts. Travel time curves with different round latency ($t_{lat} = 22$ and 60 seconds) for the trust points blockchain and under different vehicle arriving rates ($\gamma = 0.2$ and 0.33 vehicle per second) are plotted.

trigger the redressing process as the honest vehicles have higher stake now. Fig. 5.4 shows that our framework can soon discover the Sybil and voting attack, and recover to normal status. We also analyze the impact of vehicle arriving rate and round latency of trust points blockchain, and observe that a higher vehicle arriving rate leads to longer travel time under attack but faster recovery time in our framework. A smaller round latency may reduce the travel time under attack and also shorten the recovery time.

The results above demonstrate the effectiveness of our framework in protecting against message spoofing, bad mouthing, and Sybil and voting attacks. Note that with our proof-of-travel blockchain, it is even harder to perform these attacks (especially Sybil attack), as it takes more

effort for the malicious attackers to build up a travel record in the region with high stake.

5.3.4 Round Latency and Region Size

In the results above, we have seen the impact of round latency of the trust points blockchain on system performance. This latency is closely related to the region size. Here we conduct more in-depth analysis on the relationship between the two.

In [170] and [171], the authors simulated 50k Algorand users in the m4.2xlarge virtual machines on Amazon's EC2 platform, with 50 users per VM. The message transmission in Algorand is enabled by gossip protocol, which is similar to Bitcoin. The bandwidth for each Algorand process is set to 20 Mbps. Latency for one round of Algorand is about 22 seconds, dominated by the time to gossip a 1 MB block through the user network [171].

We assume that the blockchain will record all transactions sent by vehicles in a circular region with a radius of d_0. The size of a block is computed as:

$$S_b = (\frac{2\pi d_0 \beta_l \beta_c}{\beta_v} + \pi d_0^2 \beta_d \beta_t) t_{lat} S_t^{SCID} \tag{5.4}$$

where β_l is the number of lanes connect to outer space per perimeter of the region, β_c is the traffic capacity of a single lane with the unit as the number of vehicles traveled per hour per lane, β_v is the number of vehiwhere cles that can be claimed in a transaction of type SCID=0002, β_d is the density of vehicles in this region, β_t is the generation rate of transactions of type SCID=0000/0001, t_{lat} is the latency (period) to generate a block, S_t^{SCID} is the size of a transaction.

We set the capacity of lanes that connect to other regions, β_c, to be 3000 vehicles per hour per lane, as observed in the data collected from a section of the highway I-5 in Southern California [195] and should be similar in other regions. The density of vehicles β_d is set to 300 veh/km^2,

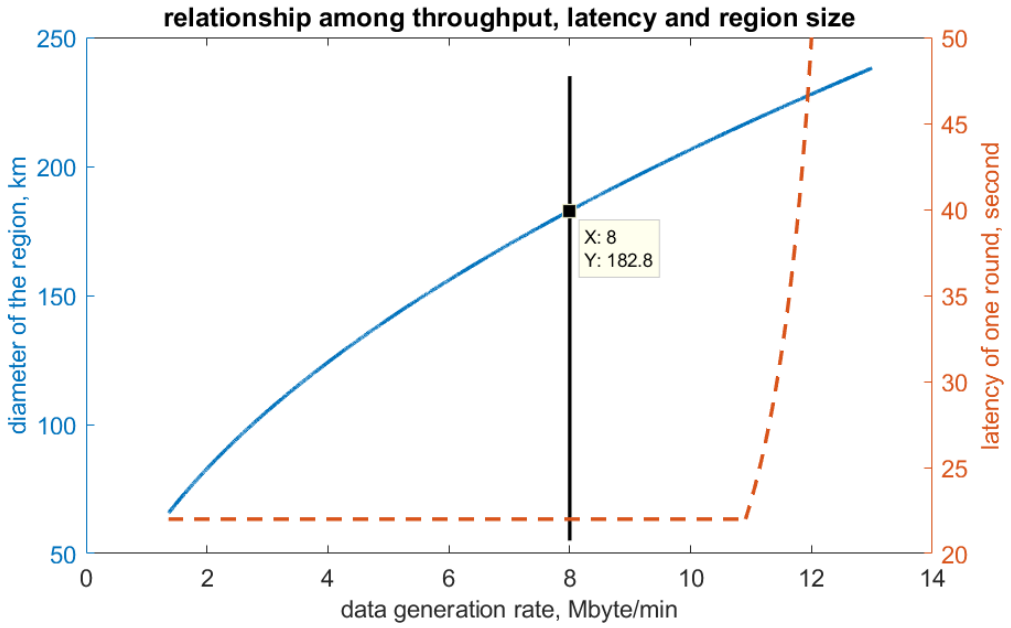

Figure 5.5: Maximum region size and minimum round latency under different transaction generation rate (best viewed in color). The solid blue line and the dashed orange line represent the maximum region size and the minimum round latency under different transaction generation rate, respectively. Drawing a vertical line, we can see the minimum round latency corresponding to the region size, e.g., 22 seconds for a region with a radius of 182.8 km.

which is similar to the density in Beijing [196]. Considering that attacking events are rare and vehicles are not driving on the road all the time, β_t is set to 0.05 transactions per hour per vehicle. S_t^{SCID} is 250 byte per transaction. β_v is set to 10 vehicles per transaction and β_l is 1 lane per km.

Taking a region with a similar size to Beijing, the transaction data generation rate is about 1.6 Mbyte/min. According to [171], the latency for one round of Algorand has little variation on the number of users. It basically remains the same when the block size is smaller than 4MB and then increases proportionally with a larger block size. We plot the relationship among throughput (data generation rate), minimum latency, and region size in Fig. 5.5, and we can clearly see the trade-offs between the minimum round latency and the maximum region size.

5.4 Resource Demand Analysis

We use three metrics to measure the resource demand of our framework on computation, communication, and storage cost. Here, the computation cost is the CPU resource used for running blockchain on each vehicle. Communication cost is measured by the total size of messages received per minute by each vehicle. Storage cost is measured by the total size of data maintained in our framework.

We leverage the results from Algorand in our analysis. The Algorand users use CPU resources mainly for verifying signatures and verifiable random functions (VRFs). Each Algorand process uses about 6.5% of a core averagely, in a 2.4 GHz Intel Xeon E5-2676 v3 (Haswell) Processor. While the on-board computing platform for vehicles may be different from the ones used in Algorand's analysis, we hope the overall trends shown in this section could still provide some valuable observations.

Next, we will analyze the three metrics for the trust points blockchain in Section 5.4.1, 5.4.2, and 5.4.3, and briefly discuss the resource demand for the proof-of-travel blockchain in Section 5.4.4.

5.4.1 Computation Cost

Note that the computation cost mainly depends on the verifier committee size, rather than the total amount of vehicles, because only block proposers and verifiers will broadcast messages. According to [171], the verifier committee size is directly computed from the fraction of honest vehicles h and a parameter F. Here we set $F = 5 \times 10e - 9$, denoting a negligible probability that the verifier committee reaches consensus on a falsified block [171]. By assuming that the computation cost is proportional to the number of messages received, we plot Fig. 5.6, which shows the relationship

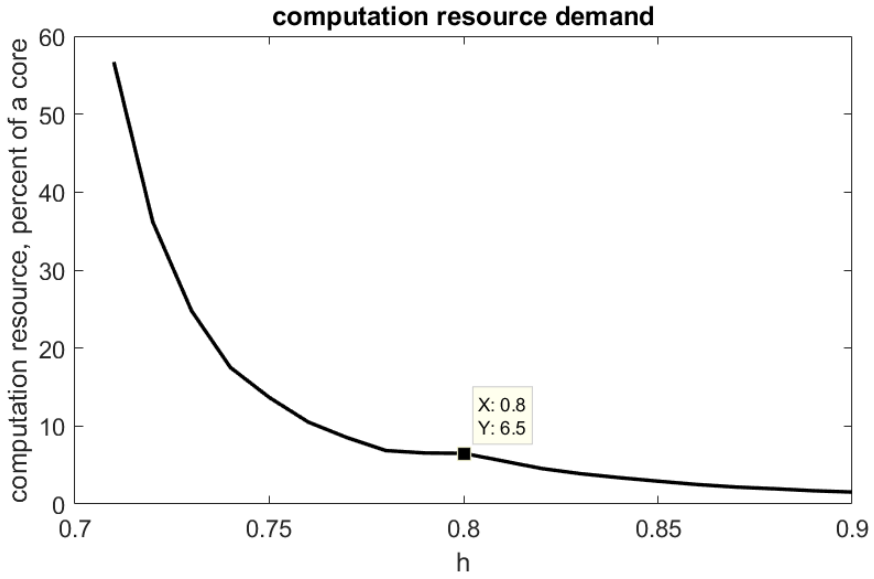

Figure 5.6: Computation resource demand under different fractions of honest vehicles.

between computation cost and the fraction of honest vehicles h. Here the computation cost is measured by the utility of one core in a 2.4 GHz Intel Xeon E5-2676 v3 (Haswell) Processor, e.g., it is 6.5 percent when 80 percent of the vehicles are honest.

5.4.2 Communication Cost

According to the consensus process described in [170] and [171], communication cost in one round is computed as

$$C_{cm} = S_b + S_c\eta_p + S_c\eta_v(\eta_s - 2) \tag{5.5}$$

where S_b and S_c are the size of a block and a short message for every step in the consensus process, respectively. As in [171], S_c is assumed to be no more than 200 bytes. Let η_p denote the number of block proposers in every round. All these proposers will broadcast a short message to claim their identity in Algorand. Let η_v and η_s denote the number of verifiers and the number of steps to reach consensus. Then $\eta_v(\eta_s - 2)$ is the number of messages sent by verifiers in one round because they

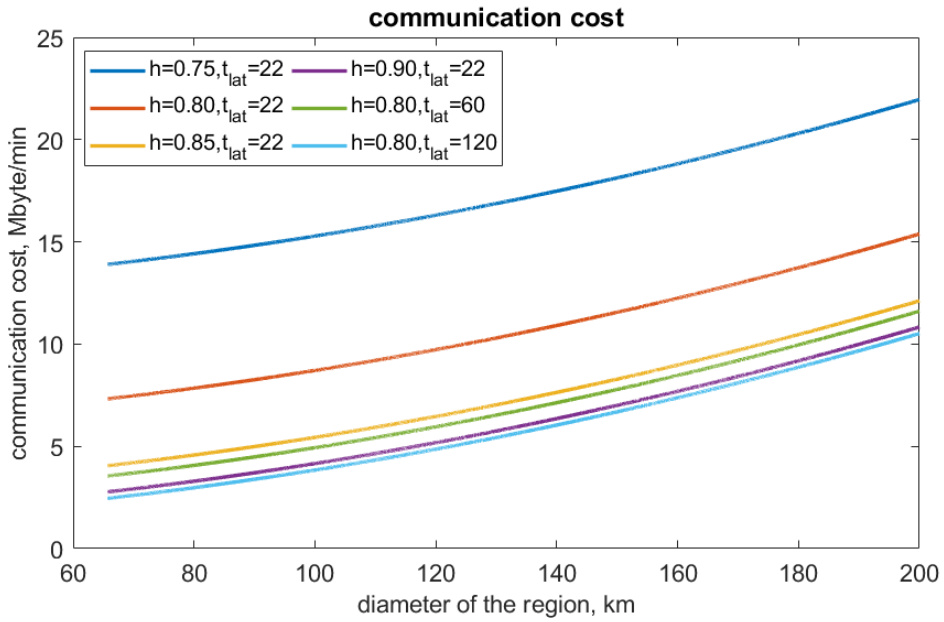

Figure 5.7: Communication cost under different fraction of honest vehicles (h), round latency (t_{lat}), and region size.

do not send messages in the first and last steps.

We plot communication cost under different fraction of honest vehicles, different round latency, and different region size in Fig. 5.7. We can see that the communication cost will increase with a smaller fraction of honest vehicles, a smaller round latency, or a larger region size.

5.4.3 Storage Cost

Storage resource needed by every vehicle for one-day data can be computed as

$$C_s^{1d} = \frac{24 \times 60}{t_{lat}\alpha_s} S_b \tag{5.6}$$

where $\frac{24\times60}{t_{lat}}$ is the number of blocks generated every day, and α_s is a sharding parameter. For N shards, vehicles only store blocks whose round number equals their public key modulo N, so that

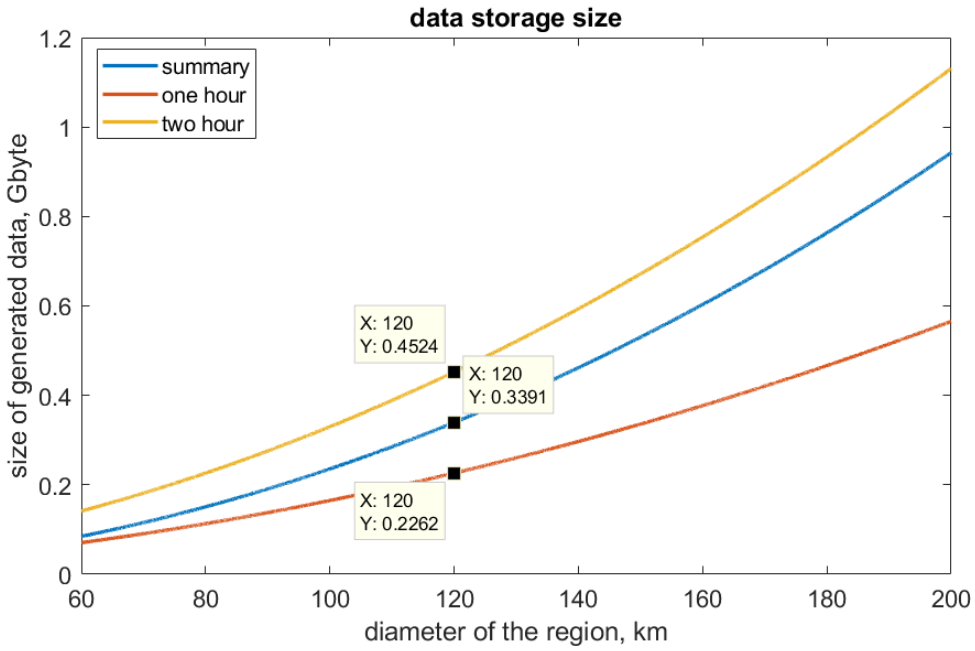

Figure 5.8: Storage cost under different region size. We can see that the summary of all vehicles' records has a size that is between the sizes of transactions generated in one hour and in two hours.

storage cost can be reduced by the divisor of α_s.

As we mentioned in Section 5.3.4, for a region with the similar size to Beijing, the blockchain generates 1.6 Mbyte data per minute, which means 69.12 Gbyte per month. Even if we leverage the sharding technique with $\alpha_s = 10$, blockchain users will still have to maintain about 7 Gbyte of new data every month. Moreover, the sharding technique will not help new vehicles. A new vehicle will have to download all the data to get the correct status of all vehicles.

To overcome this problem, we propose a new summary step for blockchain. In our design, selected block proposers will summarize the status of all vehicles in new blocks with a period of t_{sum}. By indicating the type of block in its header, a new vehicle can easily find the latest summary blocks and do not need to download blocks generated before those.

In this way, without considering the sharding technique, storage resource demand C_s is

$$C_s = S_b + t_{sum} C_s^{1d} \tag{5.7}$$

where S_b is the data size of summary blocks, as computed below:

$$S_b = \pi d_0^2 \beta_d S_u \tag{5.8}$$

where S_u denotes the size of data for summarizing one vehicle's status. As shown in Fig. 5.8, the size of these summary blocks is between the size of transactions generated in one hour and the size of transactions generated in two hours.

These summary blocks are communication overhead. A smaller t_{sum} means larger communication overhead, smaller storage cost, and smaller size of data a new vehicle needs to download to join the blockchain. Taking a region with a radius of 120 km for example and reading data from Fig. 5.8, with a period of $t_{sum} = 24$ hours, the communication overhead would be about $\frac{0.339}{0.2263 \times 24} = 6.24\%$, storage cost would only be $24 \times 0.2263 + 0.339 = 5.77$ Gbyte. That is, a new vehicle may only need to download 5.77 Gbyte data to get the latest status of blockchain.

5.4.4 Resource Demand for Proof-of-Travel Blockchain

To alleviate resource demand for the proof-of-travel blockchain, vehicles can generate blocks and update the blockchain in a much larger period than that of the trust points blockchain, e.g., one day or longer. Considering that vehicles usually stay most time in their permanent address/region, proof-of-travel blockchain is maintained by vehicles in the same permanent region to mitigate communication overhead.

For example, for a region with a 10 km radius, there are about 100 thousand registered vehicles

under density β_d. We assume that a vehicle can record at most 500 vehicles with higher credits in its transaction, and the updating period is one day. The size of one transaction is estimated as $20 \times 500 = 10k$ bytes if each vehicle record consumes 20 bytes. The size of aggregated transactions will be $100k \times 10k = 1G$ bytes. After arrangement and inter-region communication, the final block should have a much smaller size. In this way, with a block generating period of one day or longer, the resource demand for the proof-of-travel blockchain should be much less than that for the trust points blockchain.

CHAPTER 6
CONCLUSION AND FUTURE WORK

6.1 Conclusion

In this dissertation, we discussed safety and security challenges in connected and autonomous vehicles and presented our solutions to address some of these challenges.

Autonomous driving is safety-critical. To ensure safety, we proposed a hierarchical neural network-based planner design based on the underlying physical scenarios of the system, and developed novel overapproximation techniques for the reachability analysis of such hierarchical design. Through theoretical analysis, we showed that our hierarchical design can improve the safety and verifiability for systems that may evolve into different physical scenarios, compared with single neural network-based planners. Through two case studies of unprotected left turn and highway merging in autonomous driving, we further demonstrate such advantages empirically.

To prevent over-conservative planning while ensuring safety, we proposed a novel safety-driven interactive neural network-based planning framework. The framework includes a safety-driven behavior adjustment module for safety assurance and an aggressiveness assessment module for avoiding over-conservative planning. Extensive experiments on synthetic examples and real-world challenging scenarios demonstrated the effectiveness of our approach in improving system safety. Furthermore, we improved the planning framework by incorporating connectivity for mixed traffic scenarios. The framework can significantly improve performance by coordinating with surrounding connected vehicles in dynamic environment. Extensive experiments demonstrated the strength of our planner design in improving performance while ensuring safety. Our experiments sug-

gest that (1) connectivity of the immediate following vehicle plays a more important role for ego vehicle's lane changing than the connectivity of leading vehicles, and (2) when there are more connected leading vehicles, the system performance can be further improved because more vehicles can coordinate to leave larger space for the ego vehicle. We also demonstrated the system robustness under different extent of promise violation rate of surrounding connected vehicles.

For those challenging scenarios that surrounding vehicles' behaviors and trajectories cannot be accurately predicted, we proposed a speculative planning framework with adaptive prediction to ensure safety and improve performance. Our method considers all possible predicted behaviors and trajectories, ensures system safety by ruling out actions that may be unsafe in the worst case, and improves system performance by sampling all possibilities and choosing the action that maximizes the expected reward. By adapting the prediction results according to the system states, impossible behaviors and trajectories of surrounding vehicles are filtered out, thus leading to more effective planning. Through the case study of lane changing in a multi-lane highway, we demonstrated the advantages of our proposed planner over various baseline methods.

To secure communication for CV applications, we proposed a dual cyber-physical blockchains framework that includes a trust points blockchain and a proof-of-travel blockchain for building trust. The trust points blockchain has a quick response to suspicious behavior, with smart contracts designed for instant voting, redressing, and transferring records. The proof-of-travel blockchain builds up reputations from vehicles' long-term travel records. The trust points and the proof-of-travel credits are used together to compute the vehicle stake for running the consensus mechanisms in both blockchains. Experimental results demonstrated the effectiveness of our framework in protection against message spoofing, bad mouthing, and Sybil and voting attacks, in representative CV applications. We also conducted a preliminary analysis on the framework's resource demand.

6.2 Future Work

Our work addressed some safety and security challenges under certain assumptions. However, the wide adoption of autonomous driving is still facing lots of challenges.

There is an urgent need for more accurate and general models of surrounding traffic participants in various environments. It is the key to ensuring safety and improving performance. Current state-of-the-art methods usually assume the prediction results reflect the true driving behaviors and trajectories. However, this assumption may not hold all the time. A promising solution is to build a prediction model based on traffic rules and typical driving norms, and to ensure safety when others obey traffic rules and driving norms.

Scalability is important for wide adoption of these techniques. We expect the cost to be low for adapting the machine learning-based perception, prediction, planning models and conducting corresponding verification processes in different environments. Demonstrating the scalability of the technique should be an interesting and critical direction.

There is doubt on whether it is appropriate to focus on Level 4 or Level 5 autonomous driving currently. Because there are so many corner cases, which may take a long time to be learned by neural network-based models. There is always a trade-off between safety and performance. Some interesting questions could be: if safety is ensured, how is the performance of current state-of-the-art techniques? What is the performance threshold to adopt it widely without significantly interfering with human drivers?

The realistic traffic environment is much more complex than our assumptions. Extending our work to those complex scenarios with more traffic participants, and finding backup solutions for others' unexpected behaviors are also interesting directions.

REFERENCES

[1] Z. Zhang, R. Tian, R. Sherony, J. Domeyer, and Z. Ding, "Attention-based interrelation modeling for explainable automated driving," *IEEE Transactions on Intelligent Vehicles*, 2022.

[2] J. B. Kenney, "Dedicated short-range communications (DSRC) standards in the United States," *Proceedings of the IEEE*, vol. 99, no. 7, pp. 1162–1182, 2011.

[3] M. H. C. Garcia, A. Molina-Galan, M. Boban, J. Gozalvez, B. Coll-Perales, T. Şahin, and A. Kousaridas, "A tutorial on 5g nr v2x communications," *IEEE Communications Surveys & Tutorials*, 2021.

[4] Z. Wang, G. Wu, K. Boriboonsomsin, M. J. Barth, K. Han, B. Kim, and P. Tiwari, "Cooperative ramp merging system: Agent-based modeling and simulation using game engine," *SAE International Journal of Connected and Automated Vehicles*, vol. 2, no. 2, 2019.

[5] B. Zheng, C.-W. Lin, S. Shiraishi, and Q. Zhu, "Design and analysis of delay-tolerant intelligent intersection management," *ACM Transactions on Cyber-Physical Systems*, vol. 4, no. 1, pp. 1–27, 2019.

[6] S. Stock, M. Mansouri, F. Pecora, and J. Hertzberg, "Online task merging with a hierarchical hybrid task planner for mobile service robots," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2015, pp. 6459–6464.

[7] Z. Cao, E. Bıyık, W. Z. Wang, A. Raventos, A. Gaidon, G. Rosman, and D. Sadigh, "Reinforcement learning based control of imitative policies for near-accident driving," *arXiv preprint arXiv:2007.00178*, 2020.

[8] G. Pettet, A. Mukhopadhyay, M. Kochenderfer, and A. Dubey, "Hierarchical planning for resource allocation in emergency response systems," *arXiv preprint arXiv:2012.13300*, 2020.

[9] D. D. Fan, J. Nguyen, R. Thakker, N. Alatur, A.-a. Agha-mohammadi, and E. A. Theodorou, "Bayesian learning-based adaptive control for safety critical systems," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2020, pp. 4093–4099.

[10] X. Liu, C. Huang, Y. Wang, B. Zheng, and Q. Zhu, "Physics-aware safety-assured design of hierarchical neural network based planner," *2022 ACM/IEEE 13th International Conference on Cyber-Physical Systems (ICCPS)*, pp. 137–146, 2022.

[11] W. Zhan, L. Sun, D. Wang, H. Shi, A. Clausse, M. Naumann, J. Kummerle, H. Konigshof, C. Stiller, A. de La Fortelle, *et al.*, "Interaction dataset: An international, adversarial and cooperative motion dataset in interactive driving scenarios with semantic maps," *arXiv preprint arXiv:1910.03088*, 2019.

[12] R. Jiao, X. Liu, T. Sato, Q. A. Chen, and Q. Zhu, "Semi-supervised semantics-guided adversarial training for trajectory prediction," *arXiv preprint arXiv:2205.14230*, 2022.

[13] Z. Zhang, D. Shen, R. Tian, L. Li, Y. Chen, J. Sturdevant, and E. Cox, "Implementation and performance evaluation of in-vehicle highway back-of-queue alerting system using the driving simulator," in *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, IEEE, 2021, pp. 1753–1759.

[14] Z. Zhang, R. Tian, V. G. Duffy, and L. Li, "The comfort of the soft-safety driver alerts: Measurements and evaluation," *International Journal of Human–Computer Interaction*, pp. 1–11, 2022.

[15] M. Cheng, C. Yin, J. Zhang, S. Nazarian, J. Deshmukh, and P. Bogdan, "A general trust framework for multi-agent systems," in *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, 2021, pp. 332–340.

[16] H. Yu, H. E. Tseng, and R. Langari, "A human-like game theory-based controller for automatic lane changing," *Transportation Research Part C: Emerging Technologies*, vol. 88, pp. 140–158, 2018.

[17] P.-Y. Huang, K.-W. Liu, Z.-L. Li, S. Park, E. Andert, C.-W. Lin, and A. Shrivastava, "Compatibility checking for autonomous lane-changing assistance systems," in *2022 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, IEEE, 2022, pp. 1161–1164.

[18] X. Liu, N. Masoud, Q. Zhu, and A. Khojandi, "A markov decision process framework to incorporate network-level data in motion planning for connected and automated vehicles," *Transportation Research Part C: Emerging Technologies*, vol. 136, p. 103 550, 2022.

[19] M. Khayatian, M. Mehrabian, and A. Shrivastava, "Rim: Robust intersection management for connected autonomous vehicles," in *2018 IEEE Real-Time Systems Symposium (RTSS)*, IEEE, 2018, pp. 35–44.

[20] Y. Fu, C. Li, T. H. Luan, Y. Zhang, and G. Mao, "Infrastructure-cooperative algorithm for effective intersection collision avoidance," *Transportation research part C: emerging technologies*, vol. 89, pp. 188–204, 2018.

[21] C. Pek, S. Manzinger, M. Koschi, and M. Althoff, "Using online verification to prevent autonomous vehicles from causing accidents," *Nature Machine Intelligence*, vol. 2, no. 9, pp. 518–528, 2020.

[22] S. Shalev-Shwartz, S. Shammah, and A. Shashua, "On a formal model of safe and scalable self-driving cars," *arXiv preprint arXiv:1708.06374*, 2017.

[23] Z. Li, W. Zhan, L. Sun, C.-Y. Chan, and M. Tomizuka, "Adaptive sampling-based motion planning with a non-conservatively defensive strategy for autonomous driving," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 15 632–15 638, 2020.

[24] X. Liu, R. Jiao, B. Zheng, D. Liang, and Q. Zhu, "Neural network based interactive lane changing planner in dense traffic with safety guarantee," *arXiv preprint arXiv:2201.09112*, 2022.

[25] X. Liu, R. Jiao, B. Zheng, D. Liang, and Q. Zhu, "Safety-driven interactive planning for neural network-based lane changing," in *Proceedings of the 28th Asia and South Pacific Design Automation Conference*, 2023, pp. 39–45.

[26] H. Cui, V. Radosavljevic, F.-C. Chou, T.-H. Lin, T. Nguyen, T.-K. Huang, J. Schneider, and N. Djuric, "Multimodal trajectory predictions for autonomous driving using deep convolutional networks," in *2019 International Conference on Robotics and Automation (ICRA)*, IEEE, 2019, pp. 2090–2096.

[27] M. Liang, B. Yang, R. Hu, Y. Chen, R. Liao, S. Feng, and R. Urtasun, "Learning lane graph representations for motion forecasting," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, Springer, 2020, pp. 541–556.

[28] H. Zhao, J. Gao, T. Lan, C. Sun, B. Sapp, B. Varadarajan, Y. Shen, Y. Shen, Y. Chai, C. Schmid, *et al.*, "Tnt: Target-driven trajectory prediction," in *Conference on Robot Learning*, PMLR, 2021, pp. 895–904.

[29] J. Gu, C. Sun, and H. Zhao, "Densetnt: End-to-end trajectory prediction from dense goal sets," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 15 303–15 312.

[30] Y. Luo, Y. Xiang, K. Cao, and K. Li, "A dynamic automated lane change maneuver based on vehicle-to-vehicle communication," *Transportation Research Part C: Emerging Technologies*, vol. 62, pp. 87–102, 2016.

[31] A. Li, L. Sun, W. Zhan, M. Tomizuka, and M. Chen, "Prediction-based reachability for collision avoidance in autonomous driving," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2021, pp. 7908–7914.

[32] W. Zhan, C. Liu, C.-Y. Chan, and M. Tomizuka, "A non-conservatively defensive strategy for urban autonomous driving," in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2016, pp. 459–464.

[33] B. Luo, X. Liu, and Q. Zhu, "Credibility enhanced temporal graph convolutional network based sybil attack detection on edge computing servers," in *2021 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2021, pp. 524–531.

[34] X. Liu, Y. Luo, A. Goeckner, T. Chakraborty, R. Jiao, N. Wang, Y. Wang, T. Sato, Q. A. Chen, and Q. Zhu, "Invited: Waving the double-edged sword: Building resilient cavs with edge and cloud computing," in *Proceedings of the 60th Annual Design Automation Conference*, 2023, pp. 1–4.

[35] Q. A. Chen, Y. Yin, Y. Feng, Z. M. Mao, and H. X. Liu, "Exposing congestion attack on emerging connected vehicle based traffic signal control.," in *NDSS*, 2018.

[36] A. Abdo, S. M. B. Malek, Z. Qian, Q. Zhu, M. Barth, and N. Abu-Ghazaleh, "Application level attacks on connected vehicle protocols," in *22nd International Symposium on Research in Attacks, Intrusions and Defenses ({RAID} 2019)*, 2019, pp. 459–471.

[37] S. Parkinson, P. Ward, K. Wilson, and J. Miller, "Cyber threats facing autonomous and connected vehicles: Future challenges," *IEEE transactions on intelligent transportation systems*, vol. 18, no. 11, pp. 2898–2915, 2017.

[38] S. Gurung, D. Lin, A. Squicciarini, and E. Bertino, "Information-oriented trustworthiness evaluation in vehicular ad-hoc networks," in *International Conference on Network and System Security*, Springer, 2013, pp. 94–108.

[39] M. Raya, P. Papadimitratos, V. D. Gligor, and J.-P. Hubaux, "On data-centric trust establishment in ephemeral ad hoc networks," in *IEEE INFOCOM 2008-The 27th Conference on Computer Communications*, IEEE, 2008, pp. 1238–1246.

[40] J. Kolb, M. AbdelBaky, R. H. Katz, and D. E. Culler, "Core concepts, challenges, and future directions in blockchain: A centralized tutorial," *ACM Computing Surveys (CSUR)*, vol. 53, no. 1, pp. 1–39, 2020.

[41] R. Iqbal, T. A. Butt, M. Afzaal, and K. Salah, "Trust management in social internet of vehicles: Factors, challenges, blockchain, and fog solutions," *International Journal of Distributed Sensor Networks*, vol. 15, no. 1, p. 1 550 147 719 825 820, 2019.

[42] M. Singh and S. Kim, "Branch based blockchain technology in intelligent vehicle," *Computer Networks*, vol. 145, pp. 219–231, 2018.

[43] Z. Yang, K. Yang, L. Lei, K. Zheng, and V. C. Leung, "Blockchain-based decentralized trust management in vehicular networks," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1495–1505, 2018.

[44] Z. Zhang, R. Tian, and V. G. Duffy, "Trust in automated vehicle: A meta-analysis," in *Human-Automation Interaction: Transportation*, Springer, 2022, pp. 221–234.

[45] Z. Zhang, V. G. Duffy, and R. Tian, "Trust and automation: A systematic review and bibliometric analysis," in *HCI International 2021-Late Breaking Papers: Design and User Experience: 23rd HCI International Conference, HCII 2021, Virtual Event, July 24–29, 2021, Proceedings 23*, Springer, 2021, pp. 451–464.

[46] W. Farag, "Complex trajectory tracking using pid control for autonomous driving," *International Journal of Intelligent Transportation Systems Research*, pp. 1–11, 2019.

[47] X. Liu, G. Zhao, N. Masoud, and Q. Zhu, "Trajectory planning for connected and automated vehicles: Cruising, lane changing, and platooning," *SAE International Journal of Connected and Automated Vehicles*, vol. 4, no. 12-04-04-0025, pp. 315–333, 2021.

[48] X. Liu, N. Masoud, and Q. Zhu, "Impact of sharing driving attitude information: A quantitative study on lane changing," in *2020 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2020, pp. 1998–2005.

[49] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Transactions on intelligent vehicles*, vol. 1, no. 1, pp. 33–55, 2016.

[50] L. Claussmann, M. Revilloud, D. Gruyer, and S. Glaser, "A review of motion planning for highway autonomous driving," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 5, pp. 1826–1848, 2019.

[51] F. Gritschneder, K. Graichen, and K. Dietmayer, "Fast trajectory planning for automated vehicles using gradient-based nonlinear model predictive control," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2018, pp. 7369–7374.

[52] C. Katrakazas, M. Quddus, W.-H. Chen, and L. Deka, "Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions," *Transportation Research Part C: Emerging Technologies*, vol. 60, pp. 416–442, 2015.

[53] X. Zeng and J. Wang, "Globally energy-optimal speed planning for road vehicles on a given route," *Transportation Research Part C: Emerging Technologies*, vol. 93, pp. 148–160, 2018.

[54] G. De Nunzio, C. C. De Wit, P. Moulin, and D. Di Domenico, "Eco-driving in urban traffic networks using traffic signals information," *International Journal of Robust and Nonlinear Control*, vol. 26, no. 6, pp. 1307–1324, 2016.

[55] H. Rakha and R. K. Kamalanathsharma, "Eco-driving at signalized intersections using v2i communication," in *2011 14th international IEEE conference on intelligent transportation systems (ITSC)*, IEEE, 2011, pp. 341–346.

[56] B. Li, Y. Zhang, Y. Feng, Y. Zhang, Y. Ge, and Z. Shao, "Balancing computation speed and quality: A decentralized motion planning method for cooperative lane changes of connected and automated vehicles," *IEEE Transactions on Intelligent Vehicles*, vol. 3, no. 3, pp. 340–350, 2018.

[57] C. Liu, C.-Y. Lin, and M. Tomizuka, "The convex feasible set algorithm for real time optimization in motion planning," *SIAM Journal on Control and optimization*, vol. 56, no. 4, pp. 2712–2733, 2018.

[58] L. Zhang, J. Sun, and G. Orosz, "Hierarchical design of connected cruise control in the presence of information delays and uncertain vehicle dynamics," *IEEE Transactions on Control Systems Technology*, vol. 26, no. 1, pp. 139–150, 2017.

[59] G. Orosz, "Connected cruise control: Modelling, delay effects, and nonlinear behaviour," *Vehicle System Dynamics*, vol. 54, no. 8, pp. 1147–1176, 2016.

[60] J. Hardy and M. Campbell, "Contingency planning over probabilistic obstacle predictions for autonomous road vehicles," *IEEE Transactions on Robotics*, vol. 29, no. 4, pp. 913–929, 2013.

[61] E. Galceran, A. G. Cunningham, R. M. Eustice, and E. Olson, "Multipolicy decision-making for autonomous driving via changepoint-based behavior prediction.," in *Robotics: Science and Systems*, vol. 1, 2015.

[62] S. Brechtel, T. Gindele, and R. Dillmann, "Probabilistic decision-making under uncertainty for autonomous driving using continuous pomdps," in *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2014, pp. 392–399.

[63] D. Yang, S. Zheng, C. Wen, P. J. Jin, and B. Ran, "A dynamic lane-changing trajectory planning model for automated vehicles," *Transportation Research Part C: Emerging Technologies*, vol. 95, pp. 228–247, 2018.

[64] S. Sivaraman and M. M. Trivedi, "Dynamic probabilistic drivability maps for lane change and merge driver assistance," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 5, pp. 2063–2073, 2014.

[65] Y. Wu, A. Xiao, H. Chen, S. Zhang, and Y. Liu, "Amphibious robot's trajectory tracking with dnn-based nonlinear model predictive control," in *2020 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, IEEE, 2020, pp. 2019–2024.

[66] B. Xia, H. Luan, Z. Zhao, X. Gao, P. Xie, A. Xiao, J. Wang, and M. Q.-H. Meng, "Collaborative trolley transportation system with autonomous nonholonomic robots," *arXiv preprint arXiv:2303.06624*, 2023.

[67] Y. Chen, Z. Xu, Z. Jian, G. Tang, Y. Yangli, A. Xiao, X. Wang, and B. Liang, "Quadruped guidance robot for the visually impaired: A comfort-based approach," *arXiv preprint arXiv:2203.03927*, 2022.

[68] Z. Jian, Z. Lu, X. Zhou, B. Lan, A. Xiao, X. Wang, and B. Liang, "Putn: A plane-fitting based uneven terrain navigation framework," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2022, pp. 7160–7166.

[69] Y. Tassa, N. Mansard, and E. Todorov, "Control-limited differential dynamic programming," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2014, pp. 1168–1175.

[70] J. van den Berg, "Iterated lqr smoothing for locally-optimal feedback control of systems with non-linear dynamics and non-quadratic cost," in *2014 American Control Conference*, IEEE, 2014, pp. 1912–1918.

[71] J. Chen, W. Zhan, and M. Tomizuka, "Constrained iterative lqr for on-road autonomous driving motion planning," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2017, pp. 1–7.

[72] J. Chen, W. Zhan, and M. Tomizuka, "Autonomous driving motion planning with constrained iterative lqr," *IEEE Transactions on Intelligent Vehicles*, vol. 4, no. 2, pp. 244–254, 2019.

[73] A. Talebpour, H. S. Mahmassani, and S. H. Hamdar, "Modeling lane-changing behavior in a connected environment: A game theory approach," *Transportation Research Procedia*, vol. 7, pp. 420–440, 2015.

[74] C. Burger, T. Schneider, and M. Lauer, "Interaction aware cooperative trajectory planning for lane change maneuvers in dense traffic," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2020, pp. 1–8.

[75] D. Isele, "Interactive decision making for autonomous vehicles in dense traffic," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, IEEE, 2019, pp. 3981–3986.

[76] X. Liu, J. Chen, S. Li, Y. Zhang, H. Yu, F. Huang, J. Liu, C. Wang, L. Li, and Q. Zhu, "Interactive trajectory planner for mandatory lane changing in dense non-cooperative traffic," *arXiv preprint arXiv:2303.02309*, 2023.

[77] E. Zhang, N. Masoud, M. Bandegi, and R. K. Malhan, "Predicting risky driving in a connected vehicle environment," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 10, pp. 17 177–17 188, 2022.

[78] E. Zhang, R. Zhang, and N. Masoud, "Predictive trajectory planning for autonomous vehicles at intersections using reinforcement learning," *Transportation Research Part C: Emerging Technologies*, vol. 149, p. 104 063, 2023.

[79] T. Awal, M. Murshed, and M. Ali, "An efficient cooperative lane-changing algorithm for sensor-and communication-enabled automated vehicles," in *2015 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2015, pp. 1328–1333.

[80] M. Bouton, A. Nakhaei, K. Fujimura, and M. J. Kochenderfer, "Cooperation-aware reinforcement learning for merging in dense traffic," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, IEEE, 2019, pp. 3441–3447.

[81] S. Bae, D. Saxena, A. Nakhaei, C. Choi, K. Fujimura, and S. Moura, "Cooperation-aware lane change maneuver in dense traffic based on model predictive control with recurrent neural network," in *2020 American Control Conference (ACC)*, IEEE, 2020, pp. 1209–1216.

[82] Z. Zhang, R. Tian, and Z. Ding, "Trep: Transformer-based evidential prediction for pedestrian intention with uncertainty," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, 2023.

[83] A. Ladino and M. Wang, "A dynamic game formulation for cooperative lane change strategies at highway merges," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 15 059–15 064, 2020.

[84] Y. Hou, P. Edara, and C. Sun, "Modeling mandatory lane changing using bayes classifier and decision trees," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 2, pp. 647–655, 2013.

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

[86] M. Bouton, A. Nakhaei, D. Isele, K. Fujimura, and M. J. Kochenderfer, "Reinforcement learning with iterative reasoning for merging in dense traffic," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2020, pp. 1–6.

[87] R. Zhang, S. Masoud, and N. Masoud, "Impact of autonomous vehicles on the car-following behavior of human drivers," *Journal of Transportation Engineering, Part A: Systems*, vol. 149, no. 3, p. 04 022 152, 2023.

[88] D. P. Losey and D. Sadigh, "Robots that take advantage of human trust," *arXiv preprint arXiv:1909.05777*, 2019.

[89] E. Bıyık, D. Lazar, R. Pedarsani, and D. Sadigh, "Altruistic autonomy: Beating congestion on shared roads," *arXiv preprint arXiv:1810.11978*, 2018.

[90] R. Tian, S. Li, N. Li, I. Kolmanovsky, A. Girard, and Y. Yildiz, "Adaptive game-theoretic decision making for autonomous vehicle control at roundabouts," in *2018 IEEE Conference on Decision and Control (CDC)*, IEEE, 2018, pp. 321–326.

[91] J. F. Fisac, E. Bronstein, E. Stefansson, D. Sadigh, S. S. Sastry, and A. D. Dragan, "Hierarchical game-theoretic planning for autonomous vehicles," in *2019 International Conference on Robotics and Automation (ICRA)*, IEEE, 2019, pp. 9590–9596.

[92] E. Stefansson, J. F. Fisac, D. Sadigh, S. S. Sastry, and K. H. Johansson, "Human-robot interaction for truck platooning using hierarchical dynamic games," in *2019 18th European Control Conference (ECC)*, IEEE, 2019, pp. 3165–3172.

[93] D. Sadigh, S. Sastry, S. A. Seshia, and A. D. Dragan, "Planning for autonomous cars that leverage effects on human actions.," in *Robotics: Science and Systems*, Ann Arbor, MI, USA, vol. 2, 2016.

[94] A. D. Dragan, "Robot planning with mathematical models of human state and action," *arXiv preprint arXiv:1705.04226*, 2017.

[95] P. Wang, C.-Y. Chan, and A. de La Fortelle, "A reinforcement learning based approach for automated lane change maneuvers," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2018, pp. 1379–1384.

[96] B. Mirchevska, C. Pek, M. Werling, M. Althoff, and J. Boedecker, "High-level decision making for safe and reasonable autonomous lane changing using reinforcement learning," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2018, pp. 2156–2162.

[97] Z. Yan, K. Yang, Z. Wang, B. Yang, T. Kaizuka, and K. Nakano, "Time to lane change and completion prediction based on gated recurrent unit network," in *2019 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2019, pp. 102–107.

[98] Z. Cao, D. Yang, S. Xu, H. Peng, B. Li, S. Feng, and D. Zhao, "Highway exiting planner for automated vehicles using reinforcement learning," *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[99] J. Chen, B. Yuan, and M. Tomizuka, "Deep imitation learning for autonomous driving in generic urban scenarios with enhanced safety," *arXiv preprint arXiv:1903.00640*, 2019.

[100] L. Markolf, J. Eilbrecht, and O. Stursberg, "Trajectory planning for autonomous vehicles combining nonlinear optimal control and supervised learning," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 15 608–15 614, 2020.

[101] S. Zheng, Y. Yue, and P. Lucey, "Generating long-term trajectories using deep hierarchical networks," *arXiv preprint arXiv:1706.07138*, 2017.

[102] L. Sun, W. Zhan, C.-Y. Chan, and M. Tomizuka, "Behavior planning of autonomous cars with social perception," in *2019 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2019, pp. 207–213.

[103] J. Li, L. Sun, W. Zhan, and M. Tomizuka, "Interaction-aware behavior planning for autonomous vehicles validated with real traffic data," in *Dynamic Systems and Control Conference*, American Society of Mechanical Engineers, vol. 84287, 2020, V002T31A005.

[104] F. Ye, X. Cheng, P. Wang, C.-Y. Chan, and J. Zhang, "Automated lane change strategy using proximal policy optimization-based deep reinforcement learning," in *2020 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2020, pp. 1746–1752.

[105] A. Alizadeh, M. Moghadam, Y. Bicer, N. K. Ure, U. Yavas, and C. Kurtulus, "Automated lane change decision making using deep reinforcement learning in dynamic and uncertain highway environment," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, IEEE, 2019, pp. 1399–1404.

[106] Y. Chen, C. Dong, P. Palanisamy, P. Mudalige, K. Muelling, and J. M. Dolan, "Attention-based hierarchical deep reinforcement learning for lane change behaviors in autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 1–9.

[107] T. Shi, P. Wang, X. Cheng, C.-Y. Chan, and D. Huang, "Driving decision and control for automated lane change behavior based on deep reinforcement learning," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, IEEE, 2019, pp. 2895–2900.

[108] P. F. Orzechowski, A. Meyer, and M. Lauer, "Tackling occlusions & limited sensor range with set-based safety verification," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2018, pp. 1729–1736.

[109] D. Tian, G. Wu, P. Hao, K. Boriboonsomsin, and M. J. Barth, "Connected vehicle-based lane selection assistance application," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 7, pp. 2630–2643, 2018.

[110] Y. Xing, C. Lv, and D. Cao, "Personalized vehicle trajectory prediction based on joint time-series modeling for connected vehicles," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 2, pp. 1341–1352, 2019.

[111] J. Park, M. Abdel-Aty, Y. Wu, and I. Mattei, "Enhancing in-vehicle driving assistance information under connected vehicle environment," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 9, pp. 3558–3567, 2018.

[112] S. Aoki, T. Higuchi, and O. Altintas, "Cooperative perception with deep reinforcement learning for connected vehicles," in *2020 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2020, pp. 328–334.

[113] Y. Jin, X. Liu, and Q. Zhu, "Dsrc & c-v2x comparison for connected and automated vehicles in different traffic scenarios," *arXiv preprint arXiv:2203.12553*, 2022.

[114] J. Dong, S. Chen, Y. Li, R. Du, A. Steinfeld, and S. Labi, "Facilitating connected autonomous vehicle operations using space-weighted information fusion and deep reinforcement learning based control," *arXiv preprint arXiv:2009.14665*, 2020.

[115] K. Gao, D. Yan, F. Yang, J. Xie, L. Liu, R. Du, and N. Xiong, "Conditional artificial potential field-based autonomous vehicle safety control with interference of lane changing in mixed traffic scenario," *Sensors*, vol. 19, no. 19, p. 4199, 2019.

[116] W. Zhou, D. Chen, J. Yan, Z. Li, H. Yin, and W. Ge, "Multi-agent reinforcement learning for cooperative lane changing of connected and autonomous vehicles in mixed traffic," *Autonomous Intelligent Systems*, vol. 2, no. 1, pp. 1–11, 2022.

[117] Y. Liu, A. Zhou, Y. Wang, and S. Peeta, "Proactive longitudinal control of connected and autonomous vehicles with lane-change assistance for human-driven vehicles," in *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, IEEE, 2021, pp. 776–781.

[118] R. Du, S. Chen, Y. Li, P. Y. J. Ha, J. Dong, P. C. Anastasopoulos, and S. Labi, "A cooperative crash avoidance framework for autonomous vehicle under collision-imminent situations in mixed traffic stream," in *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, IEEE, 2021, pp. 1997–2002.

[119] X. Liu, R. Jiao, B. Zheng, D. Liang, and Q. Zhu, "Connectivity enhanced safe neural network planner for lane changing in mixed traffic," *arXiv preprint arXiv:2302.02513*, 2023.

[120] P.-C. Chen, X. Liu, C.-W. Lin, C. Huang, and Q. Zhu, "Mixed-traffic intersection management utilizing connected and autonomous vehicles as traffic regulators," in *Proceedings of the 28th Asia and South Pacific Design Automation Conference*, 2023, pp. 52–57.

[121] P. Y. J. Ha, S. Chen, J. Dong, R. Du, Y. Li, and S. Labi, "Leveraging the capabilities of connected and autonomous vehicles and multi-agent reinforcement learning to mitigate highway bottleneck congestion," *arXiv preprint arXiv:2010.05436*, 2020.

[122] O. Nassef, L. Sequeira, E. Salam, and T. Mahmoodi, "Building a lane merge coordination for connected vehicles using deep reinforcement learning," *IEEE Internet of Things Journal*, 2020.

[123] S. Han and F. Miao, "Behavior planning for connected autonomous vehicles using feedback deep reinforcement learning," *arXiv preprint arXiv:2003.04371*, 2020.

[124] K. B. Naveed, Z. Qiao, and J. M. Dolan, "Trajectory planning for autonomous vehicles using hierarchical reinforcement learning," *arXiv preprint arXiv:2011.04752*, 2020.

[125] M. S. Nosrati, E. A. Abolfathi, M. Elmahgiubi, P. Yadmellat, J. Luo, Y. Zhang, H. Yao, H. Zhang, and A. Jamil, "Towards practical hierarchical reinforcement learning for multi-lane autonomous driving," 2018.

[126] J. Li, L. Sun, J. Chen, M. Tomizuka, and W. Zhan, "A safe hierarchical planning framework for complex driving scenarios based on reinforcement learning," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2021, pp. 2660–2666.

[127] J. Wang, Y. Wang, D. Zhang, Y. Yang, and R. Xiong, "Learning hierarchical behavior and motion planning for autonomous driving," *arXiv preprint arXiv:2005.03863*, 2020.

[128] H. Ma, J. Chen, S. E. Li, Z. Lin, Y. Guan, Y. Ren, and S. Zheng, "Model-based constrained reinforcement learning using generalized control barrier function," *arXiv preprint arXiv:2103.01556*, 2021.

[129] R. Jiao, H. Liang, T. Sato, J. Shen, Q. A. Chen, and Q. Zhu, "End-to-end uncertainty-based mitigation of adversarial attacks to automated lane centering," in *2021 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2021, pp. 266–273.

[130] H. Liang, R. Jiao, T. Sato, J. Shen, Q. A. Chen, and Q. Zhu, "Wip: End-to-end analysis of adversarial attacks to automated lane centering systems," in *Workshop on Automotive and Autonomous Vehicle Security (AutoSec'21)*, 2021.

[131] H.-D. Tran, F. Cai, M. L. Diego, P. Musau, T. T. Johnson, and X. Koutsoukos, "Safety verification of cyber-physical systems with reinforcement learning control," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 18, no. 5s, pp. 1–22, 2019.

[132] C. Huang, J. Fan, W. Li, X. Chen, and Q. Zhu, "Reachnn: Reachability analysis of neural-network controlled systems," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 18, no. 5s, pp. 1–22, 2019.

[133] S. Dutta, X. Chen, and S. Sankaranarayanan, "Reachability analysis for neural feedback systems using regressive polynomial rule inference," in *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*, 2019, pp. 157–168.

[134] R. Ivanov, J. Weimer, R. Alur, G. J. Pappas, and I. Lee, "Verisig: Verifying safety properties of hybrid systems with neural network controllers," in *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*, 2019, pp. 169–178.

[135] Z. Wang, Y. Wang, F. Fu, R. Jiao, C. Huang, W. Li, and Q. Zhu, "A tool for neural network global robustness certification and training," *arXiv preprint arXiv:2208.07289*, 2022.

[136] Y. Wang, C. Huang, and Q. Zhu, "Energy-efficient control adaptation with safety guarantees for learning-enabled cyber-physical systems," in *Proceedings of the 39th International Conference on Computer-Aided Design*, 2020, pp. 1–9.

[137] Y. Wang, C. Huang, Z. Wang, S. Xu, Z. Wang, and Q. Zhu, "Cocktail: Learn a better neural network controller from multiple experts via adaptive mixing and robust distillation," in *2021 58th ACM/IEEE Design Automation Conference (DAC)*, IEEE, 2021, pp. 397–402.

[138] Y. Wang, C. Huang, Z. Wang, Z. Wang, and Q. Zhu, "Design-while-verify: Correct-by-construction control learning with verification in the loop," in *Proceedings of the 59th ACM/IEEE Design Automation Conference*, 2022, pp. 925–930.

[139] Y. Wang, C. Huang, Z. Wang, Z. Yang, and Q. Zhu, "Joint differentiable optimization and verification for certified reinforcement learning," *arXiv preprint arXiv:2201.12243*, 2022.

[140] Y. Wang, S. S. Zhan, R. Jiao, Z. Wang, W. Jin, Z. Yang, Z. Wang, C. Huang, and Q. Zhu, "Enforcing hard constraints with soft barriers: Safe reinforcement learning in unknown stochastic environments," *arXiv preprint arXiv:2209.15090*, 2022.

[141] C. Huang, S. Xu, Z. Wang, S. Lan, W. Li, and Q. Zhu, "Opportunistic intermittent control with safety guarantees for autonomous systems," in *2020 57th ACM/IEEE Design Automation Conference (DAC)*, IEEE, 2020, pp. 1–6.

[142] J. Fan, C. Huang, W. Li, X. Chen, and Q. Zhu, "Towards verification-aware knowledge distillation for neural-network controlled systems," in *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, IEEE, 2019, pp. 1–8.

[143] H.-D. Tran, X. Yang, D. M. Lopez, P. Musau, L. V. Nguyen, W. Xiang, S. Bak, and T. T. Johnson, "NNV: The neural network verification tool for deep neural networks and learning-enabled cyber-physical systems," in *32nd International Conference on Computer-Aided Verification (CAV)*, Jul. 2020.

[144] Q. Zhu, W. Li, H. Kim, Y. Xiang, K. Wardega, Z. Wang, Y. Wang, H. Liang, C. Huang, J. Fan, *et al.*, "Know the unknowns: Addressing disturbances and uncertainties in autonomous systems," in *Proceedings of the 39th International Conference on Computer-Aided Design*, 2020, pp. 1–9.

[145] M. Naumann, H. Königshof, and C. Stiller, "Provably safe and smooth lane changes in mixed traffic," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, IEEE, 2019, pp. 1832–1837.

[146] C. Pek, P. Zahn, and M. Althoff, "Verifying the safety of lane change maneuvers of self-driving vehicles based on formalized traffic rules," in *2017 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2017, pp. 1477–1483.

[147] R. Chandru, Y. Selvaraj, M. Brännström, R. Kianfar, and N. Murgovski, "Safe autonomous lane changes in dense traffic," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2017, pp. 1–6.

[148] J. F. Fisac, A. Bajcsy, S. L. Herbert, D. Fridovich-Keil, S. Wang, C. J. Tomlin, and A. D. Dragan, "Probabilistically safe robot planning with confidence-based human predictions," *arXiv preprint arXiv:1806.00109*, 2018.

[149] K. K.-C. Chang, X. Liu, C.-W. Lin, C. Huang, and Q. Zhu, "A safety-guaranteed framework for neural-network-based planners in connected vehicles under communication disturbance," in *2023 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, IEEE, 2023.

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

[151] E. Zhang, S. Pizzi, and N. Masoud, "A learning-based method for predicting heterogeneous traffic agent trajectories: Implications for transfer learning," in *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, IEEE, 2021, pp. 1853–1858.

[152] E. Zhang, N. Masoud, M. Bandegi, J. Lull, and R. K. Malhan, "Step attention: Sequential pedestrian trajectory prediction," *IEEE Sensors Journal*, vol. 22, no. 8, pp. 8071–8083, 2022.

[153] R. Jiao, X. Liu, B. Zheng, D. Liang, and Q. Zhu, "Tae: A semi-supervised controllable behavior-aware trajectory generator and predictor," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2022, pp. 12 534–12 541.

[154] Y. Liu, J. Zhang, L. Fang, Q. Jiang, and B. Zhou, "Multimodal motion prediction with stacked transformers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 7577–7586.

[155] M. Ye, T. Cao, and Q. Chen, "Tpcn: Temporal point cloud networks for motion forecasting," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 11 318–11 327.

[156] Q. Xue, Y. Xing, and J. Lu, "An integrated lane change prediction model incorporating traffic context based on trajectory data," *Transportation Research Part C: Emerging Technologies*, vol. 141, p. 103 738, 2022.

[157] Y. Hu, X. Jia, M. Tomizuka, and W. Zhan, "Causal-based time series domain generalization for vehicle intention prediction," in *2022 International Conference on Robotics and Automation (ICRA)*, IEEE, 2022, pp. 7806–7813.

[158] N. Rhinehart, R. McAllister, K. Kitani, and S. Levine, "Precog: Prediction conditioned on goals in visual multi-agent settings," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 2821–2830.

[159] L. Sun, C. Tang, Y. Niu, E. Sachdeva, C. Cho, T. Misu, M. Tomizuka, and W. Zhan, "Domain knowledge driven pseudo labels for interpretable goal-conditioned interactive trajectory prediction," *arXiv preprint arXiv:2203.15112*, 2022.

[160] D. González, J. Pérez, V. Milanés, and F. Nashashibi, "A review of motion planning techniques for automated vehicles," *IEEE Transactions on intelligent transportation systems*, vol. 17, no. 4, pp. 1135–1145, 2015.

[161] Z. Zhu and H. Zhao, "A survey of deep rl and il for autonomous driving policy learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 9, pp. 14 043–14 065, 2021.

[162] M. Zhu, X. Wang, and Y. Wang, "Human-like autonomous car-following model with deep reinforcement learning," *Transportation research part C: emerging technologies*, vol. 97, pp. 348–368, 2018.

[163] D.-F. Xie, Z.-Z. Fang, B. Jia, and Z. He, "A data-driven lane-changing model based on deep learning," *Transportation research part C: emerging technologies*, vol. 106, pp. 41–60, 2019.

[164] R. Tian, L. Sun, A. Bajcsy, M. Tomizuka, and A. D. Dragan, "Safety assurances for human-robot interaction via confidence-aware game-theoretic human models," in *2022 International Conference on Robotics and Automation (ICRA)*, IEEE, 2022, pp. 11 229–11 235.

[165] Y. Huang and M. A. Jafari, "Risk-aware vehicle motion planning using bayesian lstm-based model predictive control," *arXiv preprint arXiv:2301.06201*, 2023.

[166] Y. Chai, B. Sapp, M. Bansal, and D. Anguelov, "Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction," *arXiv preprint arXiv:1910.05449*, 2019.

[167] T. Phan-Minh, E. C. Grigore, F. A. Boulton, O. Beijbom, and E. M. Wolff, "Covernet: Multimodal behavior prediction using trajectory sets," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 14 074–14 083.

[168] R. Jiao, J. Bai, X. Liu, T. Sato, X. Yuan, Q. A. Chen, and Q. Zhu, "Learning representation for anomaly detection of vehicle trajectories," *arXiv preprint arXiv:2303.05000*, 2023.

[169] W. Wang, D. T. Hoang, Z. Xiong, D. Niyato, P. Wang, P. Hu, and Y. Wen, "A survey on consensus mechanisms and mining management in blockchain networks," *arXiv preprint arXiv:1805.02707*, pp. 1–33, 2018.

[170] J. Chen and S. Micali, "Algorand: A secure and efficient distributed ledger," *Theoretical Computer Science*, vol. 777, pp. 155–183, 2019.

[171] Y. Gilad, R. Hemo, S. Micali, G. Vlachos, and N. Zeldovich, "Algorand: Scaling byzantine agreements for cryptocurrencies," in *Proceedings of the 26th Symposium on Operating Systems Principles*, 2017, pp. 51–68.

[172] M. Pustišek, A. Kos, and U. Sedlar, "Blockchain based autonomous selection of electric vehicle charging station," in *2016 international conference on identification, information and knowledge in the Internet of Things (IIKI)*, IEEE, 2016, pp. 217–222.

[173] F. Gao, L. Zhu, M. Shen, K. Sharif, Z. Wan, and K. Ren, "A blockchain-based privacy-preserving payment mechanism for vehicle-to-grid networks," *IEEE network*, vol. 32, no. 6, pp. 184–192, 2018.

[174] Y. Yuan and F.-Y. Wang, "Towards blockchain-based intelligent transportation systems," in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2016, pp. 2663–2668.

[175] W. Dong, Y. Li, R. Hou, X. Lv, H. Li, and B. Sun, "A blockchain-based hierarchical reputation management scheme in vehicular network," in *2019 IEEE Global Communications Conference (GLOBECOM)*, IEEE, 2019, pp. 1–6.

[176] A. Dorri, S. S. Kanhere, and R. Jurdak, "Towards an optimized blockchain for iot," in *2017 IEEE/ACM Second International Conference on Internet-of-Things Design and Implementation (IoTDI)*, IEEE, 2017, pp. 173–178.

[177] T. Jiang, H. Fang, and H. Wang, "Blockchain-based internet of vehicles: Distributed network architecture and performance analysis," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4640–4649, 2018.

[178] Q. Zhu, C. Huang, R. Jiao, S. Lan, H. Liang, X. Liu, Y. Wang, Z. Wang, and S. Xu, "Safety-assured design and adaptation of learning-enabled autonomous systems," in *Proceedings of the 26th Asia and South Pacific Design Automation Conference*, 2021, pp. 753–760.

[179] X. Chen, J. Fan, C. Huang, R. Jiao, W. Li, X. Liu, Y. Wang, Z. Wang, W. Zhou, and Q. Zhu, "Machine learning and optimization techniques for automotive cyber-physical systems," in, Springer Nature.

[180] Y. Chen, Y. Shi, and B. Zhang, "Input convex neural networks for optimal voltage regulation," *arXiv preprint arXiv:2002.08684*, 2020.

[181] A. H. Qureshi, Y. Miao, A. Simeonov, and M. C. Yip, "Motion planning networks: Bridging the gap between learning-based and classical motion planners," *IEEE Transactions on Robotics*, 2020.

[182] J. H. Seidman, M. Fazlyab, V. M. Preciado, and G. J. Pappas, "Robust deep learning as optimal control: Insights and convergence guarantees," in *Learning for Dynamics and Control*, PMLR, 2020, pp. 884–893.

[183] M. Zhou, Y. Yu, and X. Qu, "Development of an efficient driving strategy for connected and automated vehicles at signalized intersections: A reinforcement learning approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 1, pp. 433–443, 2019.

[184] W. Zeng, W. Luo, S. Suo, A. Sadat, B. Yang, S. Casas, and R. Urtasun, "End-to-end interpretable neural motion planner," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8660–8669.

[185] T. A. Henzinger, P. W. Kopke, A. Puri, and P. Varaiya, "What's decidable about hybrid automata?" *Journal of computer and system sciences*, vol. 57, no. 1, pp. 94–124, 1998.

[186] C. Huang, J. Fan, X. Chen, W. Li, and Q. Zhu, "Polar: A polynomial arithmetic framework for verifying neural-network controlled systems," *arXiv preprint arXiv:2106.13867*, 2021.

[187] I. G. Jin and G. Orosz, "Optimal control of connected vehicle systems with communication delay and driver reaction time," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 8, pp. 2056–2070, 2016.

[188] X. Liu, R. Jiao, Y. Wang, Y. Han, B. Zheng, and Q. Zhu, "Safety-assured speculative planning with adaptive prediction," 2023.

[189] Z. Hong, Y. Chen, and H. S. Mahmassani, "Recognizing network trip patterns using a spatio-temporal vehicle trajectory clustering algorithm," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 8, pp. 2548–2557, 2017.

[190] A. Alhariqi, Z. Gu, and M. Saberi, "Calibration of the intelligent driver model (idm) with adaptive parameters for mixed autonomy traffic using experimental trajectory data," *Transportmetrica B: transport dynamics*, vol. 10, no. 1, pp. 421–440, 2022.

[191] A. Xiao, H. Luan, Z. Zhao, Y. Hong, J. Zhao, W. Chen, J. Wang, and M. Q.-H. Meng, "Robotic autonomous trolley collection with progressive perception and nonlinear model predictive control," in *2022 International Conference on Robotics and Automation (ICRA)*, IEEE, 2022, pp. 4480–4486.

[192] Y. Liang, Y. Li, A. Khajepour, Y. Huang, Y. Qin, and L. Zheng, "A novel combined decision and control scheme for autonomous vehicle in structured road based on adaptive model predictive control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 9, pp. 16 083–16 097, 2022.

[193] X. Liu, B. Luo, A. Abdo, N. Abu-Ghazaleh, and Q. Zhu, "Securing connected vehicle applications with an efficient dual cyber-physical blockchain framework," in *2021 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2021, pp. 393–400.

[194] Y. L. Sun, Z. Han, W. Yu, and K. R. Liu, "A trust evaluation framework in distributed networks: Vulnerability analysis and defense against attacks," in *Proceedings IEEE INFOCOM 2006. 25TH IEEE International Conference on Computer Communications*, IEEE, 2006, pp. 1–13.

[195] *Caltrans performance measurement system (pems)*, data retrieved from `http://pems.dot.ca.gov/`.

[196] J. Yang, Y. Liu, P. Qin, and A. A. Liu, "A review of beijing's vehicle registration lottery: Short-term effects on vehicle growth and fuel consumption," *Energy Policy*, vol. 75, pp. 157–166, 2014.