Safe and Robust Cooperative Algorithm for Connected Autonomous Vehicles

by

Mohammad Khayatian

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved June 2021 by the
Graduate Supervisory Committee:

Aviral Shrivastava, Chair
Georgios Fainekos
Heni Ben Amor
Yezhou Yang
Yingyan Lou
Bob Iannucci

ARIZONA STATE UNIVERSITY

August 2021

ABSTRACT

Autonomous Vehicles (AVs) have the potential to significantly evolve transportation. AVs are expected to make transportation safer by avoiding accidents that happen due to human errors. When AVs become connected, they can exchange information with the infrastructure or other Connected Autonomous Vehicles (CAVs) to efficiently plan their future motion and therefore, increase the road throughput and reduce energy consumption. Cooperative algorithms for CAVs will not be deployed in real life unless they are proved to be safe, robust, and resilient to different failure models. Since intersections are crucial areas where most accidents happen, this dissertation first focuses on making existing intersection management algorithms safe and resilient against network and computation time, bounded model mismatches and external disturbances, and the existence of a rogue vehicle. Then, a generic algorithm for conflict resolution and cooperation of CAVs is proposed that ensures the safety of vehicles even when other vehicles suddenly change their plan. The proposed approach can also detect deadlock situations among CAVs and resolve them through a negotiation process. A testbed consisting of 1/10th scale model CAVs is built to evaluate the proposed algorithms. In addition, a simulator is developed to perform tests at a large scale. Results from the conducted experiments indicate the robustness and resilience of proposed approaches.

To my parents, Alireza & Zahra,

to my wife, Shakiba,

and to my brother, Fazel.

ACKNOWLEDGEMENT

First and foremost, I am thankful to my advisor Prof. Aviral Shrivastava who not only supported me as a mentor in life, but also provided me a unique environment for conducting high-quality research. My achievements during my Ph.D. life at Arizona State University, including this dissertation, would not have been possible without his guidance.

I would like to thank the rest of my committee members, Prof. Georgios Fainekos, Prof. Heni Ben Amor, Prof. Yezhou Yang, Prof. Yingyan Lou and Prof. Bob Iannucci for their supervision and openly sharing and discussing novel research ideas.

I am very grateful of my colleagues at the MPS-Lab (aka CML) who created a friendly environment for me to thrive.

Finally, I would like to thank my parents for their encouragement and support during the past three decades, my brother for his mentorship and most importantly my wife Shakiba for her love and companionship.

This work was partially supported by funding from NIST Award 70NANB19H144, and by National Science Foundation grants CNS 1525855 and CPS 1645578.

TABLE OF CONTENTS

CHAPTER Page

CHAPTER Page

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

Figure Page

Figure Page

Figure Page

Figure Page

Figure Page

Chapter 1

INTRODUCTION

Intelligent Transportation Systems (ITS) have the potential to revolutionize transportation by providing safer and more efficient driving experiences. In the past decade, many automotive industries were focused on improving the Advanced driver-assistance systems (ADAS) and have tried to pave the road to deploy fully Autonomous Vehicles (AVs) that can drive without human intervention. Today, more than 65 automotive companies are permitted to test their AVs on the streets of California, US CNBC (2020). When AVs become connected, they can share their information with other AVs and/or the infrastructure in order to avoid potential accidents and increase the throughput of the roads. Traffic management of Connected Autonomous Vehicles (CAVs) can take place at different places and for different purposes including but not limited to platooning in highways, cooperative merging at ramps, automated roundabout management, cooperative lane changing at highways and automated intersection management Khayatian *et al.* (2020b).

According to the Federal Highway Administration (FHA), 40 percent of all crashes happen at intersections, which is the second-largest category of accidents Administration (2019). As a result, I will first focus on making existing intersection management algorithms robust. In the past few years, the intersection management of CAVs has been the focus of many researchers and so far, a variety of intersection management approaches Dresner and Stone (2008b); Lee and Park (2012); Azimi *et al.* (2014) were proposed. However, existing works are mostly focused on performance-related aspects of intersection management (e.g. which scheduling policy result in a better throughput) and safety aspects have received less attention.

My thesis statement is that Cooperative algorithms for CAVs will only be deployed in the real world if they are proved to be robust to several kinds of failures.

To support this thesis, in this dissertation, I propose several safe and robust cooperative algorithms for CAVs that have a better chance of being deployed in the real world starting from centralized approaches for intersection management and then a decentralized approach for general situations. In specific, the contributions of this dissertation are developing algorithms that are:

- **Robust against network delay and computation time:** Many of the existing techniques do not consider the communication delay between CAVs and the Intersection Manager (IM) and their computation times. As a result, a CAV may be ahead of its expected position when receiving the assigned velocity from the IM and may cause an accident. I have developed intersection management algorithms that are resilient against network delay and computation time Andert *et al.* (2017); Khayatian *et al.* (2019). Proposed approaches set a bound on the worst-case round trip delay (from the moment a CAV sends its information to the moment it receives the response) and the IM instructs the CAVs to actuate at a fixed time which is set to be greater than the worst-case round trip delay.

- **Robust against model mismatches and external disturbances:** In most of the previous works a constant velocity is assigned to CAVs to track and it is assumed that the considered model for the behavior of vehicles is perfect. However, model mismatches and external disturbances (e.g. wind, bump) can temporarily degrade the tracking of the vehicle and cause errors in the even-

tual position of vehicles (for velocity-assignment methods). I have developed intersection management algorithms that are resilient against small model mismatches and external disturbances Khayatian *et al.* (2018). In this approach, the IM assigns the desired Velocity of Arrival (VoA) and Time of Arrival (ToA) to CAVs instead of a constant velocity. Then, each CAV creates an optimal reference trajectory and track it locally. As a result, the effect of external disturbances and model mismatches can be reduced or completely compensated for.

- **Robust against Rogue Vehicles:** CAVs may become rogue unintentionally or deliberately and lie about their position, velocity, etc. when sharing their information or disobey IM's direction and accelerate or decelerate to enter the intersection earlier or later than scheduled. I have developed intersection management algorithms that are resilient against rogue vehicles (vehicles that may share wrong information and/or does not follow the expected trajectory) Khayatian *et al.* (2020a). In the proposed technique, a surveillance system is used to detect rouge vehicles and notify other CAVs to either stop (if they can stop without entering the intersection area) or continue (if a safe stop is not possible). The temporal buffer between cross time of CAVs is select to be large enough so that even in cases that a CAV cannot stop safely without entering the intersection, the rouge vehicle is far enough and cannot physically hit it.

- **Robust against blame in an accident:** Most motion planning algorithms assume that all CAVs follow the assigned trajectory while a CAV may change its plan (e.g. break down or stop to yield to a jay-walker). The state-of-the-art work (Responsibility-sensitive Safety (RSS)) does not scale to properly handle merge, intersection and unstructured road scenarios. I have developed a generic

conflict resolution algorithm for CAVs that ensures safety of CAVs even when they suddenly change their plan and can detect Khayatian *et al.* (2021). The proposed approach first determines the set of possible conflict zones for every pair of CAVs and then computes who has the advantage to enter the conflict zone first. The CAV with a disadvantage is instructed to yield to the other CAV and account for its worst-case behavior (even if it applied full brake).

- **Robust against deadlocks:** When CAVs interact, they can get into a deadlock situation, i.e., the CAVs slow down for each other, and no-one can make progress. Existing approaches for deadlock resolution of CAVs assume that all CAVs have access to information of all other CAVs and conflicts among them, which in turn, result in a high computation overhead. I have developed a distributed and efficient algorithm that can detect and resolve deadlocks among CAVs Khayatian *et al.* (2021). The proposed approach comes with a mechanism where CAVs broadcast their yield-to graph (to whom they are yielding to) and after receiving other CAVs' yield-to graph and combining them, they can detect if there is a cycle in the graph and remove it in the same way.

We have evaluated our algorithms on a testbed with 1/10th scale model CAVs and on different simulators for large scale testing. Results from our experiments indicate the robustness and resilience of proposed approaches to mentioned fault models.

The organization of this dissertation is as follows: In the second chapter, I provide a complete literature review on existing intersection management algorithms and compare existing works from different perspectives. In chapters three, four, and five, I present our proposed approaches for intersection management of CAVs and in chapter six, I present our new RSS-based conflict resolution and deadlock resolution algorithm for CAVs.

Chapter 2

A SURVEY ON INTERSECTION MANAGEMENT OF CAVS

In this chapter, we provide a thorough survey on existing intersection management algorithms for CAV that are reported in the literature to date and evaluate them from following perspectives: 1) V2V/V2I interface for intersection management, 2) scheduling policy for CAVs, 3) wireless technology, 4) model for vehicle dynamics, 5) conflict detection, 6) extension to multi-intersections, 7) support for human-driven vehicles, 8) safety and robustness, 9) emergency situations and recovery, 10) security concerns, and 11) evaluation method

2.1 V2I/V2V Interface for Intersection Management

Deployment of an intersection management algorithm in real life requires certain specifications to be defined by designers. For instance, the algorithm must specify what information will be exchanged, who is responsible for the scheduling of CAVs -is there a separate infrastructure near the intersection or will one of the CAVs take the responsibility?

Existing decentralized/centralized approaches are different in terms of communication protocol and information that is shared. Some of the existing works specifically mention what information needs to be exchanged while some other works, do not and assume that a CAVs or the Intersection Manager (IM) have to access to all information of CAVs.

Based on the fact that who manages the intersection, we categorize existing works into two groups: 1) Distributed, where CAVs do the scheduling themselves and 2) Centralized, where there is a station near the intersection that schedules approaching

CAVs. Figure 2.1 shows an overview of a centralized and distributed intersection management interface.

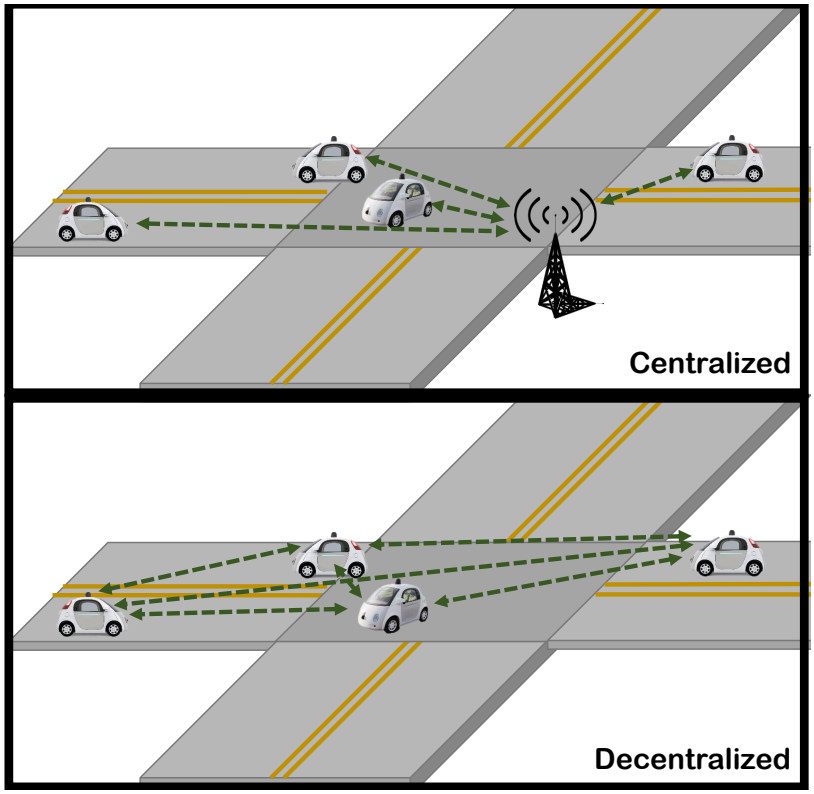


Figure 2.1: Main Interfaces to Manage the Intersection of CAVs. In Centralized Approaches, CAVs Communicate with the Infrastructure While in Distributed Approaches, CAVs Communicate with Each Other.

2.1.1 Distributed Approaches

As an advantage of distributed approaches, they do not need support from infrastructure, which means they can scale easily and be used in uncrowded intersections controlled by stop signs and those in rural areas.

Li *et al.* Li and Wang (2006) developed a distributed intersection management algorithm where CAVs randomly communicate with each other to form small groups when they are within a certain radius of the intersection. All CAVs share their ID,

width and length, incoming/outgoing lane, velocity, and position. Then, CAVs from different groups communicate with each other to collect the information of CAVs in other groups. As soon as a CAV receives the information of all vehicles, it becomes the leader or intersection manager and schedules the cross-time of CAVs. The leader also lets other CAVs know about its leadership so that they stop collecting data.

STIP (Spatio-Temporal Intersection Protocol) Azimi *et al.* (2014) is another cooperative intersection management algorithm where there are three message types that a CAV sends to the others: ENTER, CROSS, and EXIT. In this method, CAVs share their ID, current road segment, current lane, future road segment, arrival-time, exit-time, list of trajectories, list of arrival times, and message sequence. When two CAVs intend to cross the same zone and their cross-time overlaps, the CAV with higher priority continues and enters the intersection while the CAV with lower priority slows down and stops before entering the conflict zones. The priority for CAVs is determined based on the FCFS policy where a CAV with earlier arrival time has a higher priority.

In Katriniok *et al.* (2018, 2019), Katriniok *et al.* proposed a model predictive control (MPC) technique to coordinate vehicles through the intersection. Upon approaching the intersection, each CAV receives the trajectories of all other CAVs and then formulates and solves an optimal control problem to find a sequence of actions. Next, the CAV broadcasts its information including distances to collision point with other CAVs. This process is repeated again after a short time-step to handle newly approaching CAVs.

Aoki *et al.* Aoki and Rajkumar (2018) proposed a general solution for scenarios that a pair of CAVs have conflicts on their future paths including an intersection. In this work, a Request-response negotiation-based protocol is proposed to detect dynamic intersections of CAVs. CAVs notify each other about the existence of conflicts

and yielding to/interrupting other CAVs. In this approach, four message types are defined: 1) Dynamic Intersection or DI request, to notify other CAVs, 2) DI approve, to acknowledge the requested maneuver, 3) DI interrupt, to ask other vehicles to stop, and 4) DI yield, to respond to DI interrupt.

In Liu *et al.* (2018), the intersection area is divided into multiple conflict zones. Upon approaching the intersection, each CAV periodically broadcasts its arrival time and departure time with respect to all the conflict zones that it intends to occupy. If a CAV detects a conflict, it determines if it has the advantage to enter the conflict zone first. A CAV will have the advantage if it proposes to 1) leave some conflict zone later than the other CAV, 2) leave all conflict zones earlier than the other CAV, and 3) enter some conflict zone earlier than the other CAV. The vehicle that has the advantage continues with its plan and the other CAV changes its plan such that its enter time to all conflict zones is later than the exit time of the CAV with the advantage. This technique assumes that all CAVs are synchronized where the computation happens at the same time within the broadcasting period.

Belkhouche *et al.* propose a distributed collision detection system Belkhouche (2018) that is aware of the unsafe situations that may happen with respect to another CAVs that is approaching the intersection. In this approach, the set of all velocities that may cause an accident in the future are determined for a pair of CAVs. If a conflict exists, one of the CAVs must accelerate and the other one will decelerate. The optimal crossing is then determined by finding the desired velocity for both CAVs such that CAVs change their velocity minimally while avoiding the set of unsafe velocities.

In Bian *et al.* approachBian *et al.* (2019), the area before entering the intersection is divided into three zones. A CAV will first enter the observation zone, where it observes the current state of other CAVs and their order, then it enters the optimization zone, where it optimizes its trajectory, finally, it enters the control zone, where

the CAV tracks the desired trajectory. This paper assumes that the communication range is limited and therefore a CAV may not be able to receive the information of all CAVs. As a result, it estimates the state (position and velocity) of out-of-range CAVs using the information broadcast by their neighbors.

In Filocamo *et al.* (2020), CAVs send/receive position, speed, and direction upon entering the communication area and then calculate a priority based on the arrival time. A CAV with lower priority yields to CAVs with higher priorities by slowing down such that it arrives at the intersection when the intersection is cleared. This process is repeated until a CAV leaves the intersection.

Among existing works that propose a distributed intersection management interface, in Li and Wang (2006), a leader is selected dynamically to schedule CAVs while in the restAzimi *et al.* (2014); Liu *et al.* (2018); Katriniok *et al.* (2018); Aoki and Rajkumar (2018); Belkhouche (2018), each CAV determines its plan based on the shared information of other CAVs and its own state. Selecting a leader that performs intersection management is very similar to a centralized approach. Later, we will study the pros and cons of centralized and distributed intersection management. In general, each distributed approach follows a unique protocol for communication where the number of exchanged messages and their size differs.

2.1.2 Centralized Approaches

Centralized algorithms mostly follow a server-client scheme where vehicles send a request to the IM and the IM replies with a response. We categorize existing centralized approaches into two groups: query-based intersection management or QB-IM, and assignment-based intersection management AB-IM approaches. In QB-IM, vehicles query a safe passage from the IM by proposing a cross-time/velocity and the IM either accepts or rejects the vehicle's proposal. In AB-IM, vehicles share their

status with the IM and the IM assigns a cross-time to each vehicle, and vehicles follow that.

Query-based Intersection Management

Autonomous Intersection Management (AIM) Dresner and Stone (2008b) was one of the initial attempts to develop a centralized algorithm for intersection management of CAVs. In AIM, the intersection is modeled as a grid of squares. Each of these squares is represented in discrete time-steps. Vehicles approaching the intersection query safe entry to the intersection by sending their estimated time of arrival and velocity of arrival. The IM generates the future trajectory of the vehicle in terms of time-space (which square will be used and when) and checks if it conflicts with other time-space reservations (for other vehicles). If there is a conflict, the IM rejects the request and the vehicle slows down and requests again after a timeout. If no reservation is assigned to a vehicle, it will stop behind the intersection edge and request again. If there is no conflict, the vehicle continues and enters the intersection. AIM is a query-based intersection management (QB-IM) approach where vehicles query safe passage from the IM and the IM replies a YES/NO. As a result, this approach may face higher network overheads and achieve lower throughputs. This is because vehicles may come to a complete stop and have to send multiple requests until getting a reservation. Liu *et al.* (2019a) proposes a similar QB-IM methodology where vehicles send a request to the IM reporting their future conflict zone occupation time (CZOT). The IM store CZOTs of all vehicles and share it with all vehicles. Then, each vehicle finds a valid solution (a new CZOT that does not have any conflict with other CAVs) and reports it to the IM. If two CAVs request the CZOT at the same time, the IM responds to them in the order it receives the request. IM does not respond to other CAVs until it receives the proposed CZOT and updates its local CZOT Choi *et al.* (2019) is also

a similar query-based algorithm where each CAV sends a reservation to the IM and IM either accepts or rejects the request. In this approach, there are two zones, 1) queuing zone and 2) acceleration zone. The vehicle sends their request only when they are in the queuing zone.

Jin *et al.* Jin *et al.* (2013) follow another approach where platoons of CAVs are formed using V2V communication and each platoon has a leader. The leader communicates with the IM on behalf of its platoon by sending the platoon's earliest arrival time and passage time. The IM evaluates the reservation time slot and responds to the proposal of the leader by either accepting or rejecting the request and suggesting a reservation for the platoon. Bashiri *et al.* (2018) and Bashiri and Fleming (2017) are similar approaches where platoons of CAVs are formed and only leaders communicate with the IM by sending one the following messages: 1) Request, 2) Change-Request, 3) Acknowledge or 4) Done. Accordingly, IM follows a query-based approach and responds to a request by sending one the following messages: 1) Acknowledge, 2) Confirm, or 3) Reject. For the request, a leader vehicle sends its VIN (vehicle identification number) as ID, current position, velocity, acceleration, estimation for the time of arrival, and the size of the platoon. Jin *et al.* (2012a) is another QB-IM approach where vehicles send their estimated earliest arrival time to the IM to reserve a time slot. The IM uses a dynamic reservation system that accepts or rejects a request based on the priority of the request. Jin *et al.* (2012b) is a variation of the same approach using a different scheduling policy and Bento *et al.* (2012) proposes to use a similar QB-IM approach.

Assignment-based Intersection Management

In 2016, Yang *et al.* Yang *et al.* (2016) proposed an AB-IM algorithm where the IM collects information of all CAVs that are within the range of the intersection and

assigns a trajectory to each vehicle. The scheduling process is repeated when a new vehicle enters the control zone, an existing vehicle departs the intersection or it comes to a stop.

Crossroads Andert *et al.* (2017) and Crossroads+ Khayatian *et al.* (2019) are similar AB-IM approaches where vehicles first synchronize their internal clock with the IM and then, let the IM know of their presence by sending their position, velocity, and exit lane along with a timestamp that corresponds to the captured status. IM checks the status of existing vehicles and assigns a constant velocity and "time to actuate" to each vehicle. Once a vehicle receives the response, waits until the time to actuate and then accelerate/decelerate to maintain the assigned velocity. Azimi *et al.* Azimi *et al.* (2015) propose a similar approach where the IM assigns a TOA and VOA to a CAV and also checks for deadlock and resolve them. In Sayin *et al.* (2018), another AB-IM approach is presented where approaching vehicles send a request to the IM containing their utility function (u) and safety function (s) and the IM schedules vehicle such that the sum of all utility functions is maximized. Authors have also provided a mechanism for truthful utility reporting. In Lu *et al.* approachLu and Kim (2016), the IM creates a queue for approaching CAVs which is sorted based on the request time and then, assigns an occupancy in space-time to CAVs. Qian *et al.* Qian *et al.* (2019) present an interface between the IM and CAVs where each CAV sends a request by sharing its information and the IM computes a scheduling solution for it. The IM also waits for feedback from the CAV to make sure the scheduled plan is received. In Khayatian *et al.* (2018), each CAV sends its position, velocity, outgoing lane, and timestamp to the IM and the IM assigns a time of arrival and velocity of arrival to the CAV.

We have categorized existing works in terms of their interface and management algorithm in Table 2.1.

Intersection management interface		
Distributed	Centralized	
	Query-based	Assignment-based
Zohdy *et al.* (2012); Li and Wang (2006); Elhenawy *et al.* (2015); Azimi *et al.* (2014); Lee *et al.* (2013); Aoki and Rajkumar (2018); Liu *et al.* (2018); Qiao *et al.* (2018); Katriniok *et al.* (2018, 2019); Belkhouche (2018); Bian *et al.* (2019); Filocamo *et al.* (2020)	Dresner and Stone (2008b); Jin *et al.* (2012a,b); Stone *et al.* (2015); Vasirani and Ossowski (2012); Lin *et al.* (2017); Choi *et al.* (2019)	Jin *et al.* (2013); Andert *et al.* (2017); Khayatian *et al.* (2018, 2019); Yang *et al.* (2016); Bashiri and Fleming (2017); Sharon *et al.* (2017); Chen *et al.* (2018); Shi *et al.* (2018); Lu and Kim (2019); Elhadef (2015); Sayin *et al.* (2018); Lu and Kim (2016); Qian *et al.* (2019); Azimi *et al.* (2015)

Table 2.1: Existing Intersection Management Algorithms Based on the Proposed Interface for Communication among Vehicles (or with Intersection Manager).

In QB-IM, the IM either accepts or rejects a request, while in AB-IM, IM explicitly assigns a reservation to the CAV. As a result, AB-IM algorithms can achieve higher throughputs compared to QB-IM ones but the processing time of the intersection manager for an AB-IM algorithm is more than a QB-IM.

Both centralized and distributed approaches have their own pros and cons but most importantly, in centralized approaches, the IM is a single point of failure and therefore less reliable compared to distributed approaches. Also, distributed approaches are more scalable since they don't require support from infrastructure and can be deployed at every intersection. In centralized approaches, CAV's control is given to the IM once it enters the intersection zone and given back to the CAV when it leaves the intersection area. On the other hand, CAVs need to broadcast their information periodically to let newly arrived CAVs know of their cross time, while in centralized techniques, IM stores the information about the state of the intersection (e.g. occupancy times-areas) and therefore, CAVs do not have to broadcast their in-

formation periodically. As a result, distributed techniques may have higher network overheads compared to centralized ones. Time synchronization is a fundamental part of the intersection management which has received less attention. Almost all centralized and distributed approaches require having the same notion of among all nodes in order to ensure the correctness of the intersection management and safety of CAVs. Since all CAVs are equipped with GPS receivers, they can maintain an accurate notion of time up to few microseconds. However, if GPS signals are poor/not available in an area, time synchronization should be a part of the intersection management's V2V/V2I interface.

2.2 Vehicle Dynamics

Typically, a model is needed to estimate/predict future trajectories of vehicles before and at the intersection. In the literature, researchers have considered different models for vehicle dynamics. Some existing works use a simple one-dimension model, while some use more complex models. Next, we will study some of the models that are used for the dynamics of vehicles. Figure 2.2 shows different approaches used to model the dynamics of a vehicle in existing works on intersection management of CAVs.

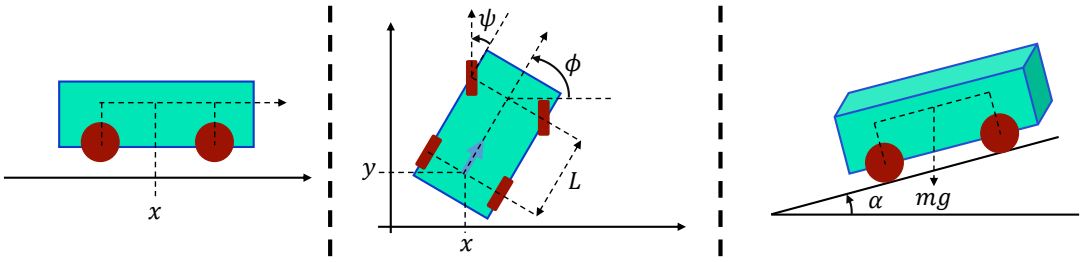


Figure 2.2: (a) Double Integrator Model - Considering the Longitudinal Movements of the CAV Only (b) 2D Model - Considering Longitudinal and Lateral Movements of the CAV (c) High-Fidelity Model - Considering the Road Slope and Aerodynamic Drag Force (F_d).

One-dimension Model (double integrator)

This model considers the longitudinal movements of the vehicle only.

$$\begin{cases} \dot{x} = v \\ \dot{v} = u \end{cases} \tag{2.1}$$

x and v are the longitudinal position and velocity of the vehicle and u is the input to the vehicle that captures the input to the throttle and brake for positive and negative inputs respectively.

4-wheel Model

This model considers both the longitudinal and latitudinal movements of the vehicle Dresner and Stone (2008b):

$$\begin{cases} \dot{x} = v\cos(\phi) \\ \dot{y} = v\sin(\phi) \\ \dot{\phi} = \frac{v}{L}\tan(\psi) \\ \dot{v} = u \end{cases} \tag{2.2}$$

x and y are the longitudinal and latitudinal position of the vehicle in Cartesian coordinate. v is the absolute velocity of the vehicle and u is the input to the vehicle that captures the input to the throttle and brake for positive and negative inputs respectively. ϕ is the heading angle of the car, ψ is the steering angle of the vehicle and L is the wheelbase distance.

Bicycle model

This is a simplified version of the 4-wheel model which is created by projecting front and rear wheels onto two virtual wheels located at the middle of the car. The vehicle dynamics for the bicycle model can be written as:

$$\begin{cases} \dot{x} = v_x cos(\theta) - v_y sin(\theta) \\ \dot{y} = v_x sin(\theta) + v_y cos(\theta) \\ \dot{\theta} = r \end{cases} \tag{2.3}$$

where v_x and v_y are the longitudinal and lateral velocities of the vehicle respectively and r is the yaw rate.

Modeling Air Drift, Road Slope, and Mass

This model considers the effect of air drag force and road slope in the vehicle model-Bian *et al.* (2019).

$$\begin{cases} \dot{x} = v \\ \dot{v} = \frac{\eta}{mr} T - \frac{C_A}{m} v^2 - g\big(\sin(\alpha) + f\cos(\alpha)\big) \end{cases} \tag{2.4}$$

where T is the torque applied to wheels, η is the mechanical efficiency of the driveline, m the mass r is the tire radius, C_A is aerodynamic drag coefficient, f is the rolling resistance, g is the gravitational acceleration and α is the road slop.

We have categorized existing works on intersection management of CAVs based on the considered model for the vehicle dynamics in Table 2.2.

The double integrator model is linear and therefore is easy to work with because the solution for the behavior can be determined analytically. However, it does not

Single dimension model (double integrator)	2D model (4-wheel vehicle)	With mass, air drift, and road slope
Malikopoulos and Zhao (2019a); Fayazi and Vahidi (2017); Malikopoulos and Zhao (2019b); Li *et al.* (2019a); Colombo and Del Vecchio (2012); Feng *et al.* (2018); Mahbub *et al.* (2019); Zhang *et al.* (2017); Murgovski *et al.* (2015); Khoury and Khoury (2014); Mirheli *et al.* (2019); Liu *et al.* (2019b); Steinmetz *et al.* (2018); Katriniok *et al.* (2018, 2019); Karlsson *et al.* (2018); Zhao and Malikopoulos (2018); Hafizulazwan (2018); Zhao *et al.* (2018); Fayazi and Vahidi (2018); Shi *et al.* (2018); Makarem and Gillet (2011); Lee *et al.* (2013); Yang *et al.* (2016); Lin *et al.* (2017); Zhang *et al.* (2016); Au and Stone (2010); Li and Wang (2006); Guler *et al.* (2014); Neuendorf and Bruns (2004); Lee and Park (2012); Onieva *et al.* (2015); Zohdy *et al.* (2012); Kloock *et al.* (2019); Filocamo *et al.* (2020); Wang *et al.* (2020); Hadjigeorgiou and Timotheou (2019)	Khayatian *et al.* (2019); Fajardo *et al.* (2011); Khayatian *et al.* (2018); Andert *et al.* (2017); Dresner and Stone (2008b); Guney and Raptis (2020)	Bashiri *et al.* (2018); Bian *et al.* (2019); Bichiou and Rakha (2018)

Table 2.2: Existing Works on Intersection Management of Cavs Categorized by the Considered Vehicle Dynamics

capture the movement of the vehicle in 2D space. To model the behavior of a CAV even more accurately, different factors like air drift, mass, friction, road slope can be considered. However, considering a high-fidelity model will put a burden on the scheduling system since more computation is needed to estimate the behavior of the CAVs and determine a feasible solution –especially in optimization-based approaches. As a result, it remains an open problem to determine the right level of fidelity. There are many parameters that should be considered to model the actual behavior of a CAV where some of them are variable e.g. road slope, wind, the mass of the vehicle, road friction coefficient, etc. Therefore, accurate prediction of the behavior of a CAV

requires an online identification mechanism to estimate such parameters.

2.3 Conflict Detection

In order to detect a possible conflict that two CAVs may have at the intersection, existing works have proposed two approaches: 1) considering a Spatio-temporal occupancy map for the intersection area and 2) considering the expected trajectories of CAVs inside the intersection.

(a) The grid represents the areas that will be occupied by vehicles at time t. A conflict exists if two areas have an overlap (depicted in red).

(b) Predefined paths are defined for crossing the intersection. A conflict exist if two paths cross and the cross times of the vehicles overlap.

Figure 2.3: Modeling a Conflict at the Intersection.

The first approach models the intersection as a grid of conflict areas and the path of a CAV inside the intersection is captured by indicating which blocks (of the grid) will be occupied by a CAV at each time-step. In this approach, the intersection management algorithm needs to make sure two CAVs are not scheduled to occupy a block at the same time. The granularity of splitting the intersection area into a grid varies among different approaches. In the extreme case, the whole intersection area is considered as a conflict area.

In the second approach, there is no need to store the occupancy map for the whole intersection area, instead, the expected path of two CAVs is used to determine the location at which two CAVs may have a conflict. This can be done offline as the

expected paths of CAVs are known e.g. for going straight or making a turn.

We have categorized existing works in terms of the way conflicts are modeled in Table 2.3.

Conflict Detection using Occupancy Map	Conflict Detection using on CAVs' Trajectory
Li and Wang (2006); Azimi *et al.* (2014); Aoki and Rajkumar (2018); Liu *et al.* (2018); Belkhouche (2018); Bian *et al.* (2019); Filocamo *et al.* (2020); Dresner and Stone (2008b); Liu *et al.* (2019a); Choi *et al.* (2019); Jin *et al.* (2013, 2012a); Bento *et al.* (2012); Andert *et al.* (2017); Azimi *et al.* (2015); Sayin *et al.* (2018); Qian *et al.* (2019); Qiao *et al.* (2018); Khoury and Khoury (2014); Bentjen (2018); Lu and Kim (2016); Pourmehrab *et al.* (2017); Fayazi *et al.* (2017)	Katriniok *et al.* (2018, 2019); Yang *et al.* (2016); Khayatian *et al.* (2019); Lu and Kim (2016); Khayatian *et al.* (2018)

Table 2.3: Categorizing Existing Works in Terms of Modeling the Conflicts Inside the Intersection Area.

Using an occupancy grid to model conflicts is computationally cheap since it involves simple boolean checking operation, however, the computation increases by considering finer conflict zones and smaller time-steps. The throughput of the intersection is directly dependent on the granularity of the Spatio-temporal grid and generally finer grids can achieve higher throughputs.

2.4 Scheduling Policy

The main purpose of intersection management is achieving higher throughputs compared to conventional traffic lights while ensuring the safety of vehicles. In this paper, the process of deciding which CAV should cross the intersection first and which CAV should cross second and so on is called "scheduling". We group existing scheduling policies into three main categories: i) First-Come First-Served, ii) Heuristic, and

iii) Optimization-based. Figure 2.4 shows an example of an intersection and possible solutions determined using the FCFS, optimization-based and a heuristic approach.

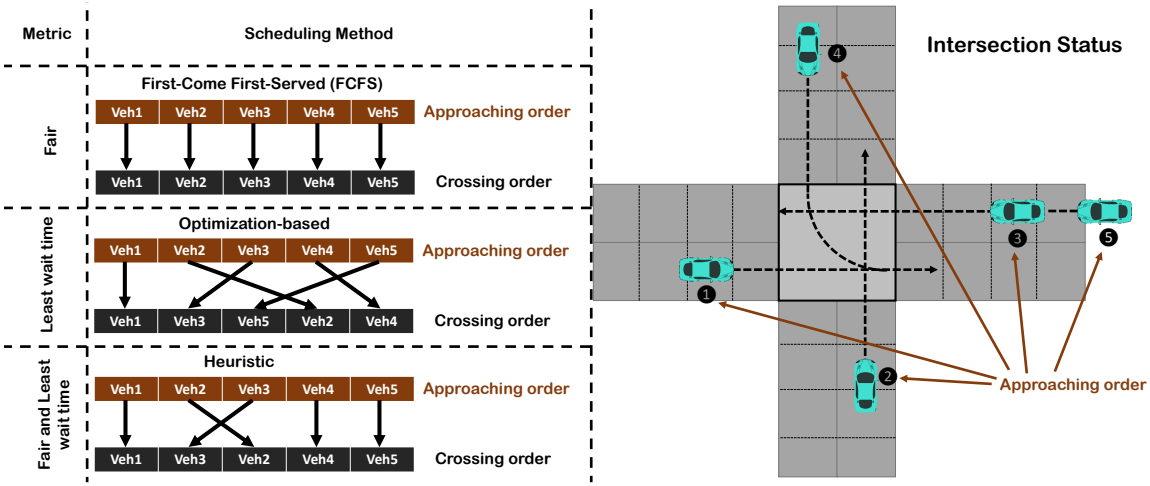


Figure 2.4: Examples of FCFS, Optimization-based and Heuristic Scheduling Policies. In the Left Section, the Approaching and Crossing Order of Vehicles is Indicated. In the Right Section, the Status of the Intersection at the Scheduling Time is Depicted.

2.4.1 First-Come First-Served Approaches

First Come First Served (FCFS) traffic control algorithms works as the name sounds, the first vehicle to arrive is the first vehicle to be served and grants entry to the intersection. One of the first implementations of an FCFS method is AIM which was proposed by Dresner *et al.* Dresner and Stone (2008b). Requests to the intersection manager are processed in the same order they are received. For scheduling the cross of a vehicle, AIM stores a reservation grid for the area of the intersection. This segmentation can be used to check if another vehicle is occupying the space at a time. FCFS was the scheduling policy for many other intersection management techniques. For instance, Azimi *et al.* (2014) considers a reservation map with smaller segmentation, Bentjen (2018), Khoury and Khoury (2014) and Qiao *et al.* (2018) similarly consider a reservation area for the intersection, Elhadef (2015) uses a predefined con-

flict table between entry lanes of the CAVs and a locking mechanism, and Khayatian *et al.* (2018), Andert *et al.* (2017) and Khayatian *et al.* (2019) use predefined trajectories of the vehicles inside the intersection for reservation. In Jin *et al.* (2013), Jin *et al.* proposed to use FCFS for platoons of CAVs instead of individual vehicles where the IM uses a reservation table to schedule the next platoon. In Filocamo *et al.* (2020), a priority value is calculated for each CAV based on the arrival time and the priority specifies the crossing order of CAVs. Lu Lu and Kim (2016) *et al.* is another FCFS approach that a queue of CAVs is created and the intersection manager serves the top CAV in the queue by assigning a time slot.

2.4.2 Optimization-based Approaches

Despite FCFS scheduling methods, optimization-based approaches try to minimize the average travel time of the whole intersection regardless of their approaching order. As a result, the crossing order of vehicles may vary from the approaching order of vehicles.

There have been several optimization-based approaches that solve the intersection management scheduling problem. The simplest type of optimization-based scheduling is done by controlling the status of the traffic light namely Signal Phase, and Timing (SPaT) to achieve a high throughput Emami *et al.* (2018); Pourmehrab *et al.* (2017); Liu *et al.* (2019b); Fayazi and Vahidi (2018); Fayazi *et al.* (2017); Bian *et al.* (2019); Xu *et al.* (2019). In such approaches, the IM suggests an optimal trajectory for the CAVs to follow such that they will hit a green light. Fayazi and Vahidi (2018) and Fayazi *et al.* (2017) use Mixed Integer Linear Programming (MILP) to solve the optimization problem and Ashtiani *et al.* (2018), extends it to a grid of connected intersections.

Researchers have also proposed optimization-based approaches for an intersection

without a traffic light. Generally, the goal is to increase the throughput which is formulated as minimizing the travel time/wait time/cross-time Hubmann *et al.* (2017); Gregoire *et al.* (2013); Lee and Park (2012); Lin *et al.* (2017); Zohdy *et al.* (2012); Bichiou and Rakha (2018). To avoid a collision in the intersection area, a set of constraints are defined based on the unsafe states e.g. two vehicles be very close to each other at any time. Hubmann *et al.* (2017) uses a POMDP (partially observable Markov decision process) to model vehicle dynamics and the Adaptive Belief Tree (ABT) algorithm for finding the optimal solution. Xu *et al.* approach Xu *et al.* (2019), similarly creates a tree for all the possible solutions for the passing order where the leaf of the tree represents the complete solution.

Guler *et al.* Guler *et al.* (2014) proposed an iterative algorithm to find the optimal arrival/departure sequence of CAVs. In Yang *et al.* (2016), they extended their work and formulated the intersection management problem using two optimization problems: 1) finding the optimal arrival/departure sequence of CAVs, and 2) finding the optimal trajectory of each vehicle once arrival/departure times are known. They propose to use the Branch-and-Bound approach to find the optimal arrival/departure sequence. Guney and Raptis (2020) also solves an optimization problem to minimize the delay of CAVs. This paper employs the particle swarm optimization (PSO) algorithm to find the optimal solution. Lu *et al.* Lu and Kim (2019) solve the optimization problem using MILP to minimize the travel time. Liu *et al.* Liu *et al.* (2018) propose to convert a centralized optimization problem into distributed optimization problems that are solved locally on each vehicle to find the optimal solution. In Bashiri and Fleming (2017), a platoon-based approach is introduced to find the optimal solution that yields minimum average delay. Similarly, Timmerman *et al.* Timmerman and Boon (2019) propose an optimization-based approach for platoons of vehicles. In Li and Wang (2006), Li *et al.* proposed to create a tree where each node corresponds to

a valid schedule. The optimal entrance of the vehicles is then determined by traversing the tree. Zhu and Ukkusuri (2015) studies the problem of managing a grid of intersections where the traffic flow of each link should be determined. Linear programming is used to solve the problem. In Jin *et al.* (2012b), Jin *et al.* linearizes the optimization problem using the big M method and then, solves it using linear programming to find the minimum travel time of vehicles. Onieva *et al.* (2015) uses a fourth-order Laplace model for vehicle dynamics and use the multi-objective fuzzy rule-based system to find the minimum travel time of vehicles. In all aforementioned approaches, a goal function was defined based on the travel time of the vehicles and dynamics of the vehicle, and safety specifications were modeled as constraints.

There are other approaches that consider velocity variation Murgovski *et al.* (2015); Mirheli *et al.* (2019); Katriniok *et al.* (2018); Karlsson *et al.* (2018); Shi *et al.* (2018); Katriniok *et al.* (2019); Malikopoulos and Zhao (2019b); Kloock *et al.* (2019), passenger discomfort Murgovski *et al.* (2015); Katriniok *et al.* (2018); Zhang *et al.* (2017), communication overhead Steinmetz *et al.* (2018), acceleration/deceleration variation Shi *et al.* (2018), absolute acceleration/deceleration amount Karlsson *et al.* (2018); Zhao and Malikopoulos (2018); Zhao *et al.* (2018); Zhang *et al.* (2016); Shi *et al.* (2018); Lee *et al.* (2013); Kloock *et al.* (2019); Wang *et al.* (2020); Hadjigeorgiou and Timotheou (2019) and fuel consumption Mahbub *et al.* (2019); Zhang *et al.* (2017); Hafizulazwan (2018); Wang *et al.* (2020) as a metric and define the goal function based on it. In Murgovski *et al.* (2015), Murgovski *et al.* reformulate the optimization problem into a sub-problem by finding the optimal entrance order of vehicles and then transformed it into a convex problem. Zhao and Malikopoulos (2018), Zhao *et al.* (2018), Zhang *et al.* (2016), Wang *et al.* (2020) and Malikopoulos and Zhao (2019a) follow optimal control approaches and use the Euler-Lagrange equation to solve the optimization problem analytically. In Lee *et al.* (2013), the opti-

mization problem is solved in three steps using Active-set Method (ASM), Sequential Quadratic Programming (SQP) and Genetic Algorithm (GA). Philippe *et al.*Philippe *et al.* (2019) propose to create a local utility function for each CAV that is a function of the inverse of distance every two CAV, the difference from the maximum velocity and difference from the initial velocity. Then, the Probability Collectives (PC) method is used to optimize the utility function. In Li *et al.* (2019b), authors propose to use Discrete Forward-Rolling Optimal Control (DFROC) to minimize the total delay of CAVs.

2.4.3 Heuristic Scheduling Approaches

Heuristic approaches take another way to solve the intersection management problem that isn't guaranteed to be optimal but is sufficient for reaching the immediate goal. For instance, researchers from MIT have proposed a scheduling algorithm called BATCHTachet *et al.* (2016) with a designating reordering period. When the IM receives a request it doesn't assign a velocity to the vehicle immediately. Instead, it waits for a designated time period and keeps the record of all requests. Once the period is over, it re-orders the entrance time of vehicles to get a better schedule. The most efficient pattern of entry is chosen. Stevanovic *et al.* proposed a quite different approach to manage the intersection through the re-arrangement of the typical lane configuration so that there are fewer conflicts in the roadway itselfStevanovic and Mitrovic (2018).

Another heuristic approach is a bidding system to resolve conflicts within CAVs Vasirani and Ossowski (2012). Vehicles can bid currency to beat out other vehicles to get reservations for the intersection. In many cases, a vehicle has to pay for the reservation of vehicles in front of it too in order to clear the queue. Wei *et al.* Wei *et al.* (2018) follow a game-theory approach to find a schedule that has the

least conflicts. Another heuristic approach is proposed by Jin *et al.* Jin *et al.* (2012a) where a mixture of a priority-based and an FCFS is implemented, where vehicles with higher priorities are processed earlier. In a similar work, Elhenaway *et al.* Elhenawy *et al.* (2015) propose a game theory-based heuristic based on the chicken game, where vehicles approaching the intersection have a joint utility function associated with each action.

In Li *et al.* (2019a), Li *et al.* proposed a similar approach where a reward function is defined based on two metrics: crossing the intersection in a timely manner, not hitting any vehicles, and keeping a reasonable distance from other vehicles. Makarem *et al.* Makarem and Gillet (2011) propose a method based on a local navigation function that takes into account a vehicle's size and ability to accelerate/decelerate quickly when being scheduled. Au and Stone (2010) follows a heuristic approach, where the IM determines the highest possible velocity of arrival that a vehicle can achieve and then selects the schedule that yields the earliest time of arrival.

Aoki *et al.* Aoki and Rajkumar (2018) propose a heuristic approach that is created from the integration of the FCFS policy and a timeout policy. CAVs are normally served based on the FCFS but when the wait time of a CAV is greater than a threshold, it interrupts the operation of the intersection and lets the CAV with excessive wait time to pass. Wu *et al.* Wu *et al.* (2019) proposed a reinforcement learning approach to figure out a policy that is collision-free. The Q-learning method was used to update the policy and intersection delay was used as the reward. In Belkhouche (2018), Belkhouche *et al.* presented a heuristic approach that finds the best crossing order based on the safety margins defined for crossing without collision. Another heuristic scheduling approach is presented in Sayin *et al.* (2018) where vehicles report their utility function to the IM and the IM determines a schedule such that it maximizes the utility values of all vehicles while maintaining the fairness when possible.

Bruni *et al.* (2013) and Colombo and Del Vecchio (2012) look at the intersection management as a verification problem where the goal is to check if there exists an input that such that the system can avoid the set of *Bad States* or an unsafe situation. Wu *et al.* Wu *et al.* (2012) propose to use the current best known local solution using the Ant Colony Optimization (ACO) approach to find the minimum wait time of vehicles. Choi *et al.* (2019) proposes to create a Red-Black Tree from conflicts and then traverse the tree and find the earliest time that the slot is available. Buckman *et al.* Buckman *et al.* (2019) propose a modified version of FCFS to schedule CAVs where a negotiation occurs between CAVs in the form of pairwise swapping. They use Social Value Orientation (SVO) to create a utility function and a swap occurs only when the summation of utility functions is increased.

We have categorized existing works based on their scheduling policy in table 2.4.

The scheduling policy of intersection management is directly related to the throughput of vehicles. In addition to throughput, fairness is a key metric in determining the scheduling policy because waiting for a long time may not be acceptable for most people. The FCFS algorithm fulfills the fairness requirement and vehicles will not wait for an improperly long time. However, FCFS may not be efficient and its performance degrades significantly as the intersection scales.

There is a tradeoff between fairness and the overall throughput that an approach achieves. We believe that both throughput and fairness are important metrics and should be taken into account for realistic implementations. On the contrary, a heuristic method can achieve better throughput compared to FCFS and all vehicles will eventually receive a reservation i.e. vehicle delay is bounded. Another disadvantage of optimization-based approaches is the delay due to the processing time of the intersection management for finding the optimal schedule, and it becomes worse as the intersection scales. On the other hand, analytical optimization-based approach and

FCFS	Optimization-based	Heuristic
Fajardo *et al.* (2011); Fok *et al.* (2012); Dresner and Stone (2008a); Quinlan *et al.* (2010); Sharon and Stone (2017); Stone *et al.* (2015); Dresner and Stone (2007); Sharon *et al.* (2017); Hausknecht *et al.* (2011); Azimi *et al.* (2014); Bentjen (2018); Khoury and Khoury (2014); Qiao *et al.* (2018); Elhadef (2015); Khayatian *et al.* (2018); Andert *et al.* (2017); Jin *et al.* (2013); Filocamo *et al.* (2020); Lu and Kim (2016)	Hubmann *et al.* (2017); Gregoire *et al.* (2013); Murgovski *et al.* (2015); Onieva *et al.* (2015); Mirheli *et al.* (2019); Steinmetz *et al.* (2018); Katriniok *et al.* (2018, 2019); Karlsson *et al.* (2018); Malikopoulos and Zhao (2019a); Zhao and Malikopoulos (2018); Hafizulazwan (2018); Lu and Kim (2019); Liu *et al.* (2019b); Zhao *et al.* (2018); Fayazi and Vahidi (2018); Liu *et al.* (2018); Shi *et al.* (2018); Lee *et al.* (2013); Lin *et al.* (2017); Bashiri and Fleming (2017); Fayazi *et al.* (2017); Ashtiani *et al.* (2018); Yang *et al.* (2016); Zhang *et al.* (2016); Zhu and Ukkusuri (2015); Zohdy *et al.* (2012); Lee and Park (2012); Jin *et al.* (2012b); Guler *et al.* (2014); Li and Wang (2006); Bian *et al.* (2019); Xu *et al.* (2019); Kloock *et al.* (2019); Wang *et al.* (2020); Hadjigeorgiou and Timotheou (2019); Guney and Raptis (2020); Li *et al.* (2019b); Bichiou and Rakha (2018)	Tachet *et al.* (2016); Stevanovic and Mitrovic (2018); Vasirani and Ossowski (2012); Wei *et al.* (2018); Jin *et al.* (2012a); Aoki and Rajkumar (2018); Makarem and Gillet (2011); Elhenawy *et al.* (2015); Au and Stone (2010); Li *et al.* (2019a); Wu *et al.* (2019); Belkhouche (2018); Sayin *et al.* (2018); Bruni *et al.* (2013); Colombo and Del Vecchio (2012); Wu *et al.* (2012); Choi *et al.* (2019); Buckman *et al.* (2019)

Table 2.4: Categorizing Existing Works Based on Their Scheduling Policy.

heuristic approaches can avoid this problem.

2.5 Wireless Technology

Vehicle to everything (V2X) is a family of communication technologies that are used for information sharing of vehicles with other vehicles (V2V), infrastructure (V2I), and pedestrian (V2P).

Currently, two types of wireless technologies exist for connected vehicles: i) DSRC (Dedicated Short-range Communication) Chen *et al.* (2009) and 2) Cellular-V2X (C-V2X). DSRC uses 802.11p protocol at the physical layer Abdelgader and Lenan (2014) and its network architecture and security are defined by IEEE WAVE standards Li (2010). DSRC uses SAE J2735 Misener (2016) standards to define message format

at the application layer and J2945/x Misener (2016) family of standards for defining performance requirements of different V2X scenarios. One of the important messages in DSRC is Basic Safety Message (BSM) Perry (2019), which is proposed to be used as a way to share information in some of the intersection management papers Azimi *et al.* (2013b); Aoki and Rajkumar (2018); Azimi *et al.* (2013a, 2014, 2012); Qian *et al.* (2019). It should be noted that most of the existing works do not specifically mention what wireless technology they propose to work.

C-V2X is a 3GPP communication technology Wang *et al.* (2017) that works with the cellular network and has controlled Quality of Service (QoS) Toukabri *et al.* (2014). C-V2X has two modes of operation, cellular communication (Uu) and direct communication (PC5). Uu mode enables V2V communications through the cellular network while PC5 allows for direct communication among vehicles similar to DSRC. DSRC achieves low latencies and high reliability when a few vehicles are present, however, its performance deteriorates in a dense environment with many vehicles. C-V2X, on the other hand, has shown more reliable latencies even in dense environments. In terms of communication range, DSRC is more suitable for low-range communications, while C-V2X can provide long-range communications. Compared to C-V2X, DSRC has been tested more often due to its availability (from 2017) Romero *et al.* (2020). Since DSRC uses message broadcasting, it benefits from user anonymity but will be inefficient as point-to-point communication is not possible.

	DSRC	C-V2X
Pros	good hardware support, proved to work with J2735 messages, Anonymity of users	Long range communication support, can perform point-to-point communication
Cons	limited range, message are broadcast only, may not reliable in dense areas	limited hardware support

Table 2.5: Comparing DSRC with Cellar-V2X

2.6 Managing Multiple Intersections

Since a city can be broken down into a grid of intersections, effective intersection management of CAVs is key to city-wide traffic management. Hausknecht *et al.* Hausknecht *et al.* (2011) extended the AIM approachDresner and Stone (2008b) and proposed an intersection management policy for a grid of intersections. In this approach, the intersection manager estimates the delay of traffic using 4 features: i) the total number of active vehicles (TAV) that exists within the range of the intersections, ii) the total number of active vehicles along the planned path (PAV), iii) the previously calculated PAV (oPAV) in the last step, and iv) the average traversal time for the planned trajectory (TWA). The estimated traversal time of a vehicle is then calculated as:

$$T_{est.} = 0.09TAV + 0.83PAV + 0.25oPAV + 0.25TWA + 2.26 \tag{2.5}$$

The above equation is determined by simulating a single intersection and linear regression approach. Once the estimated traversal time is determined for each vehicle, an A* search is performed to find the best scheduling. The proposed algorithm is evaluated for a 2x2 grid of intersections.

In a similar work Lin and Ho (2019), the problem of CAV routing is solved using an iterative A*. There are 3 steps in each iteration, i) batch processing stage, where the data of CAVs are collected using simulation, ii) routing stage, where A* is used to find the best route for vehicles, and iii) congestion checking stage, where vehicles are re-routed to avoid congestion. This approach predicts future traffic flows using simulation. This approach is evaluated on different sizes of intersections up to 9x9 using SUMO. Their iterative algorithm has shown better results compared to AIM's multi-intersection management approach.

In another work, a market-inspired Vasirani and Ossowski (2012) approach is proposed to manage a network of intersections. The idea is that CAVs bid a price to get a reservation in order to drive through the intersections and intersection managers will follow an auction-based approach to provide the reservation to CAVs. A model is provided for CAV drivers which considers the time of travel in a free-flow scenario and the price of the travel governed by the intersection managers. This approach is evaluated in a mesoscopic-microscopic simulator.

In a recent work Wang *et al.* (2020), authors propose a greedy algorithm to optimize the sequence for route planning in a grid of intersection.

Fine-grain information about the status (position, velocity, lane, route) of CAVs is more beneficial for intersection management compared to coarse-grain information like traffic flow. However, the processing of fine-grain information can be very compute-intensive and requires high-performance computing solutions.

2.7 Hybrid (Human-CAV) Intersections

Deployment of a fully autonomous intersection of CAVs is still far from happening since it is unlikely to have an intersection exclusively for CAVs only. The intermediate step will have a mixture of human-driven vehicles (HVs) and CAVs, which we refer to as *hybrid intersections*.

One of the first attempts to consider a hybrid intersection was a part of the AIM approach Dresner and Stone (2008b). Dresner and Stone proposed FCFS+Light, an intersection management mechanism that is integrated with a traffic light model. The intersection manager follows a query-based approach to assign the reservation to incoming CAVs and HVs will follow the normal traffic light rules. In a similar work, Sharon *et al.* proposed Hybrid-AIM (H-AIM) Sharon and Stone (2017), which was built on FCFS+Light. The main difference between FCFS+Light and H-AIM is that

in FCFS+Light, IM immediately rejects a reservation request that is received from a CAV if the light is red for the corresponding lane. While in H-AIM, IM rejects the request only if another vehicle with a green light is present at the intersection. H-AIM requires extra infrastructure to be integrated into the intersection management system to detect the presence of vehicles.

Semi-AIM Stone *et al.* (2015) is a modified version of AIM that allows HVs and semi-autonomous vehicles to make reservations similar to CAVs. An interface e.g. a button is designed for HVs to send a request to the IM. In semi-AIM, three vehicle models were considered: i) semi-autonomous with communication (SA-COM) only, where the driver is permitted to pass if the entire lane is available, otherwise, it has to slow down and follows the traffic signal, ii) semi-autonomous with cruise control (SA-CC), where the driver gives the control to the driver agent to guide the vehicle through the intersection. Afterward, the control is given back to the driver. The vehicle will enter the intersection if it can maintain its velocity. Otherwise, it will act like the SA-COM model. iii) semi-autonomous vehicles with adaptive cruise control (SA-ACC) where the vehicle sends an anchor request to the IM and follows the front vehicle and enters the intersection if there is any. Otherwise, it will follow the SA-CC model.

In another effort to consider HVs, researchers have considered a connected vehicle center (CVC) Lin *et al.* (2017) which can detect the movement and position of HVs through traffic detectors and set green periods for them to enter the intersection when they reach the edge of the intersection. By default, the light is red for all HVs and when the intersection is clear, CVC changes the light to green for HVs. A Fuzzy Rule-based System (FRBS) Onieva *et al.* (2015) was proposed for an intersection of CAVs and HVs where autonomous vehicles can detect the existence of HVs and take proper maneuver to avoid them. This approach does not use traffic light and is

limited to scenarios where HVs enter the intersection from one road.

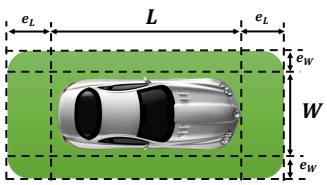
Fayazi *et al.* Fayazi and Vahidi (2017) proposed a device to be installed on the vehicle that suggests the desired speed (a range of speed) to the driver to follow so that it will reach the intersection at the desired time of arrival. This approach was tested on an actual vehicle and an API for the driver. In Shen *et al.* (2019), Shen *et al.* propose to use an On-Board Unit (OBU) to convey different communication signals to HVs. Two commands are envisioned for both CAVs and HVs, "pass" and "stop" and HVs are assumed to follow the command.

Supporting HVs at an automated intersection not only requires installing an extra device on vehicles, but it also needs training of drivers. Despite CAVs, HVs behavior may not be predictable and can disrupt the operation of the intersection. Therefore, the management approach should be flexible to handle HVs negligible mistakes or abnormal behaviors. Besides supporting human drivers, a management algorithm should account for pedestrians. So far, not much attention is paid to the management of pedestrians, and to the best of our knowledge, Niels *et al.* (2019) is the only work that considers scheduling of pedestrians.

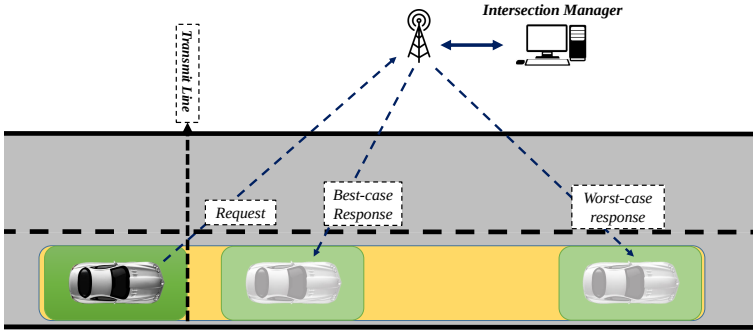
2.8 Safety and Robustness

Since intersection management is directly dealing with vehicles that transport humans, it should be safe and resilient against faults and uncertainties. Despite advances in localization approaches e.g. Simultaneous Localization and Mapping (SLAM) Bailey and Durrant-Whyte (2006), localization of autonomous vehicles is not perfect yet.

Therefore, the IM should consider a larger size of the CAV when reserving a space-time slot for a vehicle to ensures that vehicles don't collide. We refer to this barrier as *Safety Buffer*. The size of the safety buffer is directly related to the accuracy and



(a) Considering a safety buffer to account for localization errors and model mismatch



(b) Added safety buffer to account for round-trip delay.

Figure 2.5: Different Safety Buffers Considered to Account for Uncertainties

precision of sensors (encoder, IMU, GPS, camera, etc.) as well as the localization algorithms of the CAV and the maximum velocity of the vehicle. A common way to consider a safety buffer is depicted in Figure 2.5a (a). Figure 2.5a(b) depicts a safety buffer to account for position error due to round-trip delay. Besides position errors, there are a number of faults/anomalies that may occur during the operation of the intersection and can cause an accident. For example, a vehicle may break down inside the intersection or the intersection management software/hardware may crash.

Localization Errors: The AIM approach Dresner and Stone (2008b) considers a safety buffer around each vehicle to account for such uncertainties in the position due to inaccurate sensor readings (similar to Figure 2.5a). Belkhouche *et al.* Belkhouche (2018) follow another approach and consider a safety margin between the cross-time of vehicles to account for uncertainties in the position of CAVs.

Network Failures: Network delay is an inherent part of the intersection management algorithm because CAVs communicate over a wireless network. In existing papers on intersection management of CAVs, it is assumed that CAVs trust the information that is received from other CAVs and schedule their cross-time accordingly. As a result, the safety of CAVs depends on the authenticity of the information and

the timeliness of sending and receiving the information.

Processing time: In addition to network delay, checking the conflict between CAVs and determining a safe schedule –especially in optimization-based approaches takes time. Since CAVs are moving when waiting for a response from IM or other vehicles, the position at which they receive the response is dependent on the round-trip delay (RTD) i.e. from the moment they send a request and the moment they receive the response2.5b. Crossroads Andert *et al.* (2017) proposes to do synchronization and timestamping to make sure CAVs and the IM have the same notion of time. Andert *et al.* propose to assign a "time to actuate" to each CAV to make vehicles behavior deterministic. By considering an upper bound on the RTD, on-time actuation of CAVs can be guaranteed.

Vehicle Model Mismatch Another source of error is the considered model for CAVs. Any inconsistency between the actual model and the considered model can result in accidents inside the intersection. Additionally, a vehicle may face external disturbances like wind, bump, etc. that can deviate its behavior from the expected one. There are many intersection management approaches where a reference velocity profile is assigned to the CAV (to track). Although such approaches work fine in ideal situations, they are not robust to an external disturbance (e.g. wind) or model mismatches (e.g. a small mismatch in a parameter) and they can affect the eventual arrival time of the CAV at the intersection. RIM Khayatian *et al.* (2018) highlighted that the effect of bounded external disturbances and model mismatches can be compensated if a CAV tracks a reference position profile instead of a reference velocity profile.

Other Faults In the literature, researchers have modeled other sources of error and faults that can occur during the operation of the intersection. In a version of the AIM approach Dresner and Stone (2008a), authors assumed there is a way to

let the IM know an accident has happened e.g. when the airbag sensor triggers and then stop other vehicles by informing them. Another fault model that is considered is a pedestrian/obstacle that suddenly starts crossing the intersection Li and Wang (2006). Li *et al.* proposed a method where the first CAV that detects the pedestrian, lets other CAVs know that there is an obstacle so that all CAVs stop. Dedinsky *et al.* Dedinsky *et al.* (2019) propose to use infrastructure-mounted cameras to improve the robustness of the intersection against faults. In a recent study, Khayatian *et al.* Khayatian *et al.* (2020a) proposed an intersection management approach called R2IM that is resilient against a "rogue vehicle", which is referred to a CAV that does not follow the IM's command (stops or accelerates) or share wrong information (deliberately or unintentionally). R2IM approach considers a large gap between the cross-time os CAVs to ensure the safety in the presence of a rogue CAV. It was proved that no accident will happen inside the intersection area as long as there is one rogue vehicle at a time. To avoid accidents, the intersection management approach should have certain detection methods. Not all scenarios can be detected from the exchanged data and therefore, there is a need for environmental sensors to doublecheck the status of the CAVs.

Safety and efficiency of the intersection management depend on the latency, range, and rate of the communication protocol. Since Intersection management has safety-critical timing constraints, bounded time communication is needed to make sure messages are delivered to vehicles on time. The communication range also plays a significant role in the correctness of intersection management and can affect efficiency. Since CAVs cannot communicate with the infrastructure or each other until they are close enough to the intersection, they should drive at a slower speed to make sure when they receive the information for the first time, they have enough time to safely slow down or in the worst-case stop if needed. Given the total amount of data that

each CAV needs to send and receive as well as the communication rate of the wireless technology is known, the maximum capacity of the intersection management can be determined in terms of the number of vehicles that can be present at the same time.

2.9 Graceful Degradation and Recovery

In reality, unexpected situations can happen which temporarily disrupt the normal operation of the intersection e.g., an emergency vehicle approaching the intersection or a CAV breaking down inside the intersection. The intersection management approach should have certain mechanisms to resume the operation of the intersection once the emergency situation is resolved. We refer to the process of resuming the operation of the intersection as "recovery".

The AIM approach Dresner and Stone (2008a) has an inherent recovery mechanism integrated with it since it follows a QB-IM approach. When an emergency is detected, the IM rejects all requests until the emergency situation is cleared. Afterward, the IM starts accepting requests and will schedule CAVs.

Li *et al.* Li and Wang (2006) propose a recovery approach for scenarios where a pedestrian suddenly attempts to cross the intersection. In this approach, another cooperative driving plan is regenerated when the road is cleared.

Resuming the operation of the intersection is crucial to the liveness of the system and in some scenarios, recovery may not be possible e.g. the intersection area is blocked due to an earthquake or falling tree. As a result, CAVs must have a built-in recovery algorithm to re-route.

2.10 Security Concerns

Security is an important aspect of any intersection management since vehicles communicate over a shared medium (wireless communication). Security concerns are

more serious in cooperative intersection management approaches since the vehicle that schedules the intersection can be malicious and cause a catastrophe.

Currently, modern vehicles have the potential of being the target of cyberattacks Checkoway *et al.* (2011). Such attacks can be done by physically accessing the vehicle e.g. connecting to the Controller Area Network (CAN) bus Koscher *et al.* (2010) or installing malicious applications Mazloom *et al.* (2016). Also, it can be done over wireless communication Checkoway *et al.* (2011), e.g. using Bluetooth or cellular channel. Similar attacks can be applied to the intersection management system. Chen *et al.* Chen *et al.* (2018) showed that a malicious agent can spoof the data that connected vehicles send to the Intelligent Traffic Signal System (I-SIG) and therefore, cause traffic congestion. In this attack, a malicious agent sends false data to deceive the I-SIG system and cause a traffic jam.

In Bentjen (2018), Bentjen *et al.* analyzed two attack scenarios: 1) Sybil Attack, where the Sybil attacker makes a false reservation or multiple reservations at a time. They showed that certain reservations that have the most number of conflicts with other paths will have the most significant effect on traffic congestion. 2) Squatting attack, where a CAV proposes to come to a complete stop within the intersection which forces the intersection manager to assign very low velocities to other CAVs and cause a traffic jam. The authors proposed to mitigate the Sybil by using a unique signed certificate for each message or installing environmental sensors to detect vehicles. They also proposed to mitigate the Squatting attack by specifying a lower-bound on the velocity of arrival that is proposed by CAVs.

Despite the fact that extensive research is done on cybersecurity of automobiles, not much research has been done on the cyber-security of intersection management systems. There can be different types of Sybil attack Douceur (2002) that may be applied to the intersection management system: i) Nuisance, adding a delay in com-

munication, ii) Herding, deceiving several intersection managers to control a variety of cars, iii) Carjacking where the attacker spoofs the assigned speed for one or multiple cars Bentjen (2018).

2.11 Comparison of Evaluation Methods

In this section, we summarize the evaluation method of existing approaches. Some previous works use existing simulation tools, some developed their own simulation from scratch, some implemented an intersection with scale model vehicles, and some performed vehicle-in-the-loop (VIL) testing. Figure 2.6 shows an overview of some of the existing methods of evaluation. We categorized existing intersection management works based on their evaluation methods in Table 2.6.

Figure 2.6: Researchers have Evaluated Their Algorithms using Existing Simulators, Simulator that They Have Built from Scratch, Scale-Model Intersections or Vehicle-In-The-Loop testing. Top Row From Left, 1) A Simulator Developed in Java for AIM Approach Dresner and Stone (2008b), 2) Gazebo, 3) VISSIM, 4) AutoSIM, 5) A 1/12 Scale Model Intersection by Fok *et al.* Fok *et al.* (2012) 6) A 1/25 Scale Model Intersection by Beaver *et al.* Beaver *et al.* (2019). Bottom Row from Left, 1) A simulator Developed in MATLAB Khayatian *et al.* (2018), 2) SUMO, 3)Vehicle-in-the-loop Testing by Fayazi *et al.* Fayazi and Vahidi (2017), 4) A 1/20 Scale Model Intersection by Wu *et al.* Wu *et al.* (2012), and 6) A 1/8 Scale Model Intersection by Khayatian *et al.* Khayatian *et al.* (2018).

SUMOCenter (2019) and VISSIMPTVAG (2019) are the most popular simulators that are used by the researchers. AutoSIMAutoSIM (2019), GazeboGazebo (2019),

Their Own Simulators	VISSIM PTVAG (2019)	SUMO Behrisch *et al.* (2011)	Other Simulators	Scale Model Car	Vehicle in the loop
Dresner and Stone (2008b); Zohdy *et al.* (2012); Guler *et al.* (2014); Elhenawy *et al.* (2015); Yang *et al.* (2016); Vasirani and Ossowski (2012); Ashtiani *et al.* (2018); Bashiri and Fleming (2017); Lee *et al.* (2013); Zhao *et al.* (2018); Liu *et al.* (2019b); Hafizulazwan (2018); Zhao and Malikopoulos (2018); Onieva *et al.* (2015); Zhang *et al.* (2017); Li *et al.* (2019a); Malikopoulos and Zhao (2019b); Pruekprasert *et al.* (2019); Stone *et al.* (2015); Sharon *et al.* (2017); Fajardo *et al.* (2011); Dresner and Stone (2008a); Sharon and Stone (2017); Dresner and Stone (2007); Hausknecht *et al.* (2011); Bentjen (2018); Khoury and Khoury (2014)	Lee and Park (2012); Zhang *et al.* (2016); Lin *et al.* (2017); Lee *et al.* (2013); Le Vine *et al.* (2015); Chen *et al.* (2018); Zhao and Malikopoulos (2018); Emami *et al.* (2018); Mirheli *et al.* (2019)	Fayazi *et al.* (2017); Jin *et al.* (2013, 2012a,b); Lu and Kim (2019); Sayin *et al.* (2018)	Azimi *et al.* (2014); Aoki and Rajkumar (2018); Bento *et al.* (2012); Fayazi and Vahidi (2018); Mahbub *et al.* (2019); Stevanovic and Mitrovic (2018)	Andert *et al.* (2017); Khayatian *et al.* (2018); Fok *et al.* (2012); Wu *et al.* (2012); Paull *et al.* (2017); Beaver *et al.* (2019)	Quinlan *et al.* (2010); Fayazi and Vahidi (2017)

Table 2.6: Categorizing Existing Works Based on Their Evaluation Approach.

and Synchro are other simulators that have been used by researchers. For a more realistic evaluation, researchers have developed scale modelFok *et al.* (2012); Andert *et al.* (2017); Wu *et al.* (2012); Khayatian *et al.* (2018). There have been a few implementations that include full-size vehicleFayazi and Vahidi (2017); Quinlan *et al.* (2010) that are conducted using VIL.

Among existing simulators, SUMO is suitable for large-scale simulation and fast execution where the graphics are not important (simulates in 2D). SUMO, however, uses a simple model for vehicle dynamics and therefore cannot model the behavior of vehicles accurately. Similarly, VISSIM can perform large-scale simulations but it

provides a 3D view and it can integrate high fidelity models (e.g. from CarMaker). VISSIM is relatively slower than SUMO. Both SUMO and VISSIM can model pedestrians too. Gazebo simulator has a good physics engine and graphical representation. Gazebo can simulate multiple vehicles in 3D and accurately simulate vehicle sensors including LIDAR, Camera, RADAR, Ultrasonic, etc. Gazebo, however, compute-intensive and requires a high-performance computer to run smoothly when modeling multiple vehicles. Synchro and AutoSIM are other simulators that are not well documented and rarely used. The integration of an intersection management algorithm with Synchro and AutoSIM is challenging.

Currently, the state-of-the-art approach for intersection management of vehicles (either AVs, CAVs or human-driven vehicles) is through controlling the traffic light and signal free approaches have not been deployed yet to the best of our knowledge. Signal-free approaches are expected to be tested on private test tracks like M-City Briefs (2015), GoMentum StationCosgun *et al.* (2017), or Taiwan Car LabTsai *et al.* (2019) first before the actual deployment on public roads.

Simulation-based evaluations are simpler to implement and reproduce, and easier to scale. However, a simulation may not capture all challenges of an actual deployment. For instance, the effect of network delay, vehicle model mismatch, computation time on the operation of the system, and the need for implementing clock synchronization, fail-safe routines, etc. are some challenges of a real implementation.

Chapter 3

CROSSROADS+ APPROACH

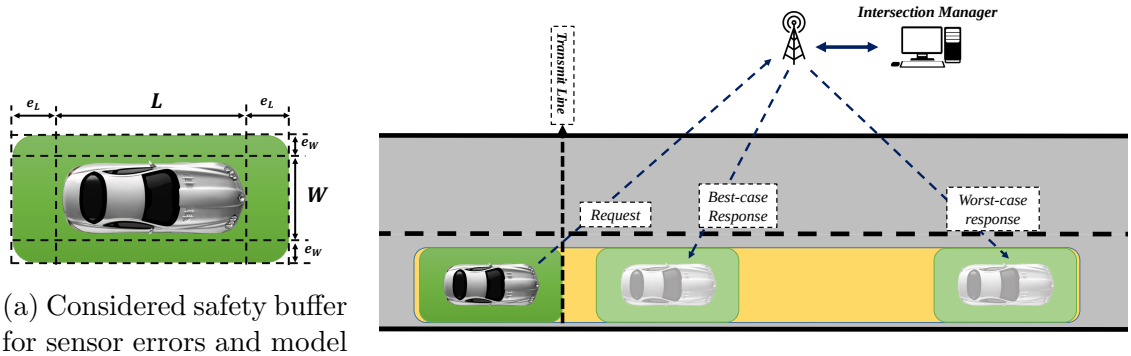
In the past few years, different methodologies have been proposed to manage intersections of CAVs. However, the timing problem due to RTD was ignored in almost all previous works. In this section, we propose Crossroads+ approach which addresses the problems due to RTD.

3.1 Backgrounds

We first start by providing some backgrounds on safety issues.

3.1.1 Safety Buffer

All localization devices used for autonomous driving like GPS, encoder, and odometer are subject to inherent errors. It's shown that Kalman Filter Setoodeh *et al.* (2004, 2007) and Particle FilterWon *et al.* (2010); Gross *et al.* (2010) based approaches can be used to improve the accuracy of localization for an autonomous agent. Nowadays, SLAM (Simultaneous Localization and Mapping) algorithms Durrant-Whyte and Bailey (2006); Bailey and Durrant-Whyte (2006) are widely used to estimate CAV location with the help of landmarks in the environment. However, it has been shown that current SLAM approaches are still not perfect Fuentes-Pacheco *et al.* (2015). Noise, drift, unknown disturbance, nonlinear behavior, and model mismatch are some common causes that can result in discrepancies between the actual and estimated positions of a CAV. Different error models exist to account for uncertainties in position. For instance, GPS error can be modeled as a circle around the CAV; while shaft encoder's error drifts over time and is typically modeled as a longitudinal



(a) Considered safety buffer for sensor errors and model mismatch

(b) Added safety buffer to account for RTD.

Figure 3.1: Safety Buffer and its Extension to Account for Nondeterministic RTD

buffer in front of/behind a CAV. Due to these errors, the IM should consider a buffer around each CAV to avoid potential collisions, which we refer to as *Safety Buffer*. The size of the safety buffer is related to the accuracy and precision of positioning sensors (odometer, IMU, GPS sensor, etc.) as well as the estimation algorithms. In this paper, we consider the error model to be similar to Figure 3.1a.

3.1.2 Round-trip Delay

In a real deployment of an intersection management algorithm, CAVs and IM communicate over a wireless network and therefore, are prone to unknown communication delays. This delay is directly related to the amount of data that needs to be transmitted. For instance, sending two arrays of data (e.g., reference position and timestamps) will incur higher delays compared to transmitting a single variable (e.g., constant velocity).

In conventional VT-IM approaches, the target velocity is safe only if it is received by a CAV at where the original request was sent and is executed right away. In reality, however, the communication delay and IM processing time are not zero and the CAV receives the velocity later than when the IM expects. As a result, the CAV's

trajectory will be different from what the IM has calculated and this may cause an accident inside the intersection. Based on the initial velocity and the assigned

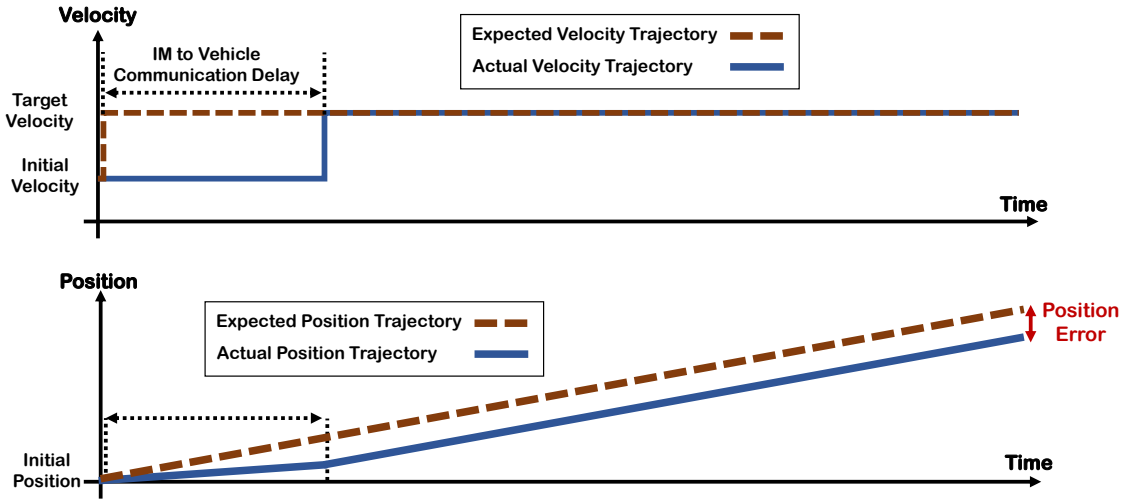


Figure 3.2: Ignoring the Network Delay Will Result in Position Error and Can Cause an Accident.

target velocity, a CAV may be ahead of or behind the expected position. If the assigned velocity is greater than the initial velocity and the RTD is not zero, then the actuation (acceleration) will be delayed and the CAV will enter the intersection later than expected and vice versa. Figure 3.2 shows an example of a CAV lagging behind the expected trajectory due to RTD (assuming instant velocity change).

One way to model the effect of communication delay and the processing time of the IM is to consider a larger safety buffer around each CAV. Figure 3.1b illustrates how the safety buffer needs to be extended longitudinally. The size of the buffer (B_{RTD}) depends on the WCRTD and the maximum possible velocity of CAVs

$$B_{RTD} = \Delta t_{WCRTD} \times v_{max} \tag{3.1}$$

Δt_{WCRTD} is defined as the duration of the worst-case delay from the time a CAV sends its information to the IM to the time it receives the response. The shorter the

WCRTD, the smaller the added buffer, and consequently, the better the throughput. This is because a much larger virtual size of the CAV is considered with an extended safety buffer.

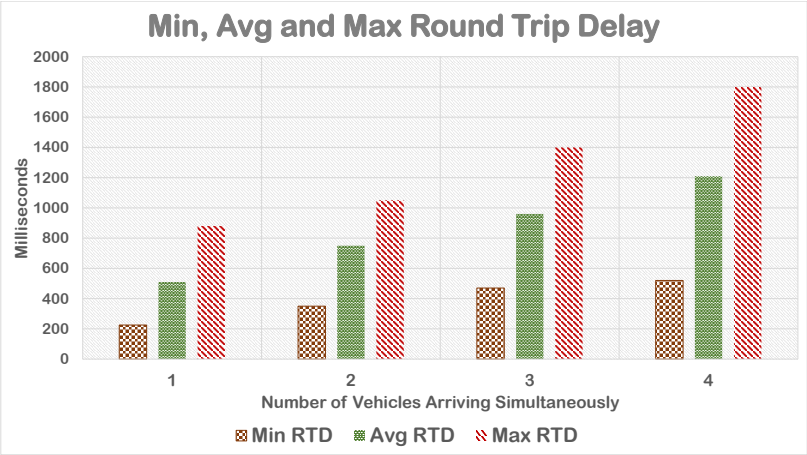


Figure 3.3: Round Trip Delay (RTD) for the Best-Case, Average-Case and Worst-Case Network Delay for 1-4 CAV(s) Sending a Request to the IM Simultaneously.

We measured the RTD when there are multiple CAVs sending requests to an IM simultaneously using our 1/10 scale model testbed (explained later). Figure 3.3 depicts the average, minimum and the maximum values of measured RTD for 4 scenarios, where the number of CAVs sending a request to the IM is equal to 1, 2, 3 and 4 respectively. We repeated each experiment 10 times. Note that for a 4-way intersection with one lane per road, the maximum number of CAVs that can send a request at the same time is 4. We can observe that the WCRTD increases as the number of CAVs increments.

3.2 Crossroads+ Algorithm

In this section, we present our Crossroads+ methodology. Algorithms 1 and 2 show the Crossroads+ interface for CAVs and the IM. Upon reaching the *sync line* (depicted in Figure 3.4), CAVs communicate with the IM to synchronize their lo-

cal clock. Synchronization is needed to have the same notion of time among all

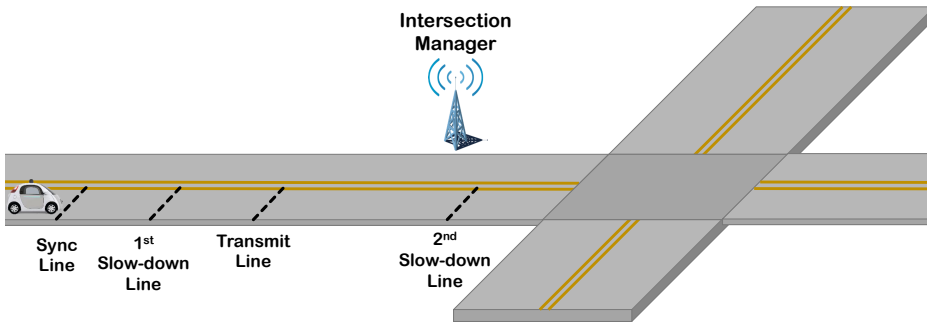


Figure 3.4: An Overview of the Crossroads+ Interface. IM Uses a FCFS Policy for Scheduling of CAVs.

nodes Shrivastava *et al.* (2016); Mehrabian *et al.* (2017); Shrivastava *et al.* (2017). When a CAV reaches the designated *transmit line*, it captures the time and sends its ID, position, velocity, maximum acceleration and deceleration rates, destination lane and the captured timestamp to the IM. After sending the request, CAVs will continue traveling with their initial velocity until either receiving a response from the IM or reaching the slow-down line. When the IM receives a request, it computes a target velocity and an actuation time based on the received information and status of other CAVs. When the CAV receives the target velocity and actuation time, it continues traveling at its current velocity until the actuation time, at which point it will take action to reach the assigned target velocity. If a CAV cannot synchronize its clock with the IM before reaching the *1st slow-down line*, it will apply the brake to stop behind the *transmit line*. If a CAV does not receive a response from the IM after the set timeout, it will request again. If no response is received from the IM before reaching the *2nd slow-down line*, the CAV will apply the brake and stop behind the intersection line. Slow-down lines are imaginary lines and are set for fail-safe operation of CAVs. Recovery from an event (such as a problem in the communication system or when an emergency car approaches) is out of the scope of this paper, but

is an important issue to be studied further.

Algorithm 1: Crossroads+ Approach - CAV

```
1  if Sync line is crossed then
2  |  Synchronize clock with IM;
3  |  while Waiting for Sync do
4  |  |  if reached the 1st slow-down line then
5  |  |  |  Apply brake;
6  |  |  end
7  |  |  if sync is successful then
8  |  |  |  Exit the loop;
9  |  |  end
10 |  end
11 end
12 if Transmit line is crossed then
13 |  Send a request to IM;
14 |  while waiting for the response do
15 |  |  if reached the 2nd slow-down line then
16 |  |  |  Apply brake;
17 |  |  end
18 |  |  if no response within the timeout then
19 |  |  |  Prepare to stop when the 2nd slow-down line is reached;
20 |  |  else
21 |  |  |  Exit the while loop;
22 |  |  end
23 |  end
24 |  Receive the response;
25 |  Wait until actuation time;
26 |  Accelerate or decelerate to achieve the target velocity;
27 end
```

3.2.1 Crossroads+ Scheduling Policy

Crossroads+ adopts an FCFS-based policy to ensure fairness among all CAVs. If two or more CAVs send their requests at the same time, the IM will use the CAVs' IDs to break the tie. When the IM receives a request from a CAV, it adds the CAV's information to the list of active CAVs. Accordingly, when a CAV leaves the

Algorithm 2: Crossroads+ Approach - IM

```
1  if a new msg is received then
2      if msg is a request then
3          Set the actuation time (t_act);
4          Find potential conflicts (x_c, t_c) w.r.t. existing CAVs ;
5          Calculate the target velocity (v_T^new) w.r.t. x_c, t_c;
6          Send back t_Act and v_T^new;
7          Add the requesting CAV's information to the list of active CAVs;
8      end
9      if msg is a leave notification then
10         remove the CAV's information from the list of active CAVs;
11     end
12 end
```

intersection, it will be removed from the list of active CAVs by the IM.

When a request is received, the IM sets the actuation time for the CAV, t_{Act}, to be:

$$t_{Act} = t_0 + \Delta t_{WCRTD} \tag{3.2}$$

where t_0 is the timestamp corresponding to the moment at which the requesting CAV has measured its status (position, velocity, etc.) and Δt_{WCRTD} is the considered WCRTD. We will later discuss how to determine a reasonable value of Δt_{WCRTD} for the intersection manager. The actuation time corresponds to a particular position along the vehicle's travel path, x_{Act}, since the CAV will drive at its initial velocity until t_{Act}:

$$x_{Act} = x_0 + v_0 \times \Delta t_{WCRTD} \tag{3.3}$$

where x_0 is the position of CAV at request time (t_0), and v_0 is the initial velocity of the CAV at t_0. The IM then computes all potential conflict points and corresponding conflict times along the requesting CAV's path with respect to each existing active CAV i. x_c and t_c are the sets of conflicts position (x_c^i) and times (t_c^i) calculated based on the set target velocities of existing active CAVs. For example, Figure 3.5

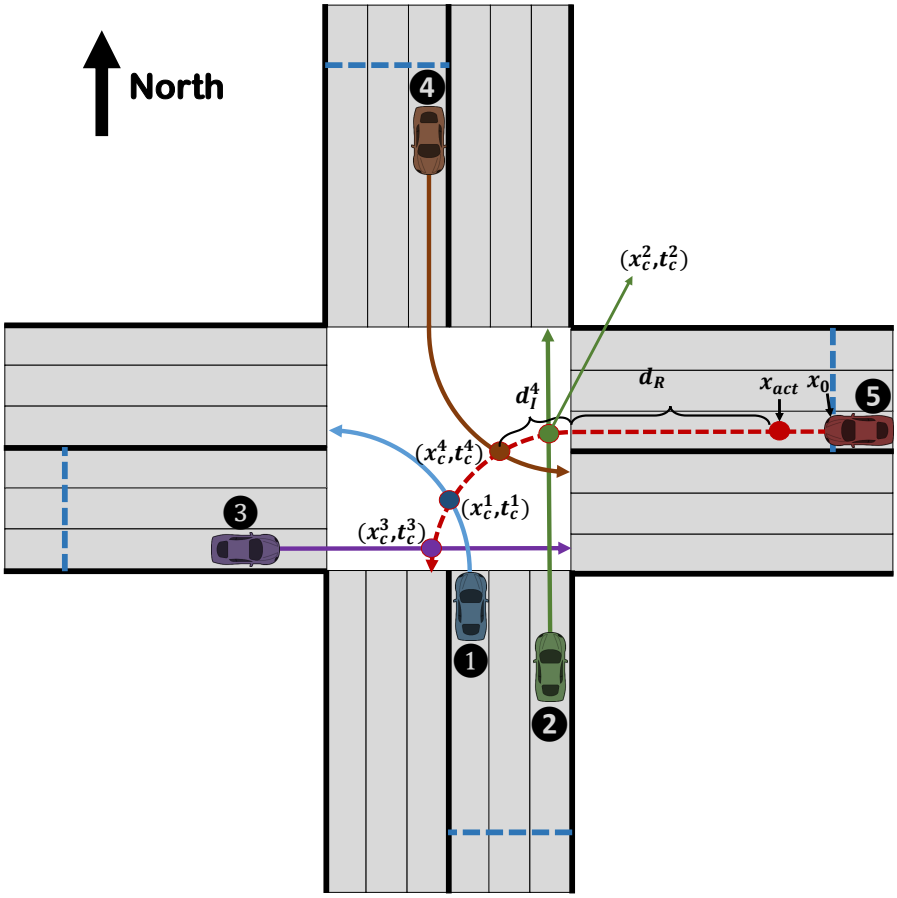


Figure 3.5: A View of a Four-Way Intersection with Three Lanes Per Road. Four Conflict Points Are Determined for the CAV Number 5 Regarding the Existing Ones.

illustrates the potential conflict positions and times calculated for the travel path of the requesting CAV, CAV #5. The travel path of CAV #5 is depicted by the dashed line. Other CAVs (#1, #2, #3 and #4) are on the list of active CAVs and have already received a target velocity and actuation time. Their paths are shown by solid lines. The intersections of paths of all CAVs with CAV #5 are denoted by solid dots. For example, the blue dot indicates the intersection between the paths of CAV #1 and CAV #5, i.e. conflict point x_c^1. t_c^1 is the time when CAV #1 reaches this point x_c^1. For CAVs on the same lane as the requesting CAV, the edge of the intersection is considered as the conflict point. With all potential conflict points determined, the

IM projects each conflict time (t_c^i) to a safe reach time (t_s^i):

$$t_s^i = t_c^i + \Delta t_{safety} \tag{3.4}$$

where Δt_{safety} is the delay buffer ensuring that the requesting CAV reaches the conflict position Δt_{safety} time units after the existing active CAV i. One way to determine the Δt_{safety} in order to achieve guaranteed safety is

$$\Delta t_{safety} = \frac{l_{max} + l_B}{v_{min}} \tag{3.5}$$

where l_B is the longitudinal size of the safety buffer due to sensor error, l_{max} is the length of the longest CAV among all CAVs and v_{min} is the slowest velocity that IM assigns to a CAV i.e., IM will not assign a velocity less than v_{min} to a CAV. The proof of safety will be presented in Section 3.4. The target velocity v_T^i corresponding to a pair of safe time and conflict position (t_s^i, x_c^i) is calculated as:

$$v_T^i = \frac{d^i}{t_s^i - t_{Act}} \tag{3.6}$$

where

$$d^i = x_c^i - x_{Act}$$

is the total distance the requesting CAV travels from its actuation position until reaching the conflict position x_c^i. We can rewrite d^i as $d^i = d_I^i + d_R$. d_I^i is the distance the requesting CAV travels *inside* the intersection to reach the conflict position x_c^i and d_R is the distance between the actuation position and edge of the intersection. The computation of d_I^i and d_R is straight-forward when the dimension of the intersection is known. For instance, it is trivial to show that d_I^4, the distance CAV #5 travels inside

the intersection to reach x_c^4 in the example of Figure 3.5, is equal to $acos(3/3.5) \times 3.5LW$, where LW is the lane width. Since an instant velocity change is impractical, the target velocity v_T^i needs to be modified to account for the requesting CAV's acceleration/deceleration. In the next section, we will explain in detail how the compensated target velocity $v_T^{new^i}$ is calculated based on v_T^i, taking into consideration the vehicle dynamics of the requesting CAV. After calculating all target velocities ($v_T^{new^i}$) the IM selects the slowest velocity and assigns it to the requesting CAV.

3.3 Safe Target Velocity Calculation

This section discusses how to calculate $v_T^{new^i}$ based on v_T^i, taking into consideration the vehicle dynamics of the requesting CAV. The superscription i is dropped in this section to simplify the notations. Without loss of generality, we only discuss the case where v_T is higher than v_0. The case of deceleration can be analyzed similarly.

In order to compute the compensated target velocities (v_T^{new}) to account for CAV acceleration/deceleration time, we need to know what the actual velocity and acceleration profiles of a CAV under v_T are. They can be calculated using a model for the CAV dynamics, with explicit considerations of acceleration limits and the underlying CAV controller.

In subsection 3.3.1, we first compute the velocity profile with realistic vehicle dynamics but unbounded acceleration, which will serve as a baseline for computing the compensated target velocity. Subsection 3.3.2 explains how v_T^{new} is calculated in the case that the resulting CAV acceleration/deceleration is always within the acceleration/deceleration limit. Subsection 3.3.3 discusses the case where the resulting CAV acceleration/deceleration from the baseline unbounded profile exceeds the limit.

3.3.1 *Solving for Velocity Profile with Realistic Vehicle Dynamics and Unbounded Acceleration*

The following differential equations are used to model CAV motion in the 2D space:

$$\begin{cases} \dot{x} = vcos(\phi) \\ \dot{y} = vsin(\phi) \\ \dot{\phi} = \frac{v}{L}tan(\psi) \\ \dot{v} = a \end{cases} \tag{3.7}$$

where x, y are the longitude and latitude of a CAV in Cartesian coordinates respectively, ϕ is the heading angle from the x-axis, v and a are linear velocity and acceleration of the CAV respectively, L is CAV's wheelbase distance and ψ is steering angle of front tires with respect to the heading of the CAV. In order to account for maximum acceleration and brake capability of each CAV, we consider the following saturation function to bound the acceleration. Therefore, Equation (3.7) is replaced by:

$$\dot{v} = \begin{cases} a_{max}, & \text{if } a > a_{max} \\ a_{min}, & \text{if } a < a_{min} \\ a, & \text{otherwise} \end{cases} \tag{3.8}$$

Without loss of generality, consider an eastbound CAV (driving along the x-axis before entering the intersection) that uses a PID controller to achieve and maintain the assigned target velocity. The vehicle dynamics presented in Equation (3.7) for

this vehicle can be written as:

$$\begin{cases} \dot{x} = v \\ \dot{y} = 0 \\ \dot{\phi} = 0 \\ \dot{v} = K_P e + K_I \int e + K_D \dot{e} \end{cases} \tag{3.9}$$

where e is the velocity error ($e = v_T - v$) and K_P,K_I and K_D are Proportional, Integral and Derivative gains of the PID controller. By substituting $\dot{e} = -\dot{v}$ and taking the derivative of Equation (3.9), we have:

$$\ddot{v} = -K_P \dot{v} + K_I(v_T - v) - K_D \ddot{v}$$

simplifying, we get:

$$\ddot{v} + \frac{K_P}{1 + K_D}\dot{v} + \frac{K_I}{1 + K_D}v = \frac{K_I V_T}{1 + K_D} \tag{3.10}$$

The solution to Equation (3.10) is the actual velocity profile of a CAV under the target velocity v_T.

Once the values of the PID gains are determined, we can find the homogeneous solution to Equation (3.10) by obtaining the characteristic equation of the system. To do so, we should replace $\ddot{v}$, $\dot{v}$ and v in Equation (3.10) with v^2, v and 1 respectively and set the left hand side of the equation equal to zero:

$$(1 + K_D)v^2 + K_P v + K_I = 0 \tag{3.11}$$

This equation can have real or complex roots, but we only consider the case with real roots. This is because complex roots correspond to overshoot in the controller

response and are not suitable for velocity tracking. The homogeneous solution to Equation (3.10), when roots of the characteristic equation of the system are real, can be written as:

$$v_H(t) = c_1 e^{-At} + c_2 e^{-Bt} \tag{3.12}$$

where A and B are roots of the characteristic equation (Eq. (3.11)), and c_1 and c_2 are constants to be determined (computed later). The particular solution to Equation (3.10) can be calculated by setting all derivatives to zero.

$$v_P(t) = v_T \tag{3.13}$$

Using initial conditions $\dot{v}(0) = 0$ and $v(0) = v_0$ (without loss of generality, let $t_{Act} = 0$), the complete solution can be written as:

$$v(t) = v_T + c_1 e^{At} + c_2 e^{Bt} \tag{3.14}$$

where

$$c_1 = \frac{-B}{A - B}(v_0 - v_T)$$

and

$$c_2 = \frac{A}{A - B}(v_0 - v_T)$$

When computing the velocity profile of a CAV with the Equations3.7, two cases should be considered: i) CAV acceleration/deceleration is always within the acceleration/deceleration limit, or ii) CAV acceleration/deceleration exceeds the limit and result in a saturated acceleration/deceleration behavior.

3.3.2 Case 1, no saturated acceleration

With the baseline velocity profile under a target velocity v_T calculated as in Equation (3.14), let us first consider the case where the corresponding acceleration to achieve this velocity profile is always within the acceleration limit.

In this case, the baseline velocity profile (Equation (3.14)) is the actual velocity profile; but the expected velocity profile the IM adopted when calculating v_T assumes instantaneous velocity change. The position discrepancy at the expected time of arrival t_s at a potential conflict position x_c can be calculated as the difference between the area under the expected and the actual velocity profiles:

$$e = \int_0^{t_s} \big(v_T - v(t)\big)dt \tag{3.15}$$

We can replace the upper bound of the integral with ∞, assuming that the CAV has achieved v_T at time t_s. This requires the response time of the controller is short enough (see Appendix A for a discussion on controller design to achieve a short response time) and/or the transmit line is sufficiently far away from the edge of the intersection (see Appendix A.2.2 for more details). As a result, we can rewrite the position discrepancy as:

$$\begin{aligned} e &= \int_0^{\infty} \big(v_T - (c_1 e^{At} + c_2 e^{Bt} + v_T)\big)dt \qquad (3.16)\\ &= \frac{c_1}{A} + \frac{c_2}{B} \end{aligned}$$

Equation (3.16) means the CAV will travel e units less than expected if the assigned velocity is higher than its initial velocity (or $|e|$ unit more than expected if the assigned velocity is lower than the initial velocity) during the time interval from t_{Act} to t_s. To

compensate this discrepancy, we modify the calculated velocity v_T as:

$$v_T^{new} = \frac{d+e}{t_s - t_{Act}} = v_T + \frac{e}{t_s - t_{Act}} \tag{3.17}$$

where v_T^{new} is the compensated velocity and d is the traveled distance from t_{Act} to t_s.

As a small example, we simulated a case where the IM does not compensate for CAV's actuation delay and a case where it does. Figure 3.6 shows the expected (ideal) and actual trajectories and velocity profiles. We can see that the CAV's position at

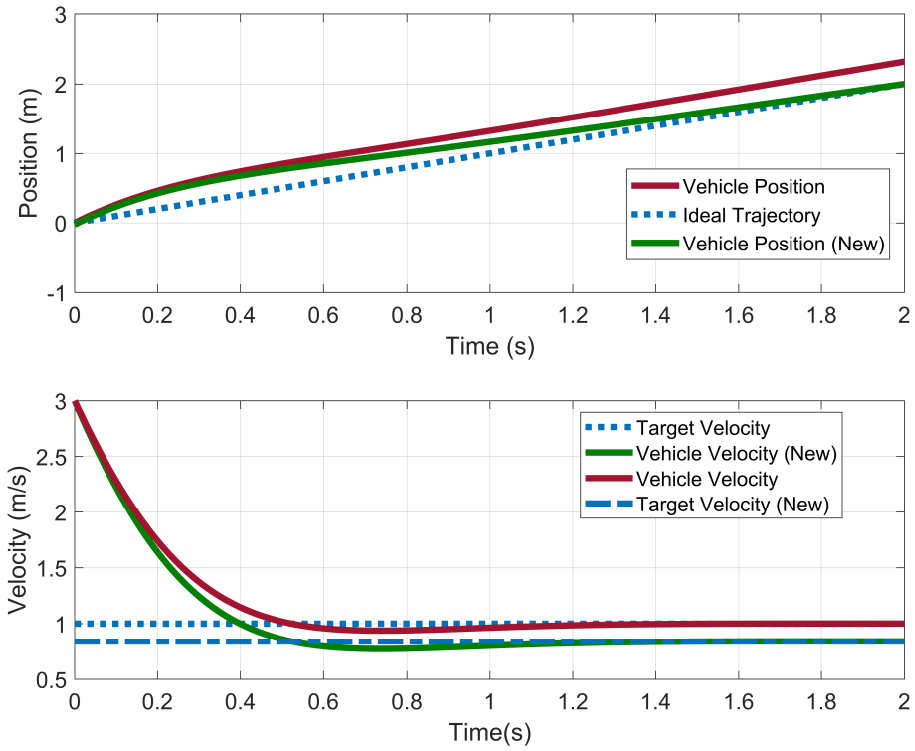


Figure 3.6: Position error of a CAV Due to Neglecting CAV Dynamics. The New Target Velocity Compensates the Effect of Response Time on the Position Error.

time $t = 2$ (when the CAV is expected to reach the conflict point) will be ahead of the expected position, if the assigned velocity is $v_T = 1m/s$, which is calculated without considering CAV dynamics. However, by modifying the assigned velocity to $v_T^{new} = 0.84m/s$, we can compensate for the position error caused by the actuation

time.

3.3.3 Case 2, with Saturated Acceleration

We now consider the case where the acceleration/deceleration required to track the baseline velocity (Equation (3.14)) exceeds the acceleration limit. The expected acceleration in order to achieve v_T (derivative of Equation (3.14)) is:

$$a(t) = Ac_1e^{At} + Bc_2e^{Bt} \tag{3.18}$$

In reality, however, the acceleration rate of a CAV is bounded (Equation (3.8)). In this case, a CAV faces saturated acceleration and its behavior will be different from what was discussed in the previous section. To determine the compensated target velocity v_T^{new}, we need to simultaneously calculate v_T^{new} and how long the vehicle will accelerate at maximum value. The position discrepancy, on the other hand, is not used in calculating v_T^{new}, unlike in the case of unsaturated acceleration. Readers interested in learning more about position discrepancy in the case of saturated acceleration are referred to Appendix A.1. Same as in the previous section, our discussion below will focus on the case where v_T is higher than v_0. The case of deceleration can be analyzed similarly.

When a CAV needs to speed up ($v_T > v_0$), and the acceleration exceeds the a_{max}, the CAV will accelerate at a_{max} and maintain it until the input from the controller is less than the limit. We refer to the time when saturated acceleration ends as t_{SAT}. Assuming the initial acceleration is large enough, Equation (3.18) when the target

velocity is v_T^{new} can be written as:

$$a'(t) = \begin{cases} a_{max}, & \text{if } t \leq t_{SAT} \\ Ac_1'e^{At} + Bc_2'e^{Bt}, & \text{if } t \geq t_{SAT} \end{cases} \tag{3.19}$$

where

$$c_1' = \frac{-B}{A-B}(v_{SAT} - v_T^{new})$$

and

$$c_2' = \frac{A}{A-B}(v_{SAT} - v_T^{new})$$

From Equation (3.19), the velocity profile can be computed as:

$$v'(t) = \begin{cases} v_0 + a_{max}t, & \text{if } t \leq t_{SAT} \\ c_1'e^{At} + c_2'e^{Bt} + v_T^{new}, & \text{if } t \geq t_{SAT} \end{cases} \tag{3.20}$$

We define v_{SAT} as the velocity at time t_{SAT} assuming the initial acceleration is zero:

$$v_{SAT} = t_{SAT}a_{max} + v_0$$

The distance traveled by the CAV can be derived from Equation (3.20) as:

$$\Delta x = \int_0^{t_{SAT}} \big(v_0 + a_{max}t\big)dt + \int_{t_{SAT}}^{t_s} c_1'e^{At} + c_2'e^{Bt} + v_T^{new}dt \tag{3.21}$$

We define $t' = t - t_{SAT}$ (note that $dt' = dt$) and replace it in the second integral as:

$$\Delta x = \int_0^{t_{SAT}} v_0 + a_{max}tdt + \int_0^{t_s - t_{SAT}} c_1'e^{At'} + c_2'e^{Bt'} + v_T^{new}dt'$$

Solving the integral assuming the response time is fast enough, we have:

$$\Delta x = v_0 t_{SAT} + \frac{a_{max} t_{SAT}^2}{2} + \frac{c_1'}{A} + \frac{c_2'}{B} + v_T^{new}(t_s - t_{SAT})$$

By setting the travelled distance equal to $v_T t_s$, we have:

$$v_T t_s = v_0 t_{SAT} + \frac{a_{max} t_{SAT}^2}{2} + \frac{c_1'}{A} + \frac{c_2'}{B} + v_T^{new}(t_s - t_{SAT}) \tag{3.22}$$

There are two unknown variables in this equation: t_{SAT} and v_T^{new}. We know that the acceleration is continuous. Therefore, from Equation (3.19), we can write:

$$a_{max} = A c_1' e^{A t_{SAT}} + B c_2' e^{B t_{SAT}} \tag{3.23}$$

The new target velocity v_T^{new} can now be determined from solving Equation (3.22) and Equation (3.23).

Figure 3.7 shows an example of the actual acceleration profile $a'(t)$ (dotted blue line) compared to the baseline $a(t)$ (red dotted curve) in the bottom sub-figure, the corresponding velocity profiles in the middle sub-figure, and the trajectories in the top sub-figure. By adopting the new target velocity (v_T^{new}), the position error due to acceleration limit will be compensated (green dashed lines).

3.4 Safety Proof

In this section, we prove that the proposed Crossroads+ methodology is safe. We start by making the following assumptions for the system:

- All CAVs are connected and autonomous (no human driver).
- All CAVs have built-in Adaptive Cruise Control (ACC) system.

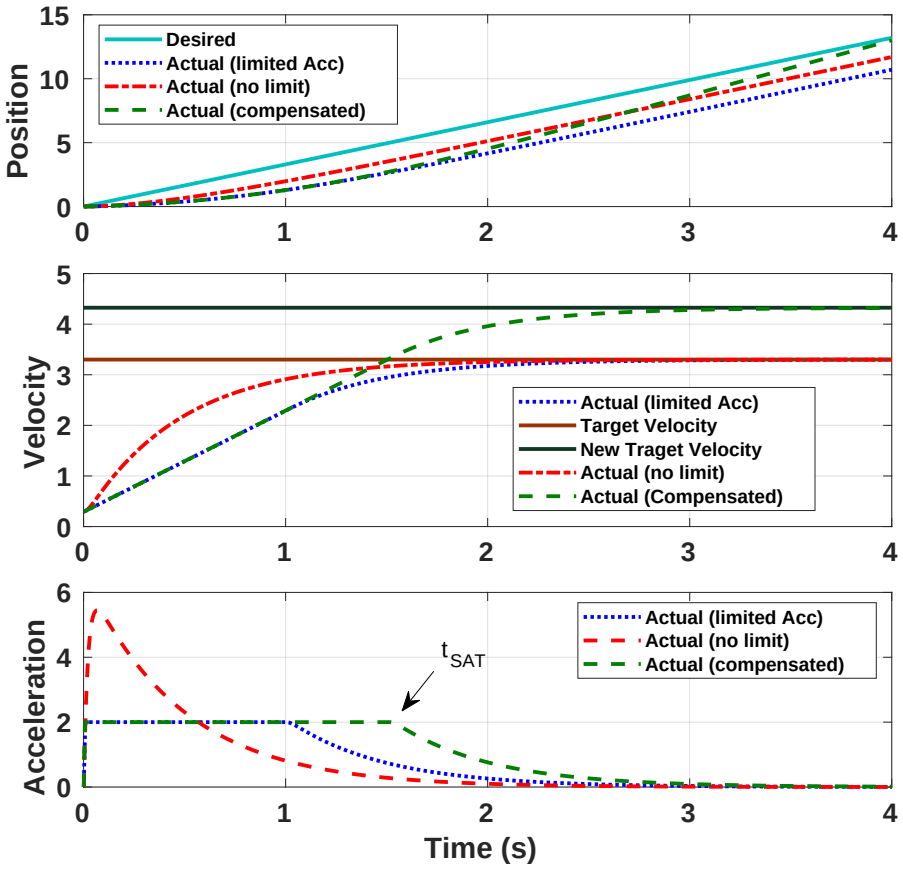


Figure 3.7: The Behavior of a Cav Using When Acceleration Is Limited and When There Is No Limit on Acceleration. Crossroads+ Is Able to Compensate for the Effect of Acceleration Delay in the Actuation of CAV.

- The spatial travel paths of all CAVs are predetermined based on the dimensions of the intersection.
- The position error due to tracking of predetermined paths is within the safety buffer.

Instead of showing that CAV trajectories don't overlap in 2D, we map their trajectories into multiple 1D ones where the longitudinal movement of the CAV matters. We assume that $x(t)$ corresponds to the center of each CAV. For a CAV of length l,

a conflict point x_c (as depicted in Figure 3.5) now becomes a conflict interval:

$$I = [x_c - l, x_c + l]$$

We use induction for the proof and define k to be the number of existing CAVs on the active list.

When $k = 0$, i.e. no existing vehicle, any velocity for the requesting vehicle will be safe.

When $k = 1$, i.e. there is one CAV on the active list, three scenarios can happen: 1) the travel paths of the two CAVs do not intersect each other; 2) the travel paths of the two CAVs intersect at a single point; and 3) the travel paths of two CAVs intersect at more than one point (paths of two CAVs on the same lane will overlap). Since the IM considers the edge of the intersection as the conflict point for two CAVs on the same lane, we can treat this case similar to the single conflict point scenario. For the first scenario, there is no conflict. For the second scenario, we assume the length of the existing CAV is l_E, its velocity is v_E, and its distance from the conflict point is d_E. The conflict time is then:

$$t_c = \frac{d_E}{v_E}$$

Recalling from Equation (3.1), the IM schedules the requesting vehicle to reach the conflict point at

$$t_s = t_c + \Delta t_{safety}$$

where

$$\Delta t_{safety} = \frac{l_{max} + l_B}{v_{min}}$$

We calculate the position of the existing vehicle at time $t_c + \frac{1}{2}\Delta t_{safety}$:

$$x_E = v_E\left(t_c + \frac{1}{2}\Delta t_{safety}\right) = v_E\left(\frac{d_E}{v_E} + \frac{l_{max} + l_B}{2v_{min}}\right)$$

or

$$x_E = d_E + (l_{max} + l_B)\frac{v_E}{2v_{min}}$$

We know that $v_{min} \leq v_E$ is always true. Therefore, we have:

$$x_E \geq d_E + \frac{l_{max} + l_B}{2}$$

which means the existing vehicle will be completely out of the conflict interval at time $t_c + \frac{1}{2}\Delta t_{safety}$ since $l_{max} \geq l_E$ and therefore,

$$x_E \notin [x_c - l_E, x_c + l_E]$$

On the other hand, assume the distance of the requesting vehicle from the conflict point is d_R. Similarly, we can calculate the position for the requesting CAV at time $t_c + \frac{1}{2}\Delta t_{safety}$ as

$$x_R = v_R\left(t_c + \frac{1}{2}\Delta t_{safety}\right) = v_R\left(t_s - \frac{1}{2}\Delta t_{safety}\right) = v_R\left(\frac{d_R}{v_R} - \frac{l_{max} + l_B}{2v_{min}}\right)$$

or

$$x_R = d_R - (l_{max} + l_B)\frac{v_R}{2v_{min}}$$

considering the fact $v_{min} \leq v_R$,

$$x_R \leq d_R - \frac{l_{max} + l_B}{2}$$

As a result, the requesting vehicle will be outside of the conflict interval:

$$x_R \notin [x_c - l_R, x_c + l_R]$$

because $l_{max} \geq l_R$.

Now, we assume that there are n CAVs on the active list that have already received a target velocity and there is no conflict among them (case $k = n$), i.e.

$$x_i \notin I_{i,j}, \forall x_i, \qquad i = 1, ..., n, \qquad j = 1, ..., n \qquad i \neq j.$$

where x_i is the longitudinal position of a CAV and $I_{i,j}$ is the conflict interval around the conflict point between CAVs i and j projected on travel path of CAV i:

$$I_{i,j} = [x_{i,j} - l_i, x_{i,j} + l_i]$$

where $x_{i,j}$ is the conflict point between CAV i and j. We will show that if a new CAV approaches the intersection and makes a request (case $k = n+1$), the assigned target velocity to the requesting CAV will be safe.

$$x_i \notin I_{i,j}, \forall x_i, \qquad i = 1, ..., n+1, \qquad j = 1, ..., n+1 \qquad i \neq j. \tag{3.24}$$

We already know that existing vehicles ($i = 1, ...n$) do not have any conflicts. As a result, we can simplify the Equation (3.24) and rewrite it as:

$$x_i \notin I_{i,n+1}, \qquad i = 1, ..., n.$$

and

$$x_{n+1} \notin I_{n+1,i}, \qquad i = 1, ..., n.$$

In other words, the new CAV (denoted by index $n+1$) should not have any conflicts with existing ones. Recalling from Section 3.2, the IM calculates target velocities with respect to each existing vehicle as:

$$v_T^i = \frac{d^i}{t_c^i + \Delta t_{safety}} \tag{3.25}$$

and assigns the slowest target velocity among all calculated target velocities to the requesting CAV:

$$v_T \leq v_T^i$$

As a result, the actual reach time of the requesting vehicle to each conflict point, t_{reach}^i will satisfy

$$\frac{d^i}{t_{reach}^i} \leq \frac{d^i}{t_s^i}$$

Since all variable are positive, we can write:

$$t_{reach}^i \geq t_s^i$$

As a result, there won't be any conflict between existing CAVs and the new one.

■

3.5 Tesbed 1 - Intersection Simulator

In order to validate the proposed method for multi-lane intersections and for arbitrary flow rates of approaching CAVs, we developed a simulator in MATLAB [1]. We simulated the IM and CAVs as separate computation nodes and communication between IM and CAVs is done using the network model. The modeled network is

[1]The simulator is available at https://github.com/mkhayatian/Traffic-Intersection-Simulator-for-Autonomous-Vehicles

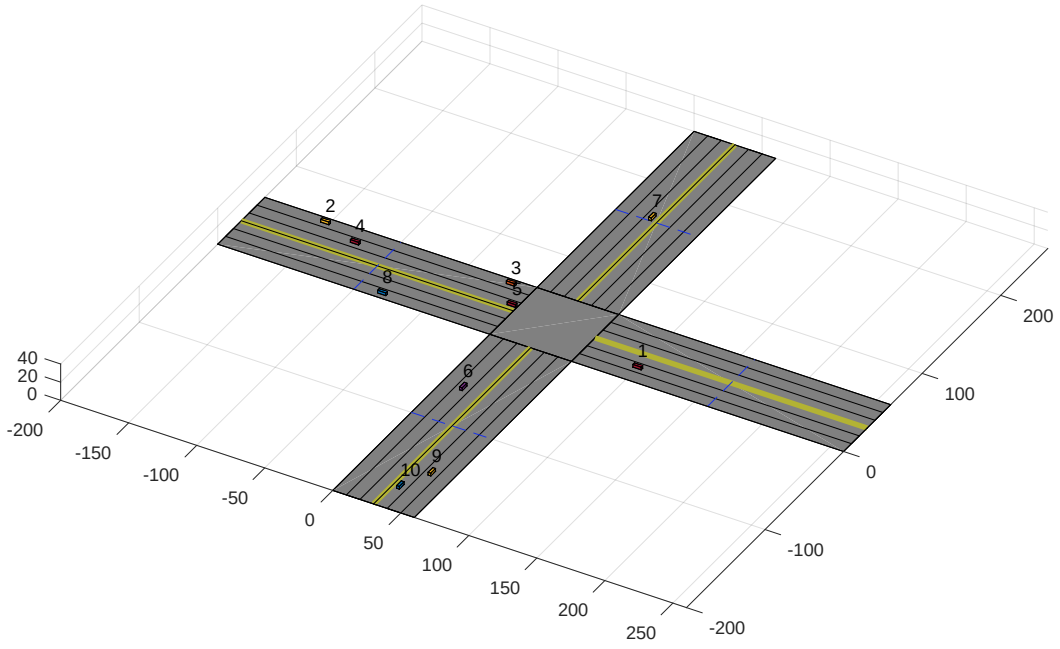


Figure 3.8: A Snapshot of Our Simulator In Matlab with Three Lane Per Road.

actually a buffer of packets with the delivery time. This allows for modeling the network delay by easily adding a random variable to the set delivery time. The random variable (D) is selected from the interval $[0, RTD_{max}/2]$. The simulated CAVs in our simulator are 6 m long and 2 m wide, and the vehicle wheelbase length is 5m. The speed limit is 50 mph ($\approx$ 22.3 m/s). The maximum value of acceleration is 5 m/s^2 and deceleration is -8 m/s^2. Roads connected to the intersection are 200 m long and lane width is 10 meters. The transmit line is 100 m away from the edge of the intersection. An overview of our simulator is depicted in Figure 3.8. In this demonstration, left turns are possible from the leftmost lane and right turns are allowed from the rightmost lane. The numbers next to each CAV in Figure 3.8 corresponds to the CAV's ID.

3.6 Results

In this section, we first show the result of experiments conducted on our testbed followed by the results of a simulated multi-lane intersection in our simulator.

3.6.1 Experiment on our testbed

We compared the throughput of VT-IM and Crossroads+. The throughput is measured in terms of the number of CAVs (8 in our case) divided by the total wait time. The wait time of a CAV is measured as the difference between the time a CAV crosses the transmit line and the time it leaves the intersection. Since existing VT-IM approaches may lead to potential collisions due to ignoring the RTD, we considered an extra buffer of B_{RTD} around each CAV to implement the VT-IM technique without accidents (see Section 3.1.2). Based on our experiments for measuring the RTD in of 1/10 scale model RC cars, Δt_{WCRTD} is $1800ms$ (Figure 3.3). Since the v_{max} is $3.5m/s$ in our testbed, the required longitudinal safety buffer for VT–IM is $630cm$, which almost 10 times of vehicle size (60cm). We tested 10 different traffic scenarios (from light to heavy traffic) using our 1/10 scale model intersection and each scenario was repeated 10 times. In the light traffic scenarios, vehicles are set to drive toward the intersection such that there is the least number of conflicts between approaching CAVs. On the other hand, for heavy traffic scenarios, CAVs are set to reach the transmit line in a short time interval. An example of a light traffic scenario and a heavy traffic scenario is depicted in Figure 3.9.

All CAVs first synchronize their clock with the IM. The IM then broadcasts the "start time" of the experiment to all CAVs. CAVs start driving when their local timer is equal to the set start time. This way, all CAVs start driving at the same time no matter where they are placed. Each CAV detects the transmit line by comparing

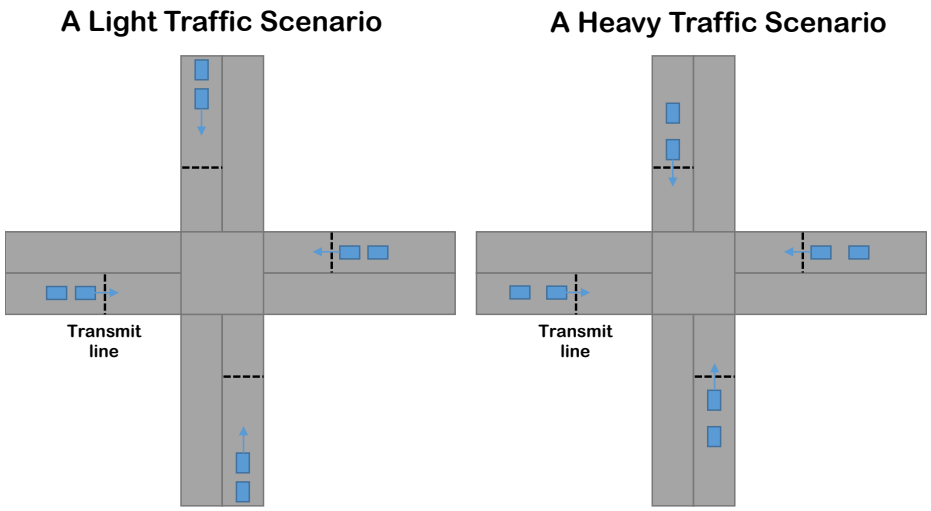


Figure 3.9: An Example of a Light Traffic Scenario and a Heavy Traffic Scenario Created by Setting the Initial Position and Velocities of CAVs.

its shaft encoder's value with the hard-coded position of the transmit line. We used the same initial position and velocity for CAVs in both VT-IM and Crossroads+ experiments. The improvement in the average wait time of CAVs in Crossroads+ is depicted in Figure 3.10. The improvement is calculated by dividing the average wait time for VT-IM by the average wait time for Crossroads+. The first scenario represents a heavy traffic case where CAVs arrive at the transmit line with short time headways. Conversely, scenario 10 represents a light traffic scenario where CAVs are spaced enough to arrive at the transmit line with large time headways. Other scenarios are generated randomly and are sorted based on the reduction in the wait time of CAVs that is achieved in our method compared to VT–IM. Since the size of the intersection and the location of the transmit line is fixed for our experiment, less wait times result in higher throughputs. Based on our results, Crossroads+ can achieve 15% better throughput on average in comparison with VT-IM. The main advantage of Crossroads+ is avoiding consideration of an extra safety buffer due to the unknown RTD.

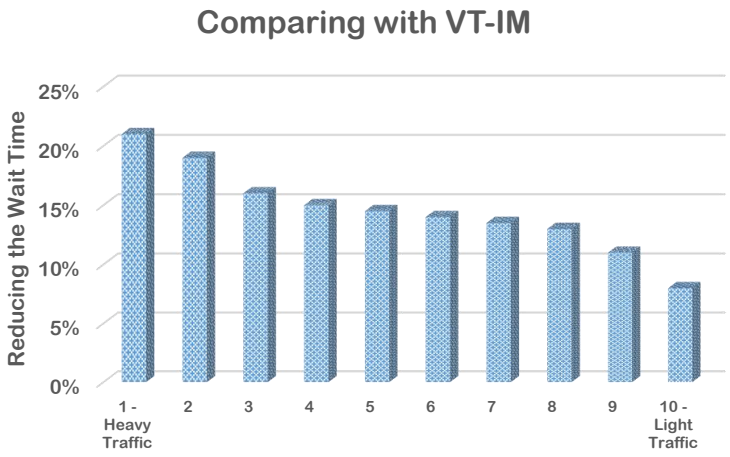


Figure 3.10: Improvement in the Wait Time for 10 Different Scenarios from Heavy to Light Traffic Conducted Using Our 1/10 Scale Model CAVs (Explained in Testbed Section). Scenario 1 Represents the Heaviest Traffic Case and Scenario 10 Represents the Lightest One.

3.6.2 Simulation of Multi-Lane Intersections

We compared QB-IM, VT-IM, and Crossroads+, in terms of both network overhead and throughput in our simulator. Since the VT-IM and QB-IM are not proposed to account for RTD, we added an extra safety buffer (Equation 3.5) for each CAV to operate the intersection without any accidents. The size of this buffer is determined from the multiplication of maximum velocity and WCRTD for a realistic case (v_{max} = 22.3 m/s or 50 mph). The maximum latency of DSRC (Dedicated Short-range Communication) is 100 ms Xu *et al.* (2017). The WCET (Worst-case Execution time) of the FCFS scheduling algorithm is 900 ms. Therefore, WCRTD is 0.1 + 0.9 + 0.1 = 1.1 s and the required safety buffer is 1.1 * 22.3 = 24.53 m, which is almost four times of the length of a vehicle. To provide a fair comparison, we conducted a simulation of all approaches with the same configuration. In particular, we used the same time vector and an initial velocity vector for generating CAVs. In terms of network overhead, VT-IM, and Crossroads+ have the same performance since the data

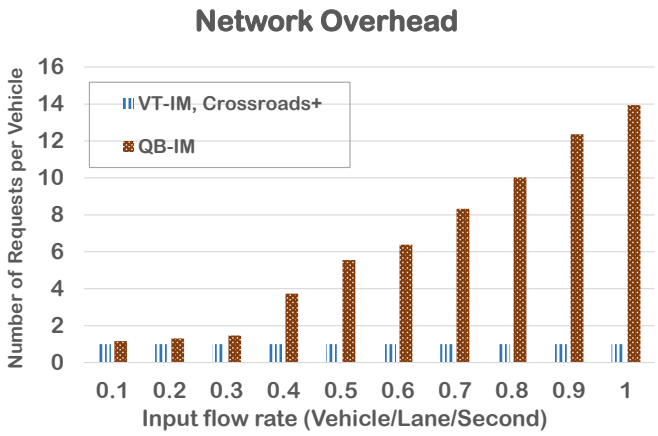


Figure 3.11: Comparison of the Network Overhead of QB-IM Approach with VT-IM and Crossroads+ in Terms of the Average Number of Messages Exchanged Between CAVs and IM.

is exchanged once unless a packet is dropped. However, due to the trial-and-error nature of the QB-IM approach, its communication overhead for heavy traffic cases is very high. We considered a 1-second timeout between two consecutive requests for the QB-IM method and counted the number of requests from CAVs. Figure 3.11 shows the average number of requests per CAV for different input flow rates of CAVs. As the input flow rate of the intersection increase in QB-IM, more CAVs fail to get a reservation, which results in a re-request and more network traffic. Based on the results, QB-IM has 14X communication overhead in the worst case.

We measured the average travel time of CAVs for different input flow rates. Due to the limitation of VT-IM, it cannot operate for flow rates more than 0.02 CAV/lane/Second. This is because the intersection becomes congested and IM cannot assign a positive velocity to a CAV. Figure 3.12 depicts the average travel time of the ideal case, QB-IM, VT-IM, and Crossroads+ (CAV per lane per second). The ideal case is indicated by a blue solid line, which represents an intersection with separated roads so that CAV can always pass the intersection while driving at the max velocity. VT-IM performs better than QB-IM because it has the advantage of assign-

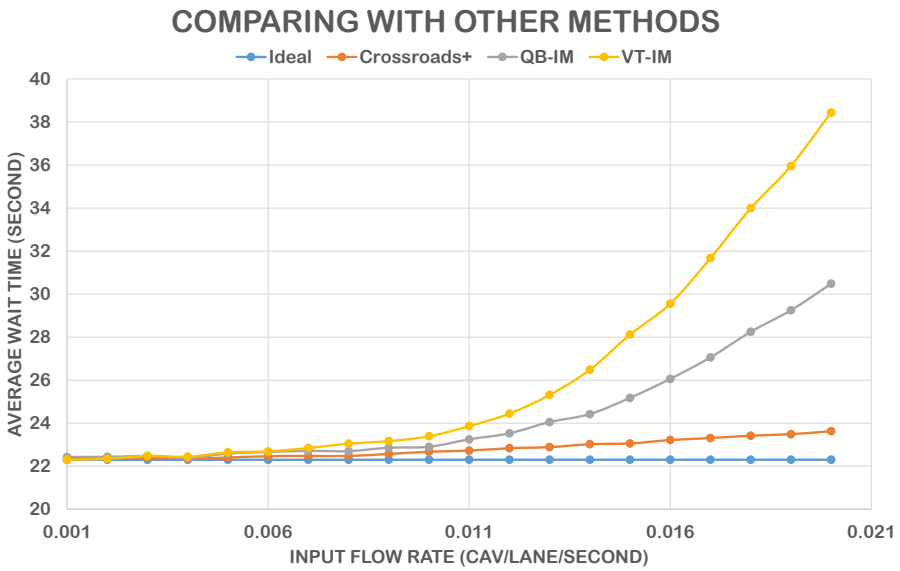


Figure 3.12: Comparing Crossroads+ with Other Techniques Implemented in Our Simulator. Crossroads+ Performs Better than Other Approaches Especially in Heavy Traffic Scenarios.

ing higher velocities if possible and can avoid the extra safety buffer. On average, Crossroads+ achieves 36% better throughputs in comparison with VT-IM and 16% in comparison with QB-IM.

In Figure 3.13, we depicted the position and velocity of a CAV for VT-IM, QB-IM and Crossroads+. In VT-IM, the CAV starts tracking the target velocity ($v_T = 6$) as soon as it's received. In QB-IM, the CAV comes to a complete stop before entering the intersection (at x = 200 m) and then accelerate after 3 second. In Crossroads+, the CAV waits for WCRTD = 1.1 second and then starts tracking the target velocity. The target velocity for the Crossroads+ is most likely greater than VT-IM since VT-IM requires an extra safety buffer due to ignoring the RTD.

We also developed an experiment to observe the effect of the safety buffer size, speed limit, WCRTD and distance of transmit line from the edge of the intersection on the throughput of an intersection. In the first experiment, the average travel

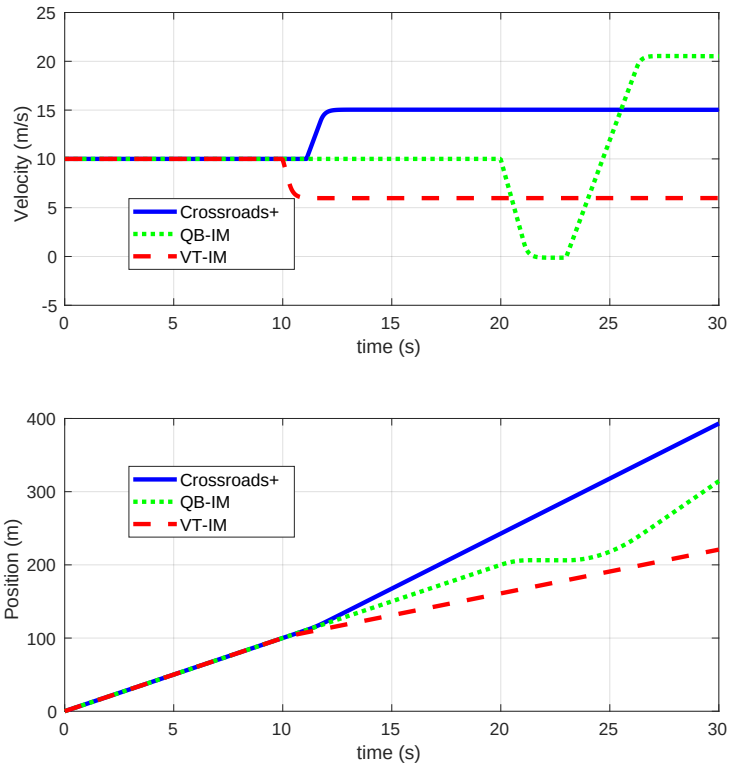


Figure 3.13: Comparing Behavior of a CAV for VT-IM, QB-IM and Crossroads+ Approaches.

time is measured for different value of WCRTD. In the original case, the WCRTD is 1100 ms and in the rest of the cases, it's decreased by 90%, 89%, 70%, 60%, and 50%. We can observe that a smaller WCRTD can slightly increase the throughput and this is because a CAV travels at its initial velocity for a shorter time. In the second experiment, we measured the average travel time of the CAVs for the base case (Vmax = 50 mph) and a reduction in the speed limit (45, 40, 35, 30 and 25 mph). A lower speed limit results in a significant increase in the average travel time of CAVs. This is because with a slower assigned velocity, a CAV occupies the intersection area for a longer time and therefore, delays the scheduling of next CAVs. In the third experiment, the distance of transmit line from the edge of the intersection varies between the original length (100 m) and a 25% increase in the length (125m). It can

be observed that the average wait time is reduced when the transmit line is placed farther from the intersection. The reason behind this reduction is better flexibility that is achieved. When the distance of the transmit line increases, the IM can assign a target velocity to a CAV earlier. In the last experiment, we measured the average travel time of the CAVs when the safety buffer is increased by 10%, 20%, 30%, 40% and 50%. Figure 3.14 shows the sensitivity of throughput on the WCRTD, speed limit, transmit line and safety buffer.

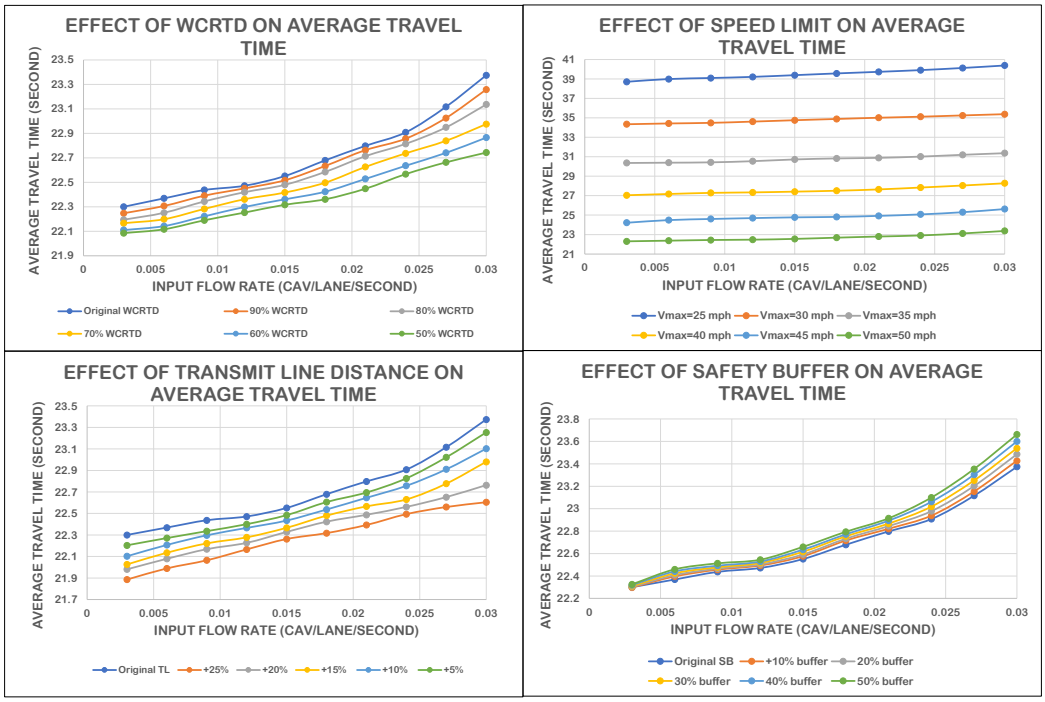


Figure 3.14: The Effect of WCRTD, Speed Limit, Transmit Line and Safety Buffer on the Throughput of the Intersection.

We can observe that by increasing the speed limit, we can improve the throughput of the intersection for all input flow rate. However, the increase in the safety buffer only affects the higher input flow rates.

Chapter 4

ROBUST INTERSECTION MANAGEMENT (RIM) APPROACH

4.1 Background

Velocity Assignment Intersection Management (VA-IM) techniques neglect the effect of network delay due to communication and the delay caused by the IM when processing the request. As a result, a vehicle will receive the target velocity with an unknown delay and the time it starts accelerating/decelerating is delayed. We refer to the summation of WCND in the forward and backward path and, WCET of the IM as Worst-Case Round-trip Delay (WCRTD). Since the value of WCRTD is not known beforehand, it cannot be compensated by the IM. Hence, the eventual position of the vehicle (and therefore its TOA) is erroneous. Figure 4.1 shows a scenario where the assigned velocity is greater than the current velocity of the vehicle and network delay and IM processing time cause a late actuation. As a result, IM should consider an extra safety buffer (the brown buffer in Figure 4.1) for each vehicle to ensure their safety. The result buffer has the same lateral size as the safety buffer while the longitudinal size is extended relative to the WCRTD and maximum velocity of the vehicle. The longitudinal size of the extra safety buffer can be as large as 3X of a vehicle size Andert *et al.* (2017). Considering such a large buffer greatly reduces the throughput of the intersection and makes such VA-IM techniques impractical to use. In 2017, Andert *et al.* proposed a velocity assignment approach, Crossroads Andert *et al.* (2017), that can skip considering an extra buffer due to WCRTD.

In this technique, all vehicles first synchronize their local clock with the IM and, then send their information (position, velocity, etc.) with a timestamp to the IM. As

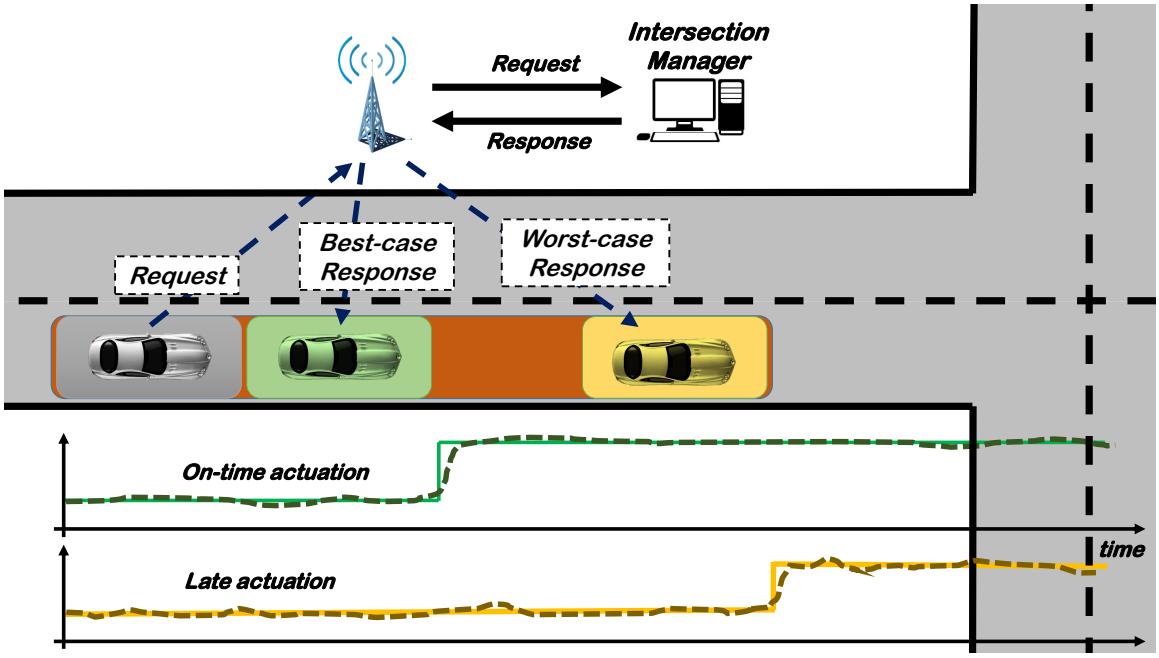


Figure 4.1: The Effect of Network Delay and IM Processing Time on the Behavior of a Vehicle in VA-IM Techniques. All VA-IM Techniques Require Considering an Extra Safety Buffer (the Brown Box) to Ensure the Safety of Vehicles.

a result, Both IM and the requesting vehicle have the same notion of time. Accordingly, IM calculates an actuation timestamp along with the target velocity to fix the actuation time of the vehicle (When to start accelerating/decelerating). Crossroads can achieve 1.6X better throughput in comparison with regular VA-IM approaches thanks to avoiding considering an extra safety buffer.

Despite the fact that vehicles have a deterministic behavior in Crossroads technique, our experiment on our 1/10 scale intersection with autonomous vehicles (explained in the Result section) shows that accidents can happen. This is because IM needs an accurate model of a vehicle in order to compute a safe target velocity for the vehicle and any mismatches between the actual model and the considered one by the IM can cause errors in the expected TOA of the vehicle and cause accidents. Specifically, the response time of a vehicle (acceleration or deceleration time) can be different from the expected one, which depends on the dynamics of the vehicle as well

as the control algorithm. IM should know how long acceleration/deceleration time of a vehicle is and how long it takes to maintain the assigned velocity. Moreover, vehicles track a constant velocity in VA-IM and Crossroads approaches and not a position trajectory. Thus, external disturbances like wind, slope, etc. can temporarily prevent them from tracking the assigned velocity and therefore, the eventual arrival time of the vehicle varies and can cause accidents. In order to demonstrate this issue, we performed an experiment on our 1/10 scale model autonomous vehicle (Result Section) and measured the position and velocity of a vehicle in presence of an external disturbance (a step input with 0.1X amplitude was added to the controller's input to the motor) and model mismatches (up to 10% in parameters of the PID controller and actuator gain). Figure 4.2 shows the position and velocity trajectories of the vehicle and the effect of model mismatches and the external disturbance on the eventual position of the vehicle. One can observe that the vehicle has a 0.54 m position error when it enters the intersection, which is almost equal to the size of the vehicle (vehicle length is 0.6m). This experiment shows that VA-IM approaches including Crossroads should consider an extra safety buffer due to the model mismatch and possible disturbances to ensure the safety of vehicles.

Another practical issue of VA-IM approaches and Crossroad is associated with the speed limit for vehicles that intend to make a turn at the intersection. A vehicle should not make the turn at high velocities to avoid rollover. Since in Crossroad IM assigns a constant velocity to all vehicles to maintain (including those that intend to make a turn) and vehicles are supposed to keep the assigned velocity until entering the intersection, the assigned velocity for vehicles that intend to make a turn will be bounded by the turn speed limit. As a result, turning vehicles have to drive at a slow speed before reaching the intersection and this reduces the throughput of the intersection.

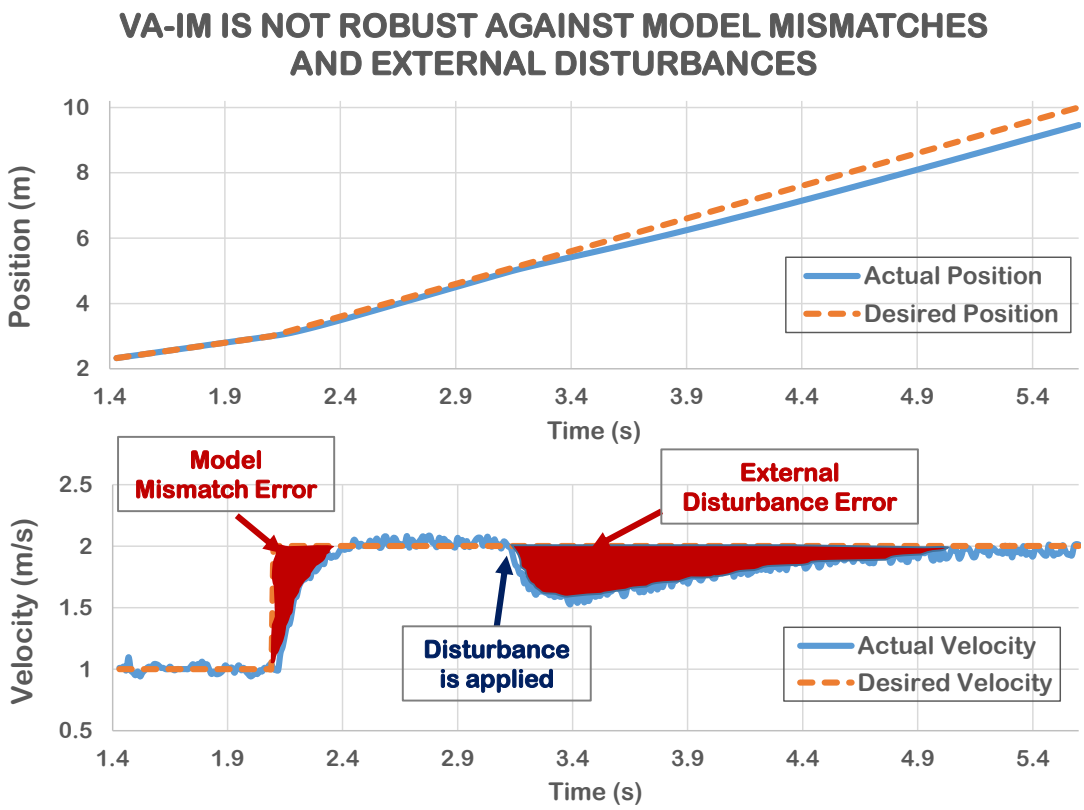


Figure 4.2: All VA-IM Techniques and Even Crossroads Are Vulnerable to Model Mismatch and External Disturbances. The Top Figure Shows the Actual and Expected Position of the Vehicle and the Bottom One Shows the Actual and Expected Velocity of the Vehicle in Presence of Model Mismatch and an External Disturbance.

To address this issue, we propose the RIM approach which solves the problem for bounded errors on the model and external disturbances. Figure 4.3 shows an overview of existing interfaces for intersection management of CAVs. In RIM approach, we divide the status of an approaching vehicle into four phases: 1) when the vehicle is within the range of the intersection and before reaching the synchronization line, 2) after the synchronization line and before the transmit line and 3) after sending the request and before receiving the response, 4) after receiving the response until entering the intersection. Figure 4.4 shows the status of a vehicle at different phases.

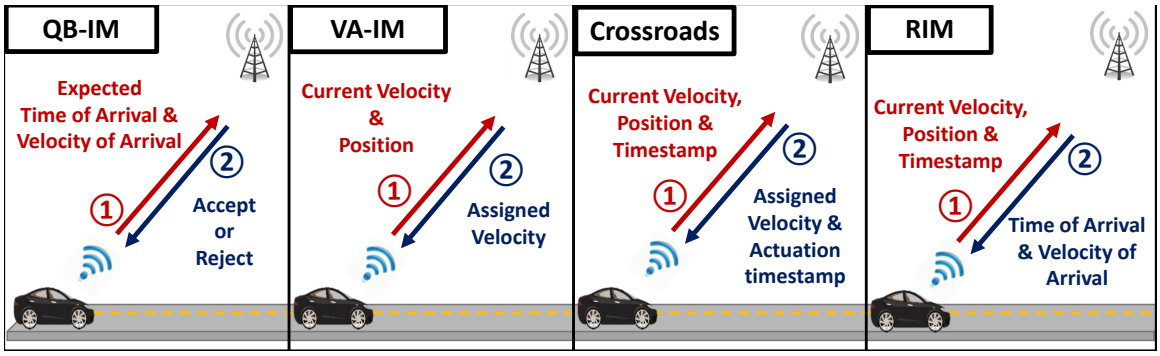


Figure 4.3: In QB-IM Techniques, Approaching Vehicles Propose a Time-space Slot in the Intersection, and the IM Replies with a Yes or No. In VA-IM Methods, Vehicles Report Their Position, Velocity, and Timestamp When They Arrive, and the IM Assigns Them a Velocity (Hence Velocity Assignment) at Which to Drive. In Crossroads and Crossroads+ Approaches, a Fix Actuation Timestamp Is Assigned to a Vehicle to Make It Robust Against Network Delay Variations. In the Proposed Approach RIM, the IM Assigns a Velocity of Arrival (VOA) and Time of Arrival (TOA) to Each Vehicle.

4.2 Robust Intersection Management Approach

In phase 1, all vehicles synchronize their local clock by either communicating with the IM or receiving a reference clock from a GPS sensor (GPS satellites broadcast very accurate clocks). If the synchronization is successful, the vehicle enters phase 2 and sends its position (P), velocity (V), acceleration (a) and the corresponding timestamp (TS), as well as the outgoing lane (LO), max/min acceleration (a_{max} and a_{min}) and the ID to the IM upon crossing the transmit line. In phase 3, IM processes the request and calculates a feasible TOA and VOA, based on the status of the vehicle (V-Info) and the scheduling policy (FCFS, BATCHTachet *et al.* (2016), etc.). Variety of scheduling policies are studied in the literature Tachet *et al.* (2016); Fayazi *et al.* (2017); Ahn and Del Vecchio (2016). Since the effectiveness of the scheduling policy is not the main focus of this paper, we use an FCFS scheduling policy for simplicity. Then, IM sends them back to the requesting vehicle. In this phase, the vehicle maintains its initial velocity until it receives the response. In phase 4, the

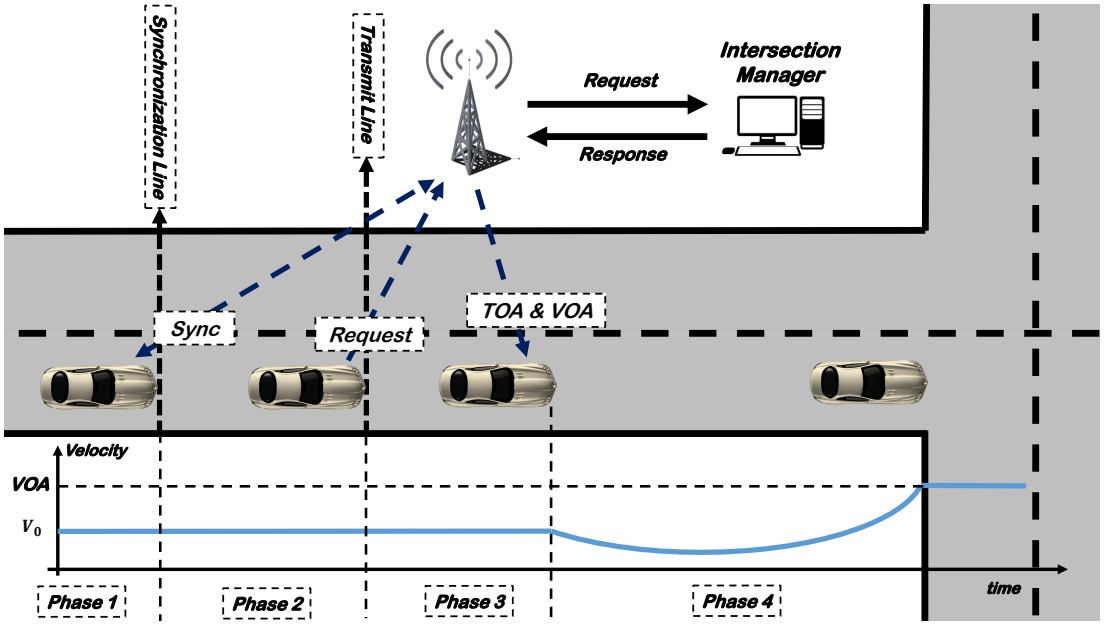


Figure 4.4: Different Phases of a Vehicle in Our Technique (RIM). Phase 1) Synchronization, Phase 2) Sending a Request, Phase 3) Receiving the Response, Phase 4) Trajectory Calculation and Tracking

vehicle creates a reference trajectory and follows it until it enters the intersection. We consider the following model for the behavior of vehicles in 2D:

$$\begin{cases} \dot{x} = vcos(\phi) \\ \dot{y} = vsin(\phi) \\ \dot{\phi} = \frac{v}{L}tan(\psi) \\ \dot{v} = u(t) \\ u(t) = K_a\big(- K_p e - K_i \int e - K_d \dot{e} + d(t)\big) \end{cases} \tag{4.1}$$

where x, y are longitude and latitude of the vehicle in Cartesian coordinates respectively, ϕ is the heading angle of the vehicle from the x-axis, v is the linear velocity of the vehicle, L is vehicle's wheelbase distance, ψ is the steering angle of front tires and u is the control input for the motor. K_p, K_i and K_d are PID controller gains,

e, $\int e$ and $\dot{e}$ are the error between actual velocity and target velocity, its integral and derivative respectively and $d(t)$ is the applied disturbance. K_a is a constant to model actuator's gain. The input for the motor is $u(t)$, which is generated as a Pulse Modulation Width (PWM) signal. We assume that the values of the PID controller and the actuator gain have model mismatches.

4.2.1 Vehicles

When the vehicle receives the TOA and VOA, it computes an optimal reference position trajectory and a PID controller is utilized to track the trajectory. Each vehicle has a specified timeout to bound its waiting time when waiting the response. Algorithm 3, shows a pseudocode of the vehicle's controller. The value of d_{min} is calculated based on a_{min} and v_{max}, i.e., the distance a vehicle needs for stopping. In order to compute the reference trajectory, each vehicle stores its current position, velocity, and the timestamp as initial position (x_0), velocity (v_0) and time (t_0). Additionally, final position (x_f), velocity (v_f) and TOA (t_f) of the reference trajectory are known (received from the IM). Any position trajectory that satisfies the initial and final position condition ($x(t_0) = x_0$ and $x(t_f) = x_f$) and its derivative (velocity trajectory) satisfies the initial and final velocity conditions ($v(t_0) = v_0$ and $v(t_f) = v_f$) can be a candidate for the reference trajectory. However, we are looking for an optimal trajectory for the vehicle. So, we define a functional J to minimize the acceleration of the trajectory:

$$J = \int_{t_0}^{t_f} a^2 dt \tag{4.2}$$

where a is the acceleration of a vehicle. After solving Equation (4.2) using the *Fundamental Lemma of the Calculus Variation* Gelfand *et al.* (2000), the solution (accel-

Algorithm 3: Vehicle Controller

```
1  if Sync line is crossed then
2  |  result = synchronize();
3  |  if result is not OK then
4  |  |  if distance to transmit line is less than d_min then
5  |  |  |  update(Trajectory, SD);                    /* slow down */
6  |  |  end
7  |  |  Goto Line 3;
8  |  end
9  end
10 if Transmit line is crossed then
11 |  V-Info = [P, V, a, TS, LO, a_max, a_min, ID];
12 |  send(V-Info);
13 |  Wait for the response;
14 |  if response is timed out then
15 |  |  if distance to intersection is less than d_min then
16 |  |  |  update(Trajectory, SD);                    /* slow down */
17 |  |  end
18 |  |  Goto line 12;
19 |  else
20 |  |  [TOA, VOA] = getPacket(response) ;
21 |  |  [A_0, B_0] = calculateTrajectory(TOA, VOA);
22 |  |  update(Trajectory, [A_0, B_0]);        /* set the Ref Trajectory */
23 |  end
24 end
```

eration trajectory) is found to be in the form of:

$$a(t) = A_0 t + B_0 \tag{4.3}$$

A_0 and B_0 are constant variables to be determined. Taking integral from (4.3), we have:

$$v(t) = \frac{1}{2} A_0 t^2 + B_0 t + v_0 \tag{4.4}$$

Taking integral from (4.4) results in a cubic function as:

$$x(t) = \frac{1}{6}A_0t^3 + \frac{1}{2}B_0t^2 + v_0t + x_0 \tag{4.5}$$

Without loss of generality, we assume that the initial time t_0 for the reference trajectory is zero. By substituting t, $x(t)$ and $v(t)$ for boundary condition values, t_f, x_f and v_f in Equations (4.4) and (4.5), following equations are derived:

$$x_f = \frac{1}{6}A_0t_f^3 + \frac{1}{2}B_0t_f^2 + v_0t_f + x_0 \tag{4.6}$$

and

$$v_f = \frac{1}{2}A_0t_f^2 + B_0t_f + v_0 \tag{4.7}$$

Solving Equations (4.6) and (4.7) for A_0 and B_0, yields:

$$\begin{cases} A_0 = \frac{6(2x_0-2x_f+t_fv_0+t_fv_f)}{t_f^3} \\ B_0 = \frac{-2(3x_0-3x_f+2t_fv_0+t_fv_f)}{t_f^2} \end{cases} \tag{4.8}$$

Each vehicle computes the value of A_0 and B_0 and creates its reference trajectory according to Equation (4.5). If a vehicle receives the target TOA and VOA within the worst-case delay (due to the IM's computation time and network delay), it's still able to create a feasible trajectory that meets the final conditions (TOA and VOA).

Case study

To have a better understanding, we simulated position and velocity trajectories of a vehicle (Using Equation (4.1)) that is 15 m away from the intersection while driving at 3 m/s. The worst-case delay from IM to the vehicle is 1350 ms and the assigned

Algorithm 4: IM's Scheduling algorithm

```
1  Input: Request;
2  Outputs: [TOA, VOA];
3  while Request buffer is not empty do
4  |   V-Info = read(buffer[first]);
5  |   [TOA, VOA] = Schedule(V-Info, I-Info);
6  |   Result = F-Check(TOA, VOA, V-Info, I-Info);
7  |   if Result is OK then
8  |   |   Send(TOA,VOA,Vehicle Info);
9  |   |   update(I-Info)
10 |   else
11 |   |   Increase(TOA);
12 |   |   Goto Line 6;
13 |   end
14 end
```

TOA and VOA are 4 s and 2.5 m/s respectively. Dashed lines in Figure 4.5 show position and velocity trajectories for the best-case round-trip delay (BCRTD) and solid lines depict position and velocity trajectories for the worst-case round-trip delay (WCRTD) respectively. Delay in the network or IM processing time may affect the trajectory of the vehicle. However, no matter how much is the delay, as long as it's smaller than the WCET plus WCND, the arrival time and velocity of the vehicle remains unaffected.

4.2.2 Intersection Manager

When IM receives a request, it computes a TOA and VOA based on the status of the requesting vehicle (V-Info) and the status of other vehicles that have already received a TOA and VOA (I-Info). Before sending back the computed TOA and VOA to the requesting vehicle, IM verifies the feasibility of the computed TOA and VOA using the "F-Check" function. Algorithm 4 shows the pseudo-code for the IM. In order to check the feasibility of assigned TOA and VOA, IM has to check

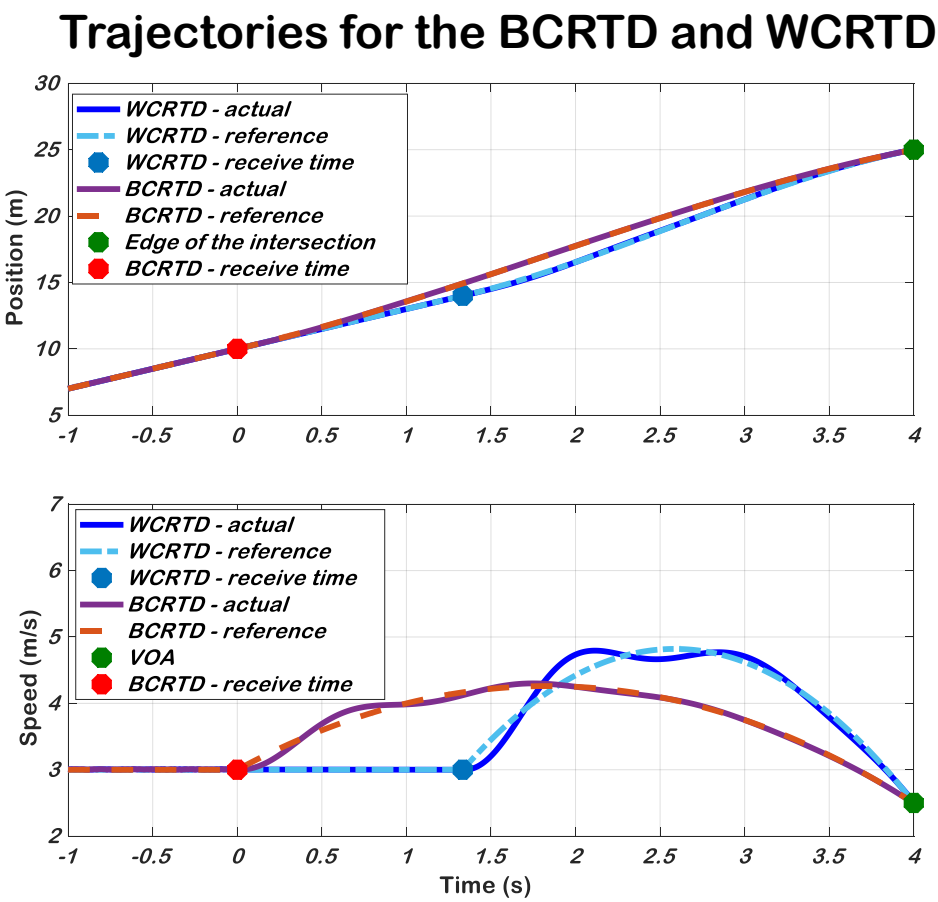


Figure 4.5: Velocity and Position Trajectories for the Best-case and Worst-case Round-trip Delay (BCRTD and WCRTD) in the Network.

the future trajectory of the vehicle and verify that road specifications ($V < V_{max}$), vehicle specifications ($a < a_{max}$) and safety specifications (No front-back accident before entering the intersection) are not violated. From Figure 4.5, one can observe that the area under the velocity profile is the same for both best-case and worst-case RTD. This is because the TOA and VOA are fixed. As a result, the vehicle will experience higher/lower velocities (a higher peak/a lower trough), as the receive time increases. Based on this observation, we can conclude that if the worst-case trajectory does not violate the maximum/minimum velocity threshold, the best-case trajectory

never exceeds such values. This way, we can check if requirements are being met only by verifying the worst-case trajectory.

4.2.3 Safety Analysis

IM needs to verify that the assigned TOA and VOA are safe. As a result, it performs a feasibility analysis for the best-case and worst-case scenarios. F-Check function in Algorithm (2) computes the values of A_0 and B_0 based on the WCND and WCET and, checks if the max value of the worst-case delay trajectory is smaller than road speed limit (V_{max}) and the min value is greater than a threshold $V_{min} > 0$. Additionally, F-Check verifies if the maximum acceleration of the worst-case trajectory is smaller than a_{max}. For different values of VOA and TOA, we simulated the position and velocity trajectories of a vehicle and depicted them in Figure 4.6 where green trajectories are feasible and red ones are infeasible. Algorithm 5 shows details of the F-Check function. Since the extreme acceleration/deceleration cases occur only at boundary conditions, IM can verify the feasibility of the worst-case reference trajectories by just checking the acceleration at the initial time. We simulated the behavior of a vehicle driving at 3 m/s for different pairs of VOA and TOA when the intersection is 15 meters away. Figure 4.6 depicts the position and velocity trajectories of the vehicle. If the velocity trajectory for the WCRTD scenario exceeds the speed limit or its slope exceeds the acceleration limit (a_max), the trajectory is not feasible and IM extends the TOA of the vehicle. However, if the velocity trajectory goes under the minimum velocity, it means that the vehicle should drive at a very slow velocity, which is not practical. Once the vehicle calculates the values of A_0 and B_0, it sends them to the IM in order to confirm that it has received the assigned TOA and VOA and, lets the IM know how the trajectory would be.

It's also possible that the trajectory of a vehicle conflicts with another vehicle

Algorithm 5: F-Check function

```
1  v = calculateVelocity ;                          /* based on Eq.(4) */
2  a = calculateAcceleration;                       /* based on Eq.(3) */
3  inLane = getLane(V-Info);
4  if max(v) < v_max and min(v) < v_min then
5  |  if max(a) < a_max then
6  |  |  For all cars ∈ I-Info s.t. V-Info.inLane = inLane
7  |  |  distance = distanceBetweenCar1andCar2;
8  |  |  if min(distance) > distance Threshold then
9  |  |  |  Result = OK;
10 |  |  else
11 |  |  |  Result = Not OK;
12 |  |  end
13 |  else
14 |  |  Result = Not OK;
15 |  end
16 else
17 |  Result = not OK;
18 end
```

in the same lane before reaching the intersection. We simulated a case where two vehicles driving in the same lane have a conflict on their position trajectory and depicted their trajectories in Figure 4.7. Blue trajectories belong to the front vehicle and the red and green ones belong to the rear car. Red trajectories are not feasible while the green ones are feasible. IM can find a feasible trajectory for the rear vehicle by increasing the TOA. If the distance between trajectories of two vehicles in the same lane is always greater than a threshold, the value of the result is "OK". Otherwise, the result will be "not OK" and the IM has to increase the TOA and verify the TOA and VOA using the F-Check function again.

4.2.4 Practical issues

Since vehicles and IM interact with each other, both should follow some rule as a prerequisite to the correct functionality of the system. For instance, the system will

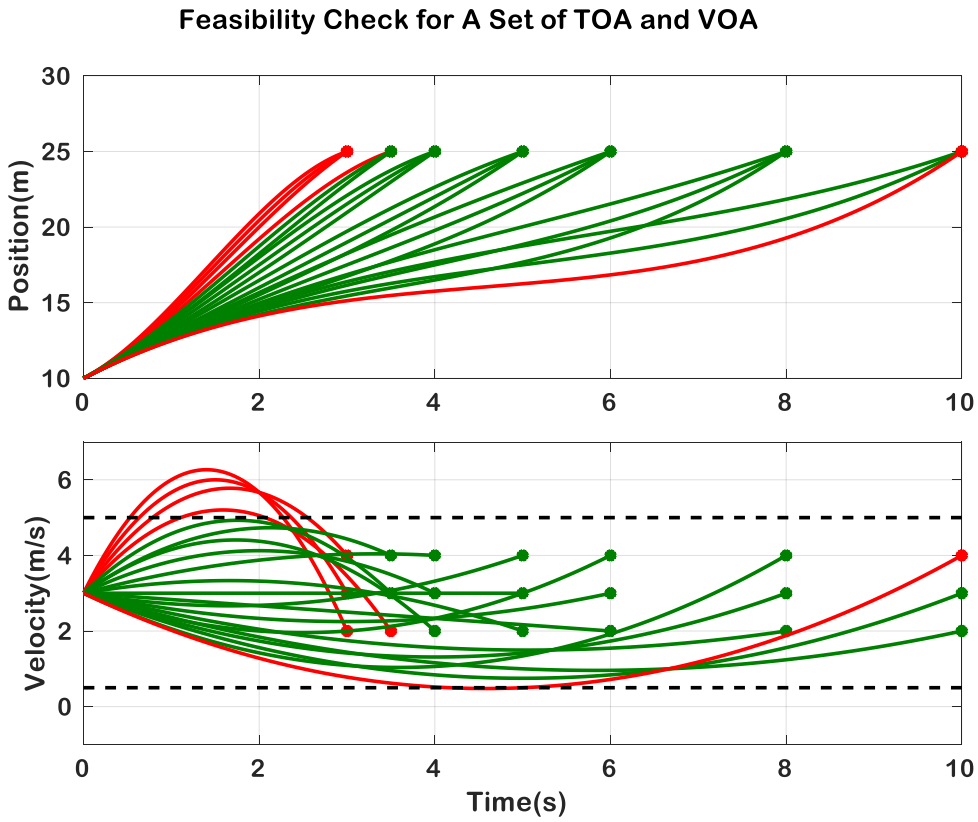


Figure 4.6: An Example of Feasibility Checking for a Set of the VOA and TOA. Based on the Specified Maximum and Minimum Velocity Thresholds, IM Rejects a Pair of TOA and VOA. Green Points on the Velocity Figure Correspond to Feasible TOAs and VOAs.

not work if the processing time of the IM is very high or if a vehicle takes a trajectory that fails to satisfy the assigned TOA and VOA. Therefore, we discuss some of the necessary requirements that should be met. It is challenging to find an upper bound for the network request because the delay in the network can be infinite. To address this issue, vehicles use a timeout mechanism to bound the waiting time of a vehicle. This ensures that a vehicle either receives the response within the expected delay or it will ignore the response if it's received afterward. The value of the timeout can be determined by measuring the average delay of the network and WCET of

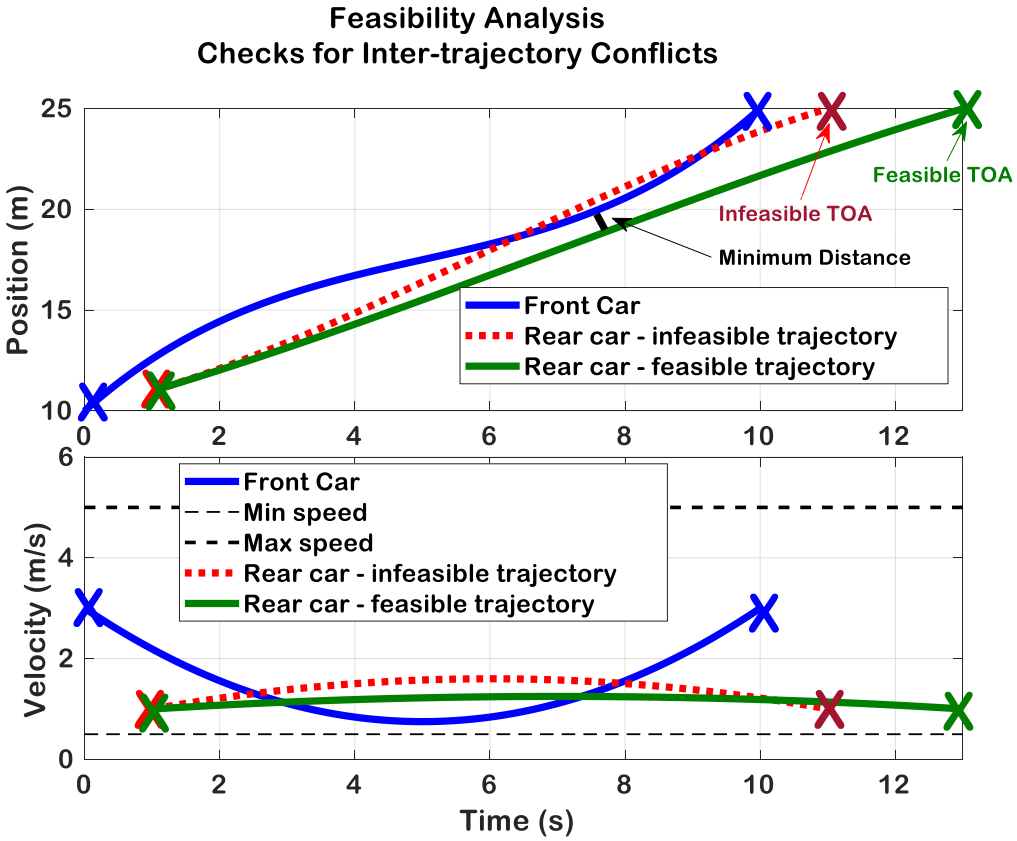


Figure 4.7: A Scenario Where F-check Fails. The Assigned TOA and VOA Cause a Front-back Accident (the Blue Position Trajectory Crosses the Red One). IM Then Assigns Another TOA (Green) That Is Safe.

the IM. WCET can be calculated statically using existing WCET analysis methods Hardy and Puaut (2008); Chattopadhyay and Roychoudhury (2009); Gustafsson *et al.* (2006). Similarly, if a vehicle fails to synchronize its clock with the IM or cannot get it from the GPS before reaching the transmit line, it should slow down and stop behind the intersection line.

As another requirement, vehicles must always retain a safe distance from their front vehicle. Typically, the Adaptive Cruise Control (ACC) system is responsible to maintain a safe distance from the front vehicle by adjusting the velocity. Based on the Responsibility-Sensitive Safety (RSS) model Shalev-Shwartz *et al.* (2017), maintain-

ing a minimum distance from the front vehicle requires having a bounded response time (from sensing to actuation). In order to guarantee the safety of the intersection, we can express a set of requirements for vehicles and IM. One way to formally express such safety requirements for each processing unit is specifying them using temporal logic (like Timestamp Temporal Logic (TTL) Mehrabian *et al.* (2017, 2018)). Here's the list of requirement:

- WCET of the IM when responding to a request should be less than a threshold, say t_{IM}.
- Settling time of the PID controller should be short enough (Settling time is referred to the time it takes for the vehicle to reach and maintain the assigned trajectory).
- The network delay should be less than a threshold, t_N.
- The response time of the ACC system should be less than a threshold to avoid accidents before reaching the transmit line and after exiting the intersection, t_{ACC}.

Thresholds are determined based on specification of the intersection (intersection size, the distance of transmit line from the intersection, turn speed limit, wireless network, etc.), IM (WCET), network (WCND) and vehicles (size, max/min acceleration rate, etc.).

4.3 Testbed 2 - 1/10 Scale Model CAVs version 1

In order to demonstrate the effectiveness of our approach, we built a single lane 1/10 scale model intersection (Figure 4.8) with 8 CAVs Khayatian *et al.* (2018). The width of each lane is 0.6 m and the size of the intersection is 1.2×1.2 m^2.

Our CAVs are RWD (Rear Wheel Drive) cars with Traxxas Slash RC chassis. The size of each car is 0.296 $m \times 0.568$ m and has 3.5 m/s maximum speed. 3.5 m/s for our 1/10 scale model corresponds to 78mph in real-life. We used DC motors with built-in quadrature encoder to measure the longitudinal position of the CAV. Encoder data is processed by an Arduino Nano board and the processed data is then sent to the main microcontroller (Arduino Mega 2560). We utilized a Bosch BNO055 absolute orientation sensor which has built-in sensor fusion and fuses the data gathered from a 3-axis magnetometer, accelerometer, and gyroscope. Each car communicates to the server via NRF24L01+, 2.4GHz wireless module and the wireless module communicates with the Arduino Mega via the SPI protocol.

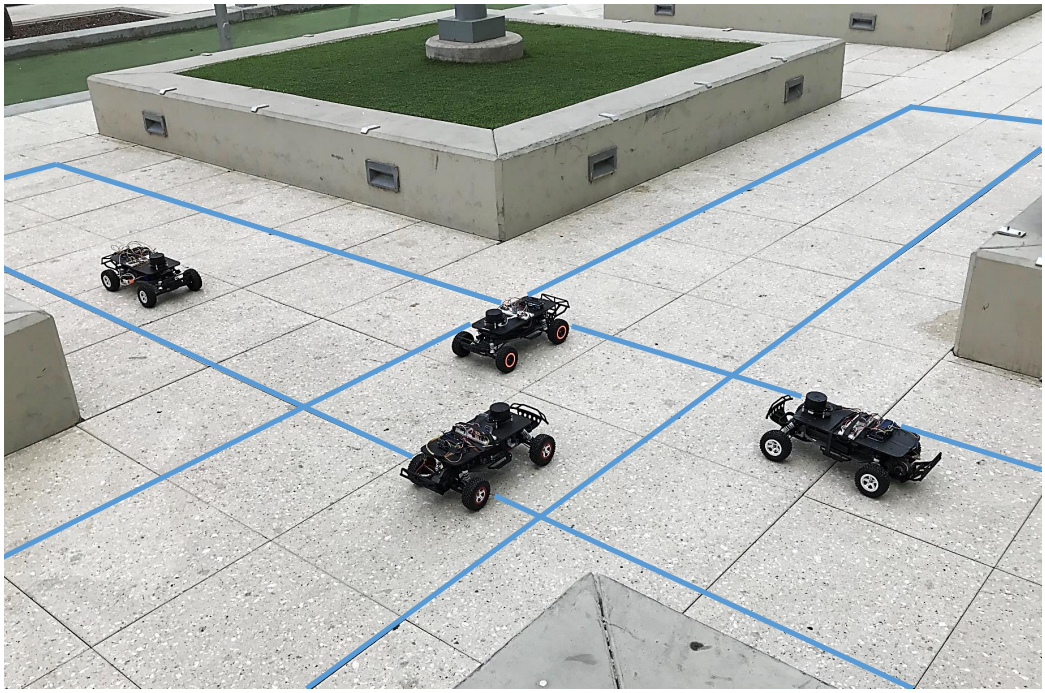


Figure 4.8: 1/10 Scale CAVs in Our Experiment. Top Speed is 3.5 m/s (78mph in full scale). Intersection and Road Lines Are Overlayed for a Better Intuition.

We set the transmit line to be 3 meters away from the edge of the intersection. CAVs are set to communicate with the IM when they reach the transmit line. The IM consists of a communication station with Arduino Mega 2560 that talks with the other

nodes via NRF24L01+,2.4GHz wireless module. Communication station sends the received data to a laptop via serial over the USB port. The IM program is executed on the laptop and is written in Matlab R2016. The laptop has 10 GB memory, Core i7 -3517u @1.9/2.4GHZ CPU and Windows 8.1 64-bit OS. Each car exchanges 44 bytes of data with IM for communication. The internal clock of all CAVs is synchronized to the IM's clock before reaching the transmit line. We developed Network Time Protocol (NTP) Mills *et al.* (1985) for synchronization and we achieved $1ms$ accuracy. We measured the WCRTD for our testbed empirically (Figure 3.3). Synchronization packet has a size of 7 bytes (1 byte for message type, 4 bytes for timestamps and 2 bytes for ID). The size of a request packet is 30 bytes, which includes ID, message type, velocity, position, captured timestamp, lane out, max acceleration, max deceleration, and max speed. The response packet has a size of 16 bytes, which includes ID, message type, TOA, VOA and transmit line distance (the distance of transmit line from the edge of the intersection). The acknowledgement packet is 8 bytes and contains A_0 and B_0. For the experiment, vehicles are placed at arbitrary positions and start driving with arbitrary initial velocities. Before reaching the transmit line, vehicles synchronize their local clock with the IM by sending a sync packet. Each vehicle monitors its position and upon crossing the synchronization line or transmit line, it sends a synchronization message or a request to the IM respectively. To estimate the worst-case delay for the IM, we need to find a reasonable value for communication delay and estimate the WCET of the IM. Figure 4.9 shows the histogram of the measured delay for the wireless network in 50 experiments. Based on the collected data, we set the network threshold to be 600 ms. As a result, the value of timeout for each vehicle can be calculated as:

$$t_{Timeout} = WCET + 2WCND$$

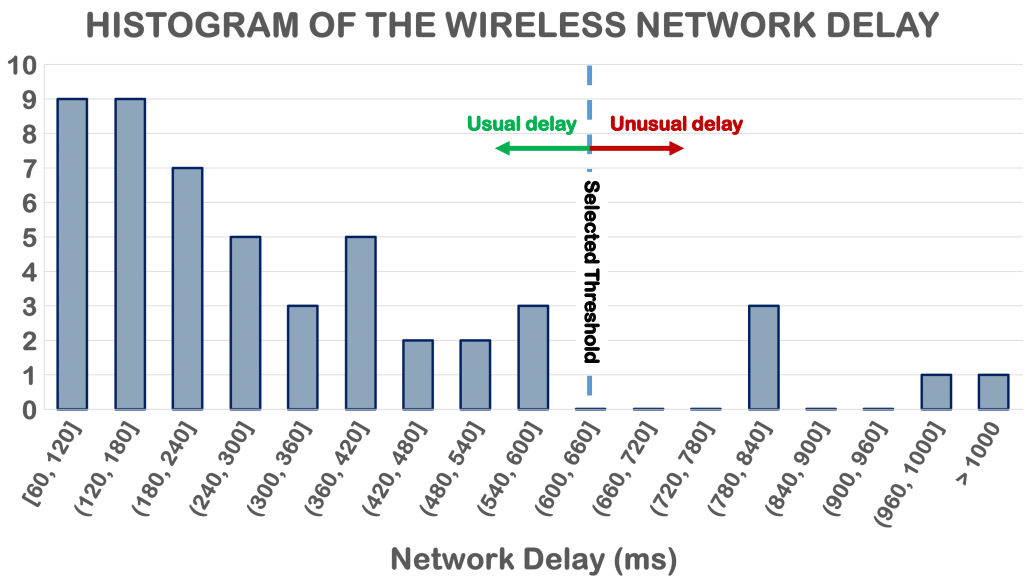


Figure 4.9: Histogram of the Measured Network Delay in Our 1/10 Scale Model Testbed. Based on the Collected Data, the Selected Threshold Value for a Communication with a Usual Delay Is Set to Be 600 ms.

The WCET of the IM is estimated based on the maximum capacity of the intersection, which is related to the maximum number of vehicles that fit in the intersection and roads before it. The estimated WCET of the IM for the microcontroller (Atmega2560 with the clock frequency of 16 MHz) is 56 ms. As a result, we set the timeout to be 1256 ms. Since vehicles ignore a response after the timeout, we can claim that the WCRTD is 1256 ms.

4.4 Results

We conducted two types of experiments: i) safety-related and ii) throughput-based experiments. The first one highlights the effectiveness of the RIM technique in reducing the position error and the second one shows the usefulness of the RIM in improving the throughput of the intersection. In safety experiments, we evaluated the impact of external disturbances and model mismatch on the eventual position of the vehicle in 3 different experiments:

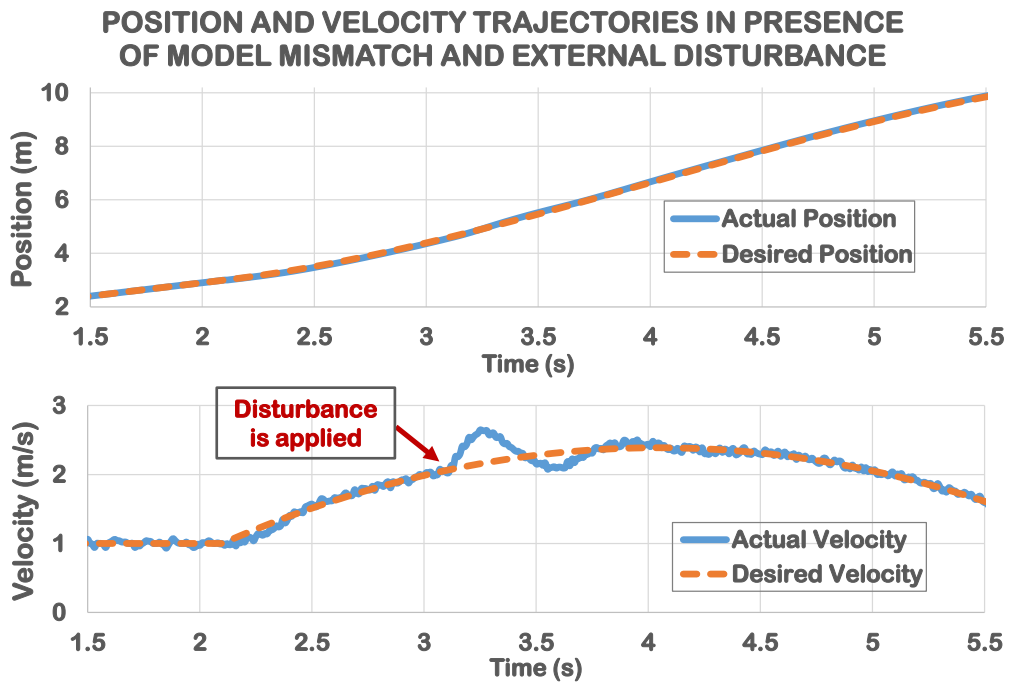


Figure 4.10: An External Disturbance Is Applied to the Vehicle That Causes a Temporary Degradation in the Velocity. However, the Vehicle Is Able to Compensate for the Effect of the Disturbance and Meet the Assigned TOA and VOA.

- **Effect of External Disturbances (ED) on TOA**

To model the external disturbance, we added a step function with the amplitude of up to 5% of the maximum range to the PWM signal (generated by the controller for the motor) and measured the position error at the expected TOA for Crossroads approach and RIM. Figure 4.10 depicts the position and velocity trajectories of a vehicle under RIM interface in presence of an external disturbance with the amplitude of 10 % of the max value. Despite the fact that the velocity trajectory of the vehicle is deviated by the external disturbance, it is still able to meet the set TOA and VOA.

- **Effect of Model Mismatches (MM) on TOA**

In Crossroads, IM has to account for the response time of vehicles when computing the target velocity and actuation time. However, the response time calculation is

done based on the considered model and can be inaccurate. To see how much model mismatches can affect the TOA, we added up to 10% error to parameters of the PID controller (K_P, K_I, and K_D), which is related to the estimated actuation time by the IM. We measured the position error at the expected TOA for both Crossroads and RIM techniques and reported the result in Figure 4.2 and Figure 4.10.

- **Effect of combined MM and ED on TOA**

In this experiment, we modeled both the external disturbance and model mismatch similar to the first and second experiments and recorded the measured position error at the expected TOA. Then, we compared the result for the Crossroads approach and RIM technique. We repeated each experiment 50 times for a different set of initial velocities and positions and, the position error is reported by storing the position of vehicles along with a timestamp on the EEPROM memory of their microcontroller. Figure 4.11 shows the average and the worst-case position error of vehicles at the expected TOA for Crossroads and RIM, normalized to the size of the vehicle. Results from Figure 4.11 indicate that on average, RIM can reduce the position error by 18X compared to the Crossroads technique. Since Crossroads and generally all VA-IM techniques ignore the effect of model mismatch and external disturbances, they are not safe and accidents can happen. In order to safely manage by just vehicles using a constant velocity, IM should consider a larger safety buffer around all vehicles to avoid accidents. Results from our experiment show that the size of the extra safety buffer can be as large as 3.2X of the vehicle length in the worst-case (MM and ED together). Considering such a large buffer around each vehicle guarantees the safety of the vehicle but is impractical since it reduces the throughput of the intersection greatly.

- **Velocity management for vehicles making a turn**

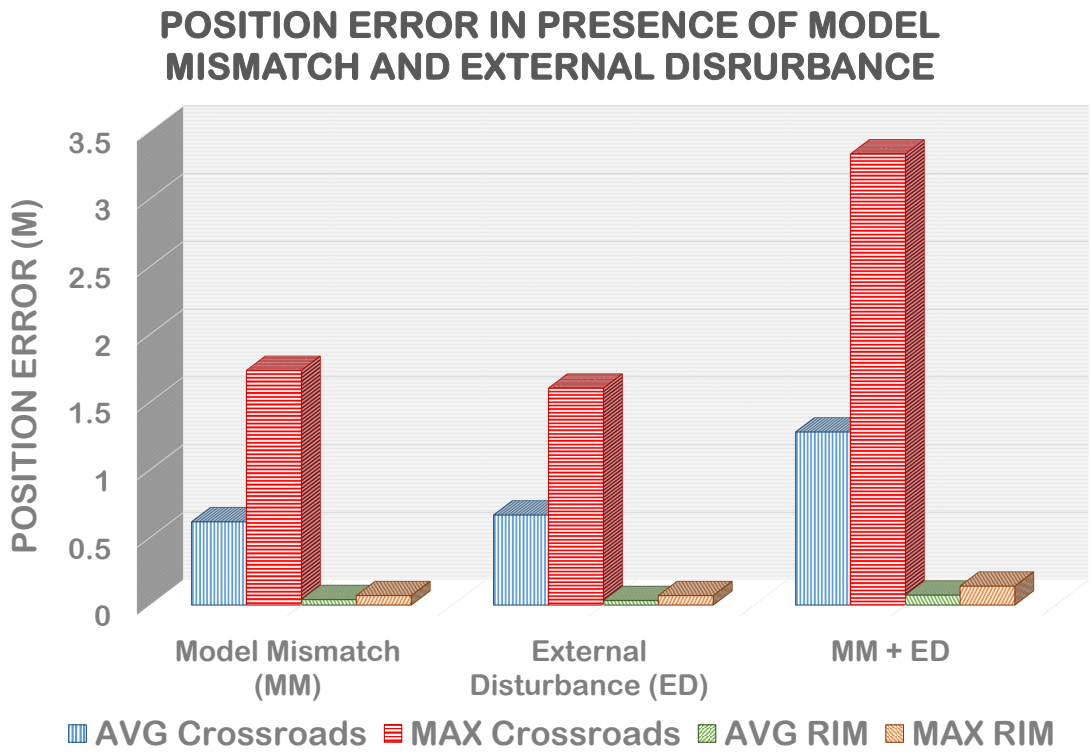


Figure 4.11: The Average and Worst-case Position Error of Vehicles At Designated TOA in Presence of i) Model Mismatches (MM), ii) External Disturbances (ED) and iii) MM and ED Together.

In intersections with a separate road for a right turn, the turn speed limit can be as high as 31 mph. However, for small intersections vehicles have to make a sharp right turn and therefore, the turn speed limit is as low as 9 mph. In this experiment, we measured the wait time of all vehicles, from transmit line to the departure of the intersection, by storing entrance and departure timestamps on the EEPROM memory of the vehicle's microcontroller. The maximum allowed velocity for making a turn in our 1/10 scale model varies from 0.4 m/s to 1.4 m/s (9 mph to 31 mph for a real intersection Fitzpatrick *et al.* (2005)) and, the speed limit (for the road) is 2.5 m/s (55 mph). Figure 4.12 shows the throughput of RIM and Crossroads normalized to the throughput of the Crossroads.

Results show that RIM can achieve 2.7X better throughputs on average in com-

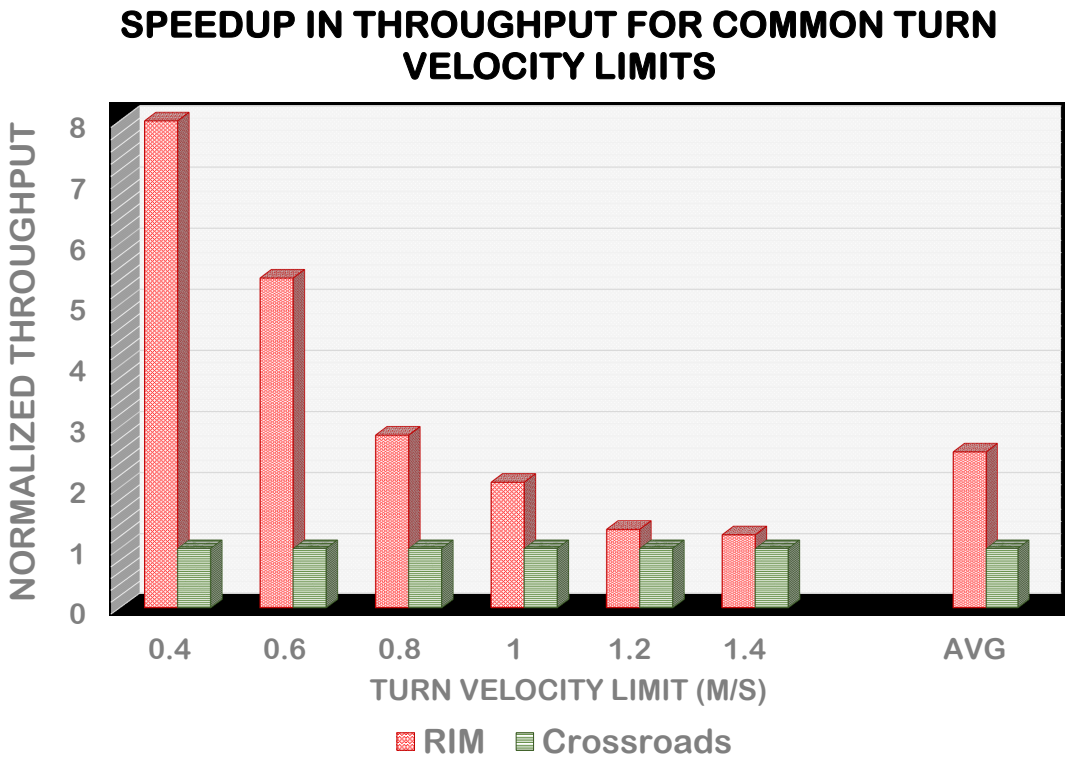


Figure 4.12: RIM Achieves Speedups in Throughput of the Intersection by Controlling the Speed Before Entering the Intersection. The Turn Velocity Limit Varies Between 0.4 to 1.4 m/s (9 mph to 31 mph for a Real-sized Intersection)

parison with Crossroads and other VA-IM techniques and, 8X in the best-case (lowest turn speed limit). The great difference in the throughput at low turn speeds has two main reasons: i) the scheduling policy of the IM and ii) induced behavior from the front vehicle. Since the scheduling policy is FCFS, a vehicle that tends to go straight will be slowed down if it is behind another vehicle that is making a turn at the intersection. For other scheduling policies like BATCHTachet *et al.* (2016), the difference can be lower. Since setting arbitrary input flow rates in real experiments is hard, we will study the effect of considering the extra safety buffer on the throughput of the intersection using our simulator.

4.4.1 *Extension to Multi-lane Intersections*

We used the result of the experiment on our 1/10 scale model autonomous vehicle to estimate the size of the extra safety buffer for the Crossroads technique and VA-IM approaches. Since the length of the vehicle is 6 m and the error due to model mismatch and possible external disturbances can be as large as 3.3X of the length of the vehicle, the extra safety buffer size is calculated as 20m (10 m in front of the vehicle and 10 m behind it). The transmit line is 200 m away from the intersection and the sync line is 250 m away from the intersection. We implemented an FCFS policy for the IM and requests are processed based on their arrival time. Figure 4.13 shows the degradation of the throughput in a single lane intersection and in a multi-lane intersection (3 lanes per road) due to considering an extra safety buffer around vehicles. Results from Figure 4.13 show that we can improve the throughput of the

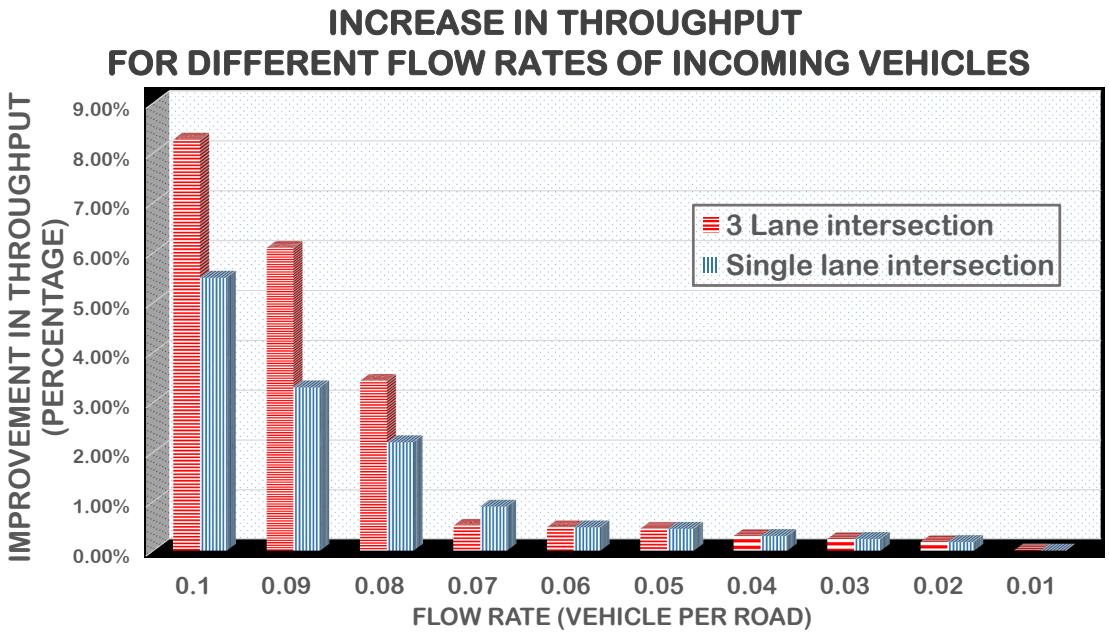


Figure 4.13: Increase in the Throughput of the Intersection for Different Values of Input Flow Rate of Incoming Vehicles. RIM Can Achieve Improvement in the Throughput since the Extra Safety Buffer for Model Mismatch and External Disturbance Is Skipped.

intersection by up to 8% for a multi-lane intersection and up to 5% for a single lane intersection when there is no need for considering an extra safety buffer for model mismatches and external disturbances. In order to fairly compare the throughput of the Crossroads technique and other VA-IM techniques against RIM, we should add the improvement result from both Figure 4.12 and Figure 4.13. This is because RIM can increase the throughput by managing the speed of vehicles making a turn at the intersection and avoid considering an extra safety buffer.

Chapter 5

ROBUST AND RESILIENT INTERSECTION MANAGEMENT (R^2IM) APPROACH

When a CAV is within the communication range of the intersection, it synchronizes its internal clock with the IM and then sends a request to the IM by sharing its position, velocity, and corresponding timestamp as well as CAV's ID and the intended outgoing lane. Accordingly, the IM calculates a safe Time of Arrival (TOA) and Velocity of Arrival (VOA) and sends it back to the CAV. Upon receiving the VOA and TOA, the CAV determines an optimal reference trajectory and lets the IM know by sharing the trajectory parameters, A_0 and B_0 (explained later). Next, the IM adds CAV's information to its list of "active CAVs" and sends an acknowledgment (ACK) to the CAV. After receiving the ACK from the IM, the CAV follows its reference trajectory until it reaches the intersection where it continues at the constant velocity of VOA. If a CAV fails to synchronize its clock or at any stage, does not receive a response from the IM within the set timeout, it will apply break and starts over by synchronizing its clock as its clock may be out of sync. IM and CAV's algorithms are presented in Alg. 6 and Alg. 11.

5.0.1 Reference Trajectory Calculation and Tracking

When a CAV receives the VOA and TOA values from the IM, it needs to make a plan to arrive at the intersection at time TOA with speed VOA. The plan is essentially a position-vs-time graph that specifies where the vehicle should be at any point in time. For simplicity, we consider a double integrator model for the behavior of the

CAVs before entering the intersection:

$$\begin{cases} \dot{p} = v \\ \dot{v} = a \end{cases} \tag{5.1}$$

where p is the longitudinal position of the vehicle, v is the velocity and a is the input acceleration. Since acceleration and deceleration rates of a CAV are bounded in real life, we consider limits for the acceleration as $a \in [a_{min}, a_{max}]$, where a_{max} and a_{min} are the maximum acceleration and deceleration rates of the CAV. Similarly, we consider an upper bound and a lower bound for the velocity of the CAV as $v \in [v_{min}, v_{max}]$, where v_{max} is the maximum velocity of the vehicle and is the same as speed limit and v_{min} is the minimum velocity of the vehicle. We determine the reference trajectory by minimizing the total amount of acceleration/deceleration for each CAV, which is linear: similar to Malikopoulos *et al.* (2018); Khayatian *et al.* (2018):

$$J = \min \int_{t_0}^{TOA} a^2 dt \tag{5.2}$$

One can construct the Hamiltonian and solves the Euler-Lagrange equations to find the optimal solution for acceleration, which will be linear:

$$a_r = A_0 t + B_0 \tag{5.3}$$

where A_0 and B_0 are constants that can be determined from initial and final conditions similar to Khayatian *et al.* (2018). by Eq. (5.5). One can determine the reference

velocity and position trajectories by taking integral from Eq. (5.3):

$$\begin{cases} v_r = \frac{1}{2}A_0(t-t_0)^2 + B_0(t-t_0) + v_0 \\ p_r = \frac{1}{6}A_0(t-t_0)^3 + \frac{1}{2}B_0(t-t_0)^2 + v_0(t-t_0) + p_0 \end{cases} \tag{5.4}$$

where v_0 and p_0 are the initial velocity and position of the CAV at the request time (t_0). Substituting initial and final conditions into Eq. 5.4, A_0 and B_0 are determined as:

$$\begin{cases} A_0 = \frac{6(2p_0 - 2POA + TOA * v_0 + TOA * VOA)}{TOA^3} \\ B_0 = \frac{-2(3p_0 - 3POA + 2TOA * v_0 + TOA * VOA)}{TOA^2} \end{cases} \tag{5.5}$$

where POA is the position of arrival (POA) at the edge of the intersection. Each CAV utilizes a PID (Proportional-Integral-Derivative) controller to track the reference position trajectory:

$$a = k_P e + k_I e_I + k_D e_D \tag{5.6}$$

where a is the control input (acceleration), e is the position error defined as $e = p_r - p$, e_I is the integral of the error ($e_I = \int e$), e_D is the derivative of e ($e_D = v_r - v$) and k_P, k_I, and k_D are positive constants which are referred to as PID gains. By tuning the PID gains, CAV's position converges to the reference position trajectory (Eq. (5.4)) within a reasonable time.

5.1 R^{2}IM with Rouge vehicles

In this section, we first present the fault model and then show the interaction of IM with CAVs to handle rogue vehicles.

5.1.1 Fault Model – Rouge Vehicle

The Rogue vehicle is a CAV that intentionally or unintentionally is lying when sharing its information to the IM, or does not follow IM's directions. The rogue CAV may either accelerate or decelerates but it never drives outside the road boundary. To generalized the rogue vehicle's definition, we use the following definition:

Definition 1 *A CAV is deemed rogue if it deviates from its expected position by a pre-set threshold.*

This fault model covers many scenarios including following extreme cases:

Acceleration (ACC) Fault Scenario: The rogue vehicle suddenly accelerates with $a = a_{max}$ toward the intersection and enters the intersection earlier than it was scheduled.

Deceleration (DEC) Fault Scenario: The rogue vehicle breaks down and suddenly stops ($a = -\infty$) inside the intersection area.

Lying about outgoing Lane: The rogue vehicle lies about its outgoing lane and takes another path once it enters the intersection. Next, we define some of the terms that we use in the algorithm.

Definition 2 ***Point of No Return (PONR)*** *is the farthest point from the intersection that if after passing this point a CAV starts applying full brake ($a = a_{min}$), it cannot fully stop without entering the intersection.*

The distance of the PONR from the edge of the intersection (POA) is d_{PONR} and can be calculated as:

$$d_{PONR} = \frac{v_{PONR}^2}{2|a_{min}|} + \frac{VL}{2} \tag{5.7}$$

VL is vehicle length and v_{PONR} is the velocity of the CAV at the PONR. Since p represents the longitudinal location of the center of a CAV, $\frac{VL}{2}$ is added to account

for the length of the CAV.

Definition 3 ***Critical zone*** *for a CAV is defined as the area between its PONR and the point it exits the intersection.*

Definition 4 ***Critical time window*** *is the time it takes for a CAV to travel through the critical zone.*

The critical window (Δt_{crit}) can be calculated as the summation of time to reach the intersection and time to travel inside the intersection:

$$\Delta t_{crit} = (TOA - t_{PONR}) + \frac{d_I + 0.5VL}{VOA} \tag{5.8}$$

where d_I is the traveled distance inside the intersection and can be determined from the dimensions of the intersection. For left and right turns, the d_I is $\frac{3\pi LW}{2}$ and $\frac{\pi LW}{2}$ respectively and for going straight d_I is $2LW$ where LW is the lane width.

Definition 5 ***Safety Barrier (SB)*** *is the maximum distance that a CAV may travel when the previously scheduled CAV is in its critical zone.*

Since the velocity of a CAV is bounded by v_{max}, the maximum distance that a CAV can travel corresponds to the case where its initial velocity is equal to v_{max}. As a result, the size of the safety barrier is:

$$d_{SB} = \Delta t_{crit} v_{max} \tag{5.9}$$

For a practical design, the IM should account for the worst-case execution time of the IM (C_{IM}) and CAVs (C_{CAV}), and the period of emergency inquiry by a CAV (T) to ensure safety. As a result, Eq. (5.7) and (5.9) is modified as follows to account

for them:

$$d_{PONR} = \frac{v_{PONR}^2}{2|a_{min}|} + \frac{VL}{2} + \rho v_{max} \tag{5.10}$$

$$d_{SB} = (\Delta t_{crit} + \rho) v_{max} \tag{5.11}$$

where $\rho = T + C_{IM} + C_{CAV}$ is the worst-case end-to-end delay from the moment a CAV becomes rogue to the moment other CAVs are notified and react.

5.1.2 IM and CAV Interaction In Presence of A Rogue Vehicle

The IM periodically calculates the distance between the estimated position of CAVs – which is determined from a CNN-based perception system like Farhadi *et al.* (2019) – and their expected position. If the distance is greater than a threshold, e_{th}, it set the emergency state to active. Whether the emergency state is active or not, the IM periodically broadcasts it. When a CAV notifies that the emergency state is active, it checks if its position is before its *Point of No Return* (PONR) (Eq. (5.7)) and can safely stop without entering the intersection. If the CAV is after its PONR, it ignores the emergency state and continues with its trajectory. If a CAV does not receive the emergency state within the set timeout, it assumes that the emergency state is active and applies the brake. During the emergency state, the IM rejects all the requests that are received from CAVs. When the emergency is resolved the IM sets the emergency state to false and starts processing the requests that it receives.

The IM adopts a First-Come First-Served (FCFS) policy for scheduling. Given the request time, position, and velocity of the requesting CAV, and expected trajectories of other CAVs are known, the IM determines a VOA and TOA pair for the requesting CAV such that the earliest arrival time is achieved. To do so, the IM solves the following optimization problem:

$$\min TOA \tag{5.12}$$

subjected to following constraint:

$$\begin{cases} p_i(t_{PONR,i-1}) < p_{SB,i} \\ p_i(t_{exit,i-1}) < p_{PONR,i} + \rho v_{max} \\ p_i(t) > p_{i,front}(t) + d_{PONR,i} & \forall i > 1 \\ a_{min} < a_i(t) < a_{max} \\ v_{min} < v_i(t) < v_{max} \\ VOA < v_{turn} \end{cases} \tag{5.13}$$

where $p_i(t_{PONR,i-1})$ is the position of the requesting CAV (i) when the last scheduled CAV ($i-1$) is at its PONR and $p_{SB,i}$ is the safety barrier point that is $d_{SB,i}$ meters away from the intersection. The first constraint ensures that the requesting CAV (i) is far enough from the intersection when the last scheduled CAV ($i-1$) is at its PONR and does not cause a conflict for the last scheduled CAV. $p_i(t_{exit,i-1})$ is the position of the CAV (i) when the last scheduled CAV ($i-1$) leaves the intersection and $p_{PONR,i}$ is the position the PONR for the requesting CAV. The second constraint ensures that the requesting CAV has enough time to stop if the last scheduled CAV slows down and stops inside the intersection. $p_{i,front}(t)$ is the position of the last scheduled CAV in the same lane as the requesting CAV or simply CAV i's front CAV if any. The third constraint ensures the requesting CAV's trajectory has always a minimum safe distance from its front CAV. Fourth and fifth constraints are considered to ensure that the velocity and acceleration of the requesting CAV are within the feasible range. Finally, v_{turn} is the maximum safe velocity for making a turn to avoid rollover. For driving straight, $v_{turn} = v_{max}$. To ensure that constraints in Eq.(5.13) are met, the IM reconstructs the trajectory of the last scheduled CAV ($i-1$) and the CAV's

front CAV (if any) from the request information (t_0, v_0, p_0), schedule information (TOA, VOA, POA) and trajectory information (A_0, B_0).

5.2 Safety Proof

We assume that only one rogue vehicle is present at a time and prove that no accident will happen inside the intersection area. We are limiting our proof to provide safety for the intersection area only because a rogue vehicle may accelerates or steer to the opposite lane and hits another vehicle and in such cases, an accident may be unavoidable. We also assume that the roads connected to the intersection are not curved and if a CAV drives out of road's boundary, it will travel a longer distance compared to driving within the lanes. We will show that the rogue vehicle cannot get involved in an accident with the last scheduled CAV or the next scheduled CAV (if any). For a better intuition, we have depicted these two corner cases in Figure 5.1.

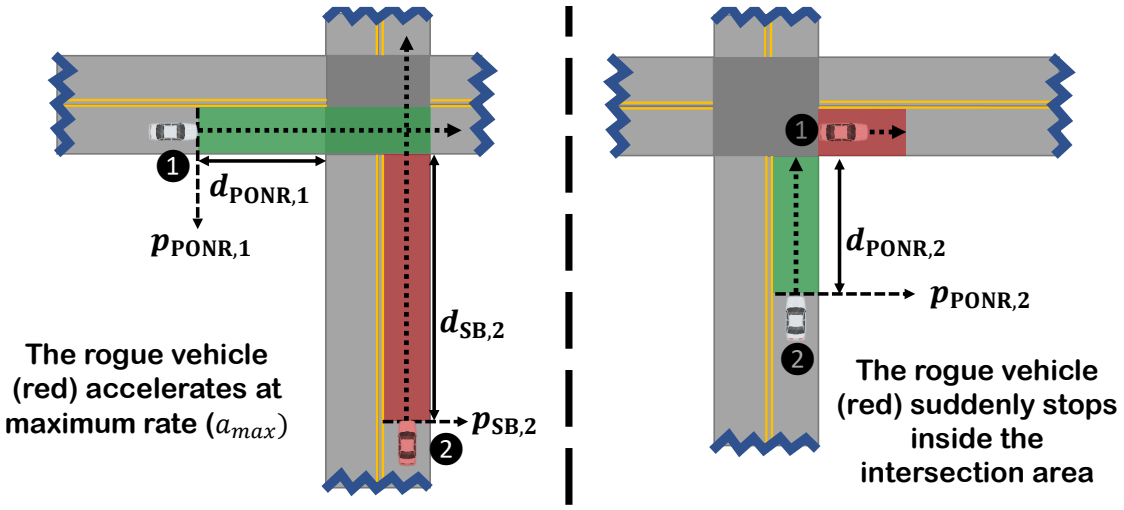


Figure 5.1: Two Corner Cases Where a CAV Becomes Rogue.

5.2.1 Interaction of Rogue Vehicle with Last Scheduled CAV

Let us assume that vehicle i becomes rogue and accelerates at time t_{rogue}. If the distance of the rogue vehicle from the intersection when it becomes rogue is more than d_{SB}, the earliest time that it can enter the intersection is after $t_{rogue} + \Delta t_{crit} + \rho$. Since the last scheduled CAV will reach and leave the intersection by Δt_{rogue}, the rogue vehicle enters the intersection when the last scheduled CAV has already left the intersection. If the distance of the rogue vehicle from the intersection when it becomes rogue is less than d_{SB}, the last scheduled CAV will be behind its point of no return according to Eq. (5.13). Therefore the last scheduled CAV can stop safely without entering the intersection.

5.2.2 Interaction of Rogue Vehicle with Next Scheduled CAV

If the rogue vehicle breaks down and suddenly stops before exiting the intersection, the next scheduled CAV will have at least $d_{PONR}+\rho v_{max}$ distance from the intersection (see Eq.(5.13)). As a result, even if it takes ρ milliseconds to notify the next scheduled CAV, it will have enough distance to stop without entering the intersection.

If a CAV lies about its position, velocity, or the corresponding timestamp, the IM will detect it using its surveillance system. Similarly, if a CAV reaches the intersection range and does not share its information, it will be detected as rouge. In both cases, all CAVs will have enough distance to stop or continue because the distance of the rogue vehicle is greater than d_{SB}. If a CAV lies about its outgoing lane and suddenly changes its direction inside the intersection, (e.g. makes a left turn instead of going straight), the next scheduled CAV will be beyond its PONR (see Eq. (5.13)) and therefore has enough distance to stop without entering the intersection similar to the deceleration case.

5.3 Testbed 3 - 1/10 Scale Model CAVs version 2

Our experimental testbed is an intersection with 1/10 scale model CAVs that are 57 cm long and 30 cm wide. An ESP8266 NodeMCU v3 board is utilized to enable wireless communication of CAVs with the IM and perform real-time motion control by adjusting the steering angle and speed for the DC motors that run the CAV. A set of markers is installed on each vehicle and the OptiTrack monitoring system is used to track CAVs' positions. We use `mocap_optitrack` library from ROS (Robot Operating System) to read the CAVs' 2D pose data. We have written a script to create a ROS node and to pass the pose data to vehicles through TCP/IP sockets. This script also acts as the IM and interacts with the CAVs. Figure 5.2 shows an overview of the intersection. The pose data and the emergency state packet is broadcast every 20 ms.

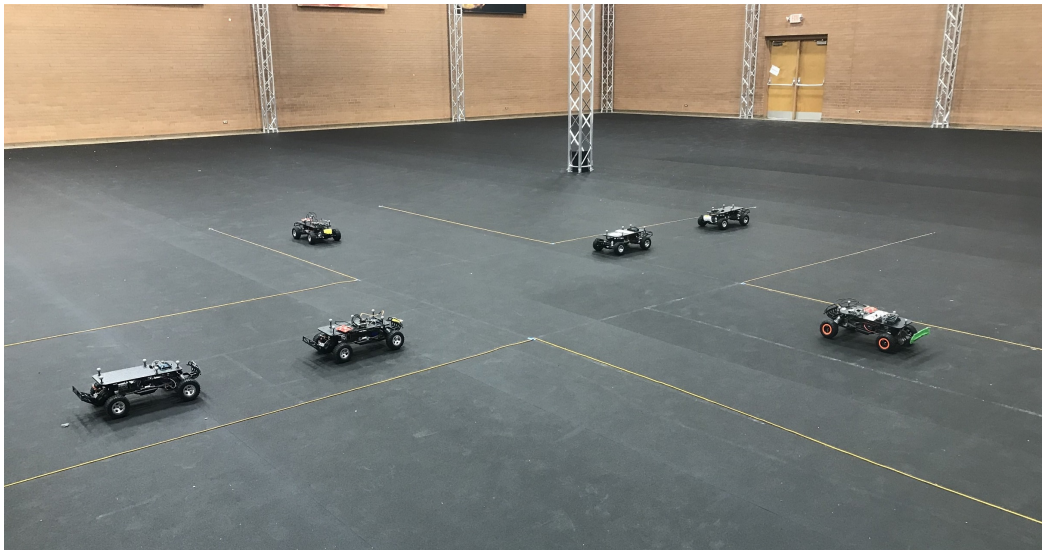


Figure 5.2: Our Testbed Is a Single-lane Intersection with 1/10 Scale Model CAVs. CAVs Position is Tracked by the OptiTrack System.

Timing constraints of each AV are also monitored at the runtime Mehrabian *et al.* (2018). CAVs start from arbitrary positions and track a set of way-points to drive within the lane and cross the intersection. When a CAV reaches the end of the road,

it makes a u-turn and randomly makes a new plan to cross the intersection. Then it drives toward the intersection and communicates with the IM again. To make sure that CAVs maintain enough distance from their front vehicle when they reach the end of the road, we have implemented an Adaptive Cruise Control (ACC) algorithm for CAVs. The maximum velocity of CAVs is $3m/s$, and the maximum acceleration and deceleration rates are measured as $2m/s^2$ and $-1.5m/s^2$ respectively. We set the minimum velocity to be $0.2m/s$. IM uses the OptiTrack real position data as monitoring data too. Since the OptiTrack system is accurate up to the millimeter level, we added a random number between [0, 0.1] m to the position of CAVs in order to emulate the error for a realistic surveillance and detection system.

5.4 Results

We performed two types of experiments to validate the safety of our approach when a rouge vehicle is present: 1) systematically injecting a fault on a CAV and 2) randomly injecting a fault.

5.4.1 Safety Validation by Systematic Fault Injection

For the systematic fault injection, we created a scenario where two CAVs are scheduled to cross the intersection and the CAV that is scheduled to cross second becomes rogue after some time and accelerates. We created the same scenario and repeated our fault injection but at different times. Using the brute-force approach, we injected a set of ACC fault on the secondly scheduled CAV at every 0.1s where the fault injection time varies from 34.5 to 36. Figure 5.3 shows the distance between CAVs and the unsafe area.

When the fault is injected before the t_{PONR}, the other CAV stops but when the fault injection time is after the t_{PONR}, the other vehicle continues. It can be observed

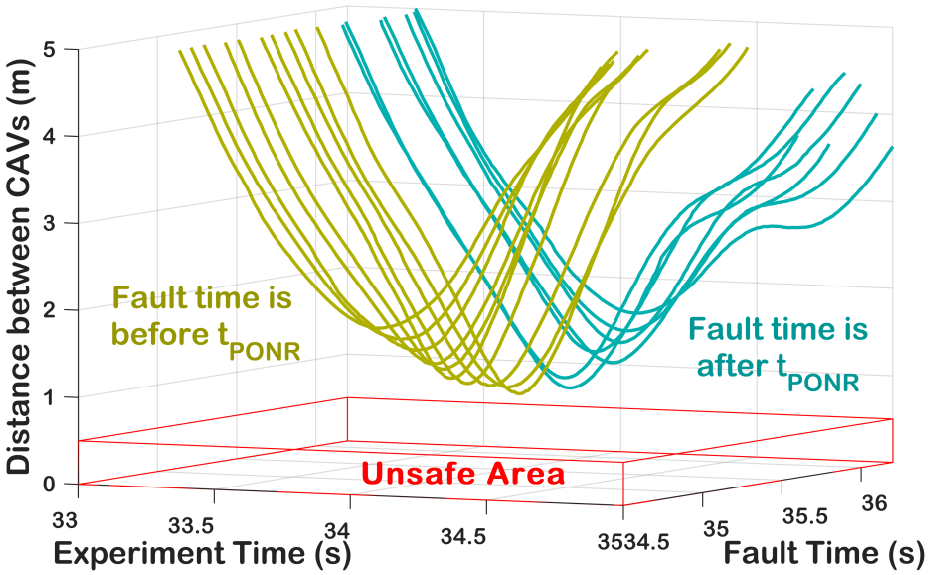


Figure 5.3: Systematic Fault Injection of an ACC Fault on Our 1/10 Scale Model Intersection. By Increasing the Fault Time, the Distance Between CAVs Decreases until the t_{PONR} and Then It Increases.

that for all experiments, the distance between vehicles is always greater than 0.5m.

To check the scalability of our approach for a real-size intersection with variable traffic patterns and also do fault injection more precisely, we built a simulator in Matlab and performed the same fault injection experiment. The length of CAVs is 5m and their width is 2m, the maximum velocity of CAVs is set to $20.1m/s$ (45mph), and maximum acceleration and deceleration rates are measured as $6m/s^2$ and $-7m/s^2$ respectively. The simulator models network communication between CAVs and IM by randomly adding a delay to the communicated packets and models the vehicle dynamics (Eq. (5.1)).

Our simulator can generate traffic patterns at a given flow rate by instantiating new CAVs. Unlike our real-life testbed, in our simulator, CAVs are spawned at a given time and removed when they leave the intersection. The simulator was executed on a desktop machine with Intel Core i7-6700 CPU @ 3.4GHz, 16.0 GB of Memory, and 64-bit Windows OS.

We created a scenario similar to the previous experiments in our simulator where two CAVs are about to cross the intersection and we conducted the same systematic fault injection experiment. Figure 5.4 shows the distance between CAVs when the secondly scheduled CAV becomes rouge at different moments. It can be seen that

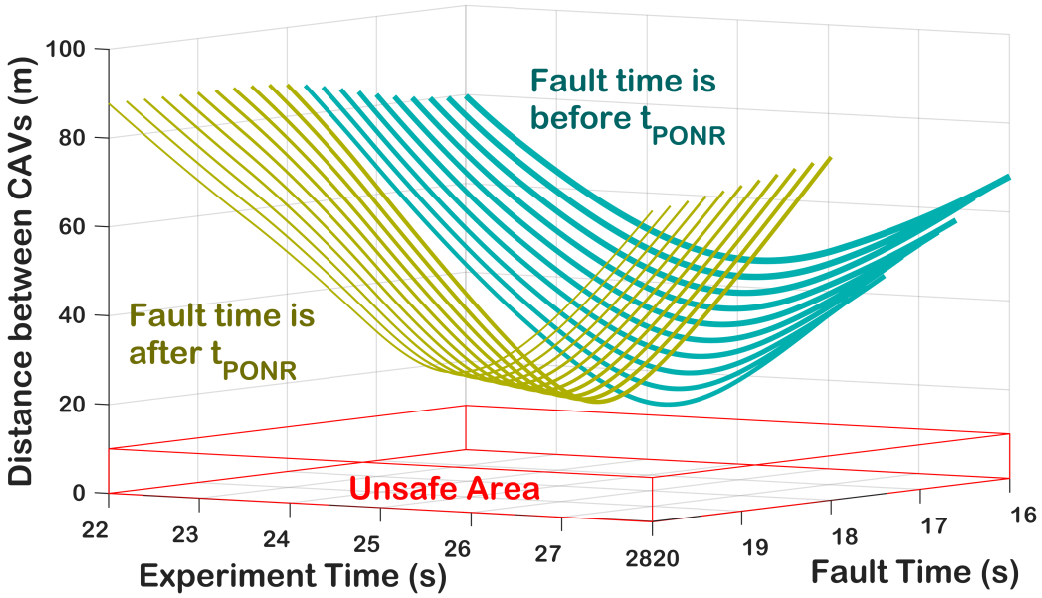


Figure 5.4: Systematic Fault Injection of an ACC Fault on Our Simulator. The Distance Between CAVs Never Reaches the Unsafe Limit.

the distance between CAVs is always greater than 10m. Also, the results from the simulation match the real-life experiments.

5.4.2 Randomized Fault Injection:

We performed another experiment where the fault injection is done randomly. This experiment was done on our simulator where the fault injection time was selected between [0, 10]s after the spawn time of a CAV, the traffic flow was randomly selected between [0.01, 0.15] car/second/lane and the probability of injecting a DEC fault was equal to an ACC fault. In total, 500 faults were injected during 20 hours of simulation,

out of which 110 could cause an accident but they were avoided.

5.4.3 Recovery Analysis

To check if the intersection can recover after a rogue vehicle leaves the intersection area (for an ACC fault) or is removed (for a DEC fault) by a tow truck, we measured the average travel time of CAVs when a fault is injected. We first measured the average travel time for the normal operation of the intersection where no rogue vehicle is present and then repeated the same experiment and injected an ACC fault with the same traffic pattern and measured the average travel time. Next, we repeated the same experiment and injected a DEC fault with the same traffic pattern and measured the average travel time. These three experiments were repeated 20 times with different traffic patterns and the results were averaged. The final results for the average travel time of CAVs are shown in Figure 5.5. For the DEC fault, the

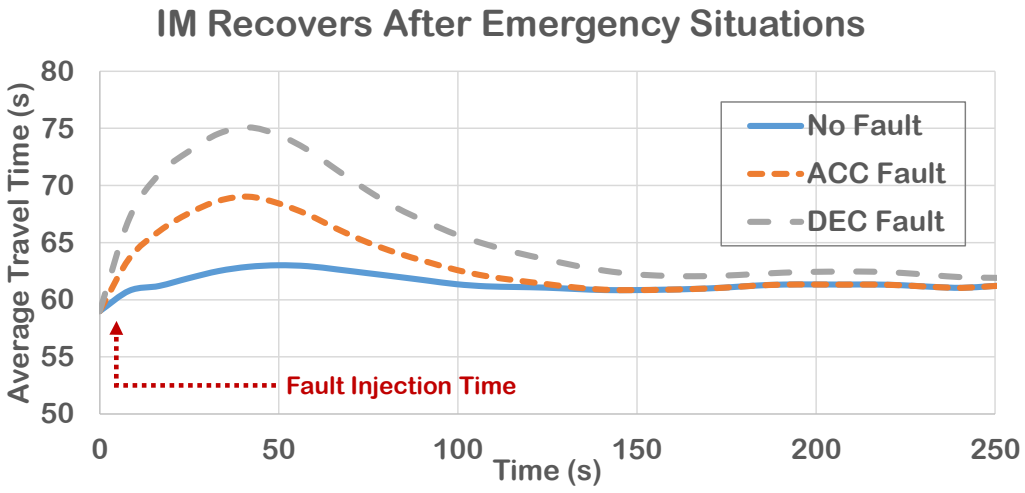


Figure 5.5: Average Travel Time of CAVs Increases When an ACC or DEC Fault Is Injected But the Intersection Recovers.

rogue vehicle stops for 10 seconds and then is removed. As can be expected, a DEC fault results in larger travel times compared to an ACC one since the duration of

the emergency state is longer. Results show that the average travel time of CAVs increases when a fault is injected but the IM is able to recover and reduce the average travel time once the emergency situation is resolved.

5.4.4 Comparison with Traffic Light

To fairly compare the throughput of the R^{2}IM with other scheduling approaches, we developed a simulator for one of the state-of-the-art technique Robust Intersection Management (RIM) Khayatian *et al.* (2018) and a simulator for an intersection that is managed by a traffic light. The green time of the traffic light is set to be 25 s, yellow time 5 s and red time 30 s. The same input flow and traffic patterns (spawn times) were used for all experiments to fairly compare the throughput of the intersection. Figure 5.6 shows the output flow rate of CAVs vs the input flow rate for R^{2}IM,

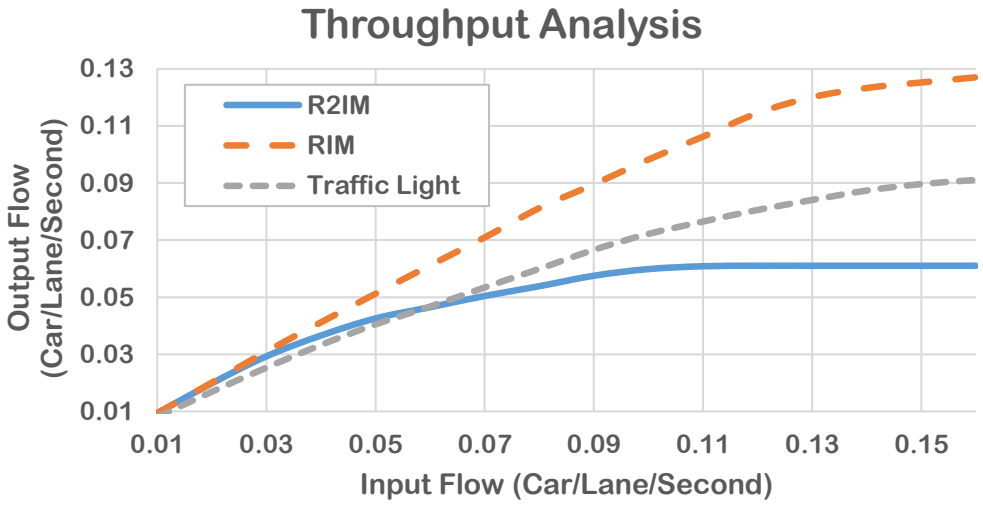


Figure 5.6: Comparing the Output Flow Rate of an Intersection Managed by a Traffic Light, R^{2}IM and RIM Approaches.

traffic light and the RIM approach when the input flow rate varies from 0.01 to 0.16 car/lane/second. The output flow is measured by counting the number of CAVs that leave the intersection in a 5-second time interval divided by the length of the interval

(5). It can be observed that R^{2}IM always achieves a lower throughput compared to the RIM technique and this is because R^{2}IM allocates a larger temporal buffer between arrival times of CAVs for protecting CAV from a rogue vehicle. When comparing our approach with a traffic light, the R^{2}IM performs better than traffic light for low traffic flows ([0.01 to 0.05]) while traffic light performs better for higher flows. However, it should be noted that R^{2}IM can avoid accidents that cannot be avoided if the intersection is managed by a traffic light or RIM approach. We believe that safety is of utmost importance and although rogue vehicles are rare, it's reasonable to sacrifice performance for safety.

5.4.5 Real-time Calculation of Optimal Solution

Solving the optimization problem presented in Eq. (5.12) and Eq. (5.13) is the most compute intensive part of the IM's algorithm and to ensure safety, the IM's processing time should be bounded by C_{IM} (see Eq. (5.11)). Conventional optimization methods/solvers can take up to seconds to find the optimal solution which makes them impractical to use. To efficiently solve the optimization problem, we use a heuristic approach, where we discretize the solution space (TOA vs VOA) and refine the search space by removing solutions that do not satisfy the constraints. We bound the search space for VOA to be between $v_{min} = 5$ and $v_{max} = 20$ and for TOA to be between $t_0 + 1$ and $t_0 + 60$. We have depicted the feasible VOA and TOA for a scenario in Figure 5.7.

Discretizing the space may result in assigning sub-optimal TOA and VOA values to CAVs and therefore can degrade the throughput of the intersection. We conducted an experiment where the discretization granularity of our approach varies. We measured the processing time of the IM for different configurations and reported the processing and average travel time time in Table 5.1. By increasing the accuracy our approach,

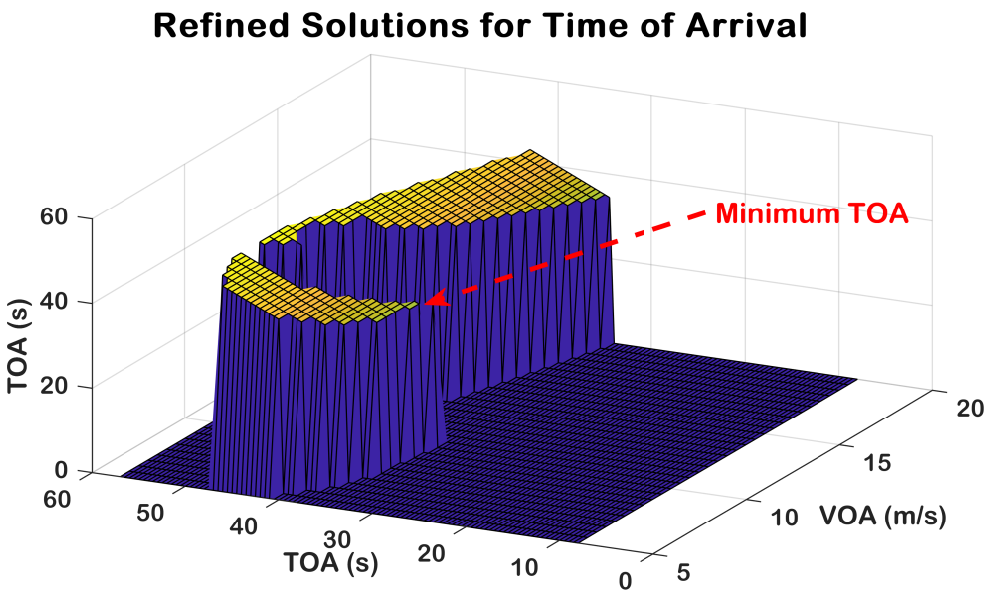


Figure 5.7: Solutions Satisfying All Constraints of Eq. (5.13) for an Example. The Minimum TOA Is Highlighted in Red.

the average travel time of CAVs decrease. However, the reduction amount is negligible (less than 1 second). Based on the result, we set the granularity for VOA and TOA to be 1 since it results a low processing time and a reasonable average travel time.

Algorithm 6: Algorithm for CAVs

```
1  while True do
2      if (within the intersection range) then
3          synchronize the clock;
4          if sync is not successful or timed out then
5              apply brake and goto line 3;
6          end
7          send a request to IM;
8          receive the TOA and VOA from IM;
9          if response is timed out then
10             apply brake and goto line 3;
11         end
12         calculate reference trajectory;
13         send trajectory information to the IM;
14         receive the ACK from the IM;
15         if ACK is timed out then
16             apply brake and goto line 3
17         end
18         inquiry emergency state from IM;
19         if emergency state is active or timed out then
20             if (After point of no return (PONR)) then
21                 follow reference trajectory and goto line 18;
22             else
23                 apply brake and goto line 3
24             end
25         else
26             if (if entered the intersection) then
27                 drive at a constant velocity (VOA)
28             else
29                 follow reference trajectory goto line 18
30             end
31         end
32     end
33 end
```

Algorithm 7: Algorithm for the Intersection Manager

```
1  while True do
2      check emergency state;
3      broadcast emergency state;
4      if a request is received then
5          if emergency state is active then
6              reject the request;
7          calculate the optimal TOA and VOA;
8          send TOA and VOA to the vehicle;
9      if trajectory information is received then
10         store the CAV's trajectory information;
11         send the ACK message;
```

Configuration #	TOA step size (s)	VOA step size (m/s)	Processing Time (s)	AVG Travel Time (s)
1	2	2	0.117	66.675
2	1	1	0.129	65.389
3	0.2	0.5	0.467	65.230
4	0.1	0.5	0.810	65.198
5	0.1	0.1	2.826	64.949

Table 5.1: IM's Processing Time for Solving the Optimization Problem and the Corresponding Average Travel Time of CAVs.

Chapter 6

RSS-BASED MOTION PLANNING FOR CAVS

Autonomous Vehicles (AVs) have the potential to make transportation safer by reducing the number of accidents that are caused due to human error. When AVs become connected (which are referred to as Connected Autonomous Vehicles (CAVs)), they can further improve road safety by sharing their information with each other such as position, velocity, future plans, etc. In addition, CAVs are projected to improve fuel consumption, travel time, and passenger comfort through cooperative driving.

Achieving cooperative behaviors among robots is typically studied under multi-agent motion planning in the robotics domain Mohanan and Salgoankar (2018); Rossi *et al.* (2018). Existing techniques can be categorized into two groups: i) centralized Kant and Zucker (1986); Peng and Akella (2005) and ii) decentralized (distributed) Erdmann and Lozano-Perez (1987); Bekris *et al.* (2007). In centralized approaches, it is assumed that a central planner exists that has access to all information and computes the trajectory for robots e.g. path-velocity decomposition technique, while in decentralized approaches, each robot is assumed to have incomplete information and autonomously determines a plan while avoiding static and moving obstacles as well as other robots. Although centralized approaches can find the optimal solution, they are computationally demanding and less tolerant of uncertainty. On the same lines, in the Intelligent Transportation System (ITS) domain, centralized and decentralized techniques Chen and Englund (2015); Rios-Torres and Malikopoulos (2016b) are proposed where CAVs share their information with each other (through V2V) or the infrastructure (through V2I) to perform traffic management at intersections Zheng *et al.* (2019); Khayatian *et al.* (2018); Lin *et al.* (2019), merges Lu and

Hedrick (2003); Rios-Torres and Malikopoulos (2016a); Aoki and Rajkumar (2017).

In general, existing motion planning algorithms and traffic management techniques consider a safety buffer around each vehicle to cover for uncertainties in the localization and trajectory tracking, and then a reference trajectory is determined. A trajectory is considered to be safe if the safety buffer of the vehicle does not overlap with obstacles or other vehicles' safety buffer at any point in time. While reasonable, this definition may not provide absolute safety because it implicitly assumes that all vehicles will follow the plan (with small errors that are within the safety buffer). However, any disruption in the plan can cause an accident. For example, consider a scenario when two vehicles are driving on a street, one behind the other. If the front vehicle suddenly stops for any unplanned reason (e.g. yielding to a jaywalker), then the rear vehicle may hit the front car. In common driving parlance – the rear vehicle should not tail-gate the front vehicle.

Responsibility-Sensitive Safety (RSS) approach Shalev-Shwartz *et al.* (2017) from Mobileye+Intel addresses the safety issue from the legal/blame perspective and allows vehicles that have the right-of-the-way according to the rules of the road to change their plans. RSS proposes a set of safety rules such that if a vehicle abides by these rules, then it cannot be blamed for an accident. In the scenario that is mentioned above, RSS rules are used to determine the minimum distance at which the rear vehicle should follow the front one so that it will be able to stop without causing an accident even in the worst-case scenario. RSS uses a lane-based coordinate system to define lateral and longitudinal distances between vehicles depending on the driving scenario. For example, there is a longitudinal rule for the scenario when two vehicles are one in front of the other, and there is a lateral rule for the scenario in which two vehicles are driving in parallel to each other. The longitudinal direction is toward the center-line of the lane and the lateral direction is perpendicular to the center-line of the lane.

The main shortcoming of RSS is that it is scenario-based and not all scenarios are covered because longitudinal and lateral distances cannot be computed for merges, intersections, and unstructured roads where lane markings are not provided.

When CAVs interact with each other in different scenarios, they may face a deadlock situation where CAVs yield to each other for an indefinite time. Researchers have proposed methods to detect and resolve deadlocks at intersections Lin *et al.* (2019); Liu *et al.* (2017); Perronnet *et al.* (2019) and roundabouts Azimi *et al.* (2014). In existing approaches, the intersection/roundabout area is divided into a grid of zones, and vehicles that intend to occupy the same zone are said to have a conflict. Then, the dependencies between CAVs (who should enter a conflict zone first and who enters second) are represented with a directed graph, and deadlocks are resolved by removing cycles in the graph. One of the limitations of existing approaches is that they use fixed zones to detect conflicts between vehicles and the size of each zone affects the efficiency and computational complexity of the conflict detection algorithm since using coarse grids makes the schedule pessimistic and using fine grids increases the number of checks. Furthermore, in existing approaches, the dependency graph is computed individually by each CAV, which is extremely inefficient because the same computing is done redundantly and the overhead grows as the number of vehicles increases.

In this chapter, we present a cooperative driving and deadlock resolution approach for CAVs Khayatian *et al.* (2021). Instead of a lane-based coordinate system, we use future trajectories of CAVs to represent their conflicts, which can be applied to any road geometries and situations. Inspired by the RSS legal/blame perspective, we develop a new set of safety rules for CAVs to guarantee that no accidents happen if CAVs abide by proposed RSS rules. We also provide an efficient and decentralized deadlock detection and resolution algorithm for CAVs. The integration of the pro-

posed RSS safety rules and deadlock resolution algorithms with motion planning is also provided. Results from conducting experiments on our realistic simulator –that considers vehicle dynamics and network delay– demonstrate that all CAVs remain safe even if one or more CAVs slow down or stop at any point in time. We evaluate the efficiency of our approach by comparing the average travel time of CAVs with a case that vehicles are autonomous but not connected. Finally, we showcase our deadlock resolution mechanism for an intersection scenario.

6.1 Generic Formulation of RSS Rules

In this section, we introduce a trajectory-based formulation for RSS rules. The advantage of this approach is that the rules are generic and can be applied to all cases, including unstructured roads.

Given the future paths of CAVs are known, each CAV can determine the set of conflict zone C. A conflict zone, $C_i \subset C$ is defined as a convex contour that includes a subset of two CAVs' future path (FP) where the distance between the future paths is less than a threshold, d_{th}. Since two CAVs may have more than one conflict, only consecutive edges that have a distance of less than d_{th} are considered to be a part of the same conflict zone. The midpoints of the edges are used to calculate the distance between two edges from two future paths. To specify the boundaries of a conflict zone, midpoints of first and last edges are used.

Based on the road geometry and rules of the road, every pair of CAVs can determine who has the advantage to enter the conflict zone first and who has the disadvantage. For simplicity, we assume the CAV with the earlier arrival time has the advantage. Without loss of generality, we assume that one of the CAVs has the advantage and the other one has disadvantage. We represent the distance of the CAV with the advantage from the beginning of the conflict zone and from the end of the

conflict zone with d^A_{begin} and d^A_{end}, respectively. Similarly, we represent the distance of the CAV with disadvantage from the beginning of the conflict zone with d^D_{begin}. Figures 6.1, 6.2, and 6.3 show different scenario where the d^A_{begin}, d^A_{end} and d^D_{begin} are shown. We assume that Equation6.12 represents the dynamics of each CAV. For simplicity, the trajectory of each CAV is projected onto its path and represented with the double-integrator model. As a result, the stop distance of the CAV with advantage is calculated as:

$$d^A_{stop} = \frac{v_A^2}{2|a_{brake}|} \tag{6.1}$$

We assume that each CAV broadcast its information every T milliseconds and the worst-case end-to-end delay (ρ) is $2T$. Taking into account the delay, the worst-case stop distance of the CAV with disadvantage is calculated as:

$$d^D_{stop} = v_D\rho + \frac{1}{2}a_{ACC}\rho^2 + \frac{(v_D + a_{ACC}\rho)^2}{|2a_{brake}|} \tag{6.2}$$

The first two terms ($v_D\rho$ and $\frac{1}{2}a_{ACC}\rho^2$) indicate that the CAV with disadvantage may be accelerating in the worst-case scenario while waiting for broadcast information from the CAV with advantage. If the distance of the CAV with advantage form the end of the conflict zone is greater or equal to the stop distance of the CAV with advantage ($d^A_{end} \geq d^A_{stop}$), there is a possibility that it may slow down and stop inside the conflict zone and block the CAV with the disadvantage. Otherwise, there is no conflict. Accordingly, we define the modified RSS rule as:

Definition 6 ***General RSS Rule:*** *Given the order of entering a conflict zone is known, the minimum safe distance to maintain from the conflict zone (d^D_{SAFE}) for the CAV with disadvantage is:*

$$d^D_{SAFE} = \begin{cases} d^D_{stop} - d^A_{extra} + \frac{VL_A + VL_D}{2} & \text{if } \; d^A_{end} > d^A_{stop} \\ 0 & \text{otherwise} \end{cases} \tag{6.3}$$

where d^A_{extra} is the scenario-dependant distance that the CAV with the advantage travels inside the conflict zone making extra room for the CAV with a disadvantage, and VL_A and VL_D are the vehicle length for the vehicle with advantage and disadvantage, respectively. Since the distance values are calculated based on the center of CAVs, the term $\frac{VL_A + VL_D}{2}$ is added.

Lemma 1 *If the CAV with disadvantage always maintains a distance of at least d^D_{SAFE} from its conflict zone, it will not hit the CAV with advantage even if it changes its plan and decelerates at any point in time.*

Proof 1 *If the distance of the CAV with the advantage from the end of the conflict zone is smaller than its stop distance, $d^A_{end} < d^A_{stop}$, it will stop outside of the conflict zone even if it decelerates at a rate of smaller than or equal to a_{brake}.*

If the distance of the CAV with the advantage from the end of the conflict zone is greater than its stop distance, $d^A_{end} > d^A_{stop}$, it may stop inside the conflict zone if it decelerates. In this case, the CAV with disadvantage will be notified after ρ milliseconds in the worst-case scenario. If the CAV with disadvantage accelerates at a rate of smaller than or equal to a_{ACC} during this time interval (ρ) and then decelerates at a rate of a_{brake}, its stop distance will be equal to d^D_{stop} (Eq. (6.2)) and it will not enter the conflict zone and no accident will happen. For scenarios where the scenario-dependent distance is not zero, $d^A_{extra} > 0$ (same lane and merge), the paths of the CAVs overlap and if the CAV with advantage decelerates, it will allow the CAV with disadvantage to travel through the conflict zone by d^A_{extra} and still be safe. As a result, the required safe distance is $d^D_{stop} - d^A_{extra}$.

Next, we study a few case studies and show how the safe RSS distance is calculated for each scenario.

6.1.1 Same Lane

Let us consider a scenario where two CAVs are driving in the same lane as depicted in Figure 6.1. The front CAV has the advantage since its arrival time at the conflict

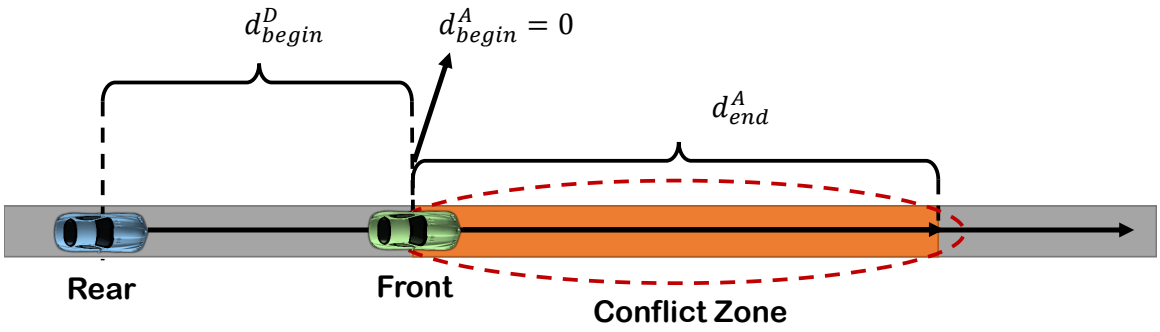


Figure 6.1: An Example of a Same Lane Scenario with Two CAVs. The Front CAV Has the Advantage and Its Distance from the Conflict Zone Is Zero. The Conflict Zone Is Highlighted in Orange.

zone is smaller than the rear CAV. Since the paths of the front CAV overlaps with the path of rear CAV, $d^A_{extra} = d^A_{stop}$, which means the front CAV travels d^A_{stop} meters inside the conflict zone before a complete stop and the rear CAV has d^A_{stop} meters more to stop. According to Equation (6.3), the required safe distance for the rear CAV (d^D_{SAFE}) to maintain from the conflict zone/front CAV is:

$$d^D_{SAFE} = d^D_{stop} - d^A_{stop} + \frac{VL_D + VL_A}{2}$$

d^D_{stop} and d^A_{stop} are calculated according to Equation6.1 and 6.2.

6.1.2 Intersection

Now, let us consider a scenario where two CAVs approach an intersection and their future path crosses inside the intersection area as it is depicted in Figure 6.2. We assume the arrival time of the green CAV to be earlier than the blue CAV and

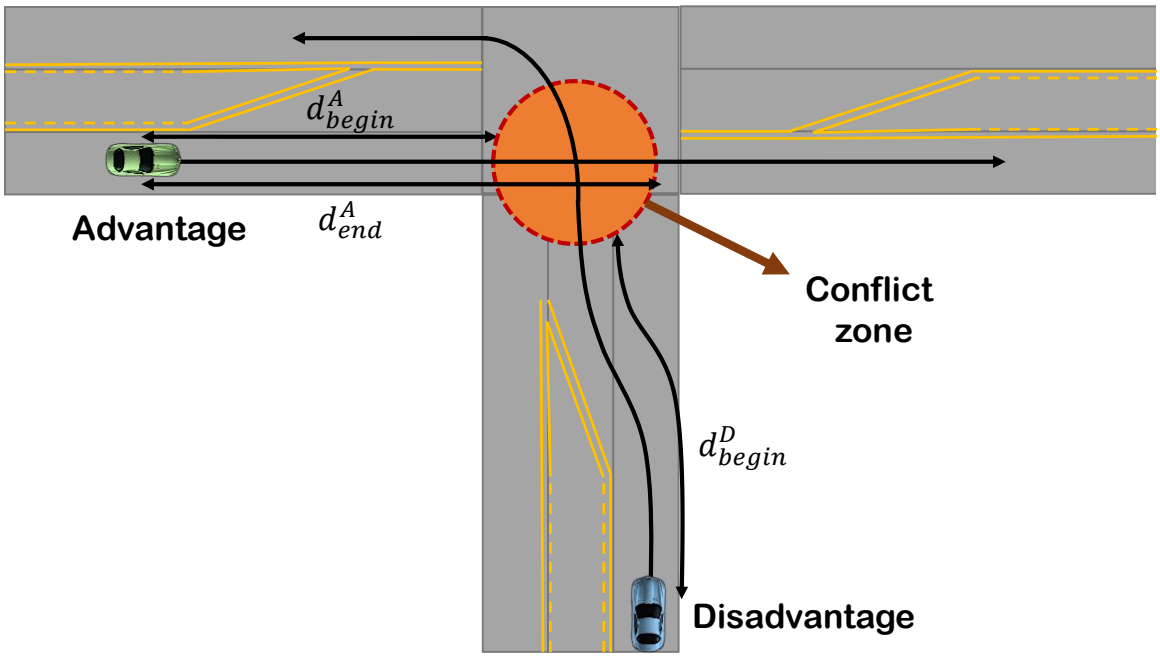


Figure 6.2: A Scenario with Two CAVs Approaching an Intersection and Their Future Path Intersect. It Is Safe to Enter the Conflict Zone after the Other CAV Leaves Conflict Zone.

therefore, it has the advantage. If the green CAV stops anywhere inside the conflict zone, it's not safe for the blue CAV to enter the conflict zone. Therefore, the scenario-dependant distance is zero, $d^A_{extra} = 0$. As a result, we have:

$$d^D_{SAFE} = \begin{cases} d^D_{stop} + \frac{VL_A + VL_D}{2} & \text{if} \quad d^A_{end} \geq d^A_{stop} \\ 0 & \text{otherwise} \end{cases}$$

If the distance of the green CAV from the end of the conflict zone is smaller than its stop distance, even in the worst-case (if it decelerates at the maximum rate), it

will stop outside the conflict zone and does not cause a conflict for the blue CAV. In this case, there will be no conflicts and $d^D_{SAFE} = 0$.

6.1.3 Merge

Next, we consider a merge scenario where two CAVs merge into the same lane as it is shown in Figure 6.3. Without loss of generality, we assume one of the CAVs

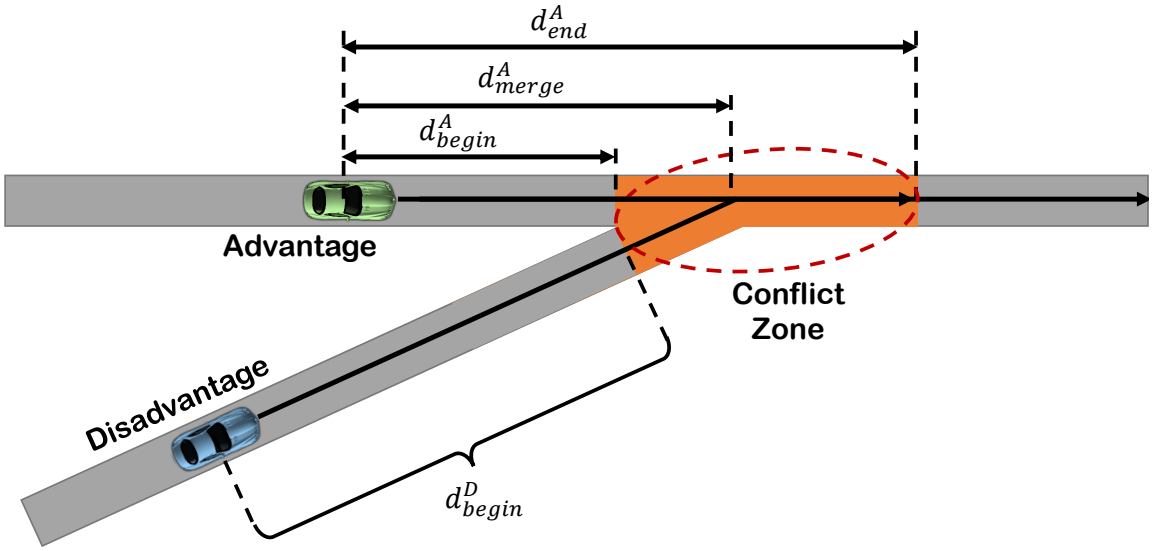


Figure 6.3: A Scenario Where Two CAVs Are Expected to Be Merged into the Same Lane. The CAV with Earlier Arrival Time Has the Advantage.

(green one) has the advantage and the other CAV has disadvantage respectively. In this scenario, the scenario-dependant distance is $d^A_{extra} = \min(0, d^A_{stop} - d^A_{merge})$, where d^A_{merge} is the distance of the CAV with advantage from the merging point, which is indicated in Figure 6.3. As a result, the blue CAV must maintain a minimum distance of

$$d^D_{SAFE} = d^D_{stop} - \min(0, d^A_{stop} - d^A_{merge}) + \frac{VL_D + VL_A}{2}$$

from the conflict zone. Note that once the blue CAV reaches the merge point, the d^A_{extra} is changed. The lateral case in the original RSS rules (two CAVs driving on

adjacent lanes) can be modeled like a merging case. If any of the CAVs attempts to merge into the other CAV's lane, it is only allowed if the created conflict zone is far enough from the other CAV i.e. at least d_{max}.

6.2 Proposed Cooperative Driving Approach

In this section, we first present the algorithm that runs on each CAV assuming no deadlock situation happens. In the next section, we explain the deadlock resolution algorithm.

6.2.1 Main Algorithm

Given the initial position and final destination of a CAV are known, the motionPlanner uses the world's map to determine the shortest path (R) that connects CAV's current position to the destination. We assume that at least one feasible path exists that connects CAV's current location to its destination. The map, $M(N, E)$, is a directed graph where N is the set of nodes or waypoints and E is the set of edges or connections between waypoints. Each edge of the map graph has a weight, w, which indicates the maximum velocity for that segment of the road. In our algorithm, we assume that the ego CAV's computation time and communication time are bounded by T.

In a periodic manner, each CAV broadcasts its ID, position, velocity, timestamp, and its future path (FP), which is an array of x-y coordinates. We assume that all CAVs synchronize their clock using GPS so that timestamps are captured with clocks that have almost the same notion of time. When the CAV receives the information of other CAVs, it checks if their paths intersect or the distance between their paths is less than a threshold. If so, the CAV computes a set of conflict zones (C). For each conflict zone, the CAV determines which vehicle has the advantage to enter the conflict zone

first based on who is expected to reach the conflicting zone first. To detect possible deadlocks, the CAV computes a graph called Partial Dependency Graph (PDG), which represents the dependency among other CAVs and itself (who should yield to who over a conflict zone). Next, the CAV broadcasts the computed PDG, and after receiving other CAVs PDG, it constructs the Complete Dependency Graph (CDG) to detect and resolve possible deadlocks. Finally, if the CAV has disadvantage over a conflict zone, it computes a safe velocity so that it always maintains a safe distance from that conflict zone. Based on the determined velocity, the weight of some of the edges are updated to reflect the presence of other CAVs and to make sure a safe distance is always maintained from the conflict zone. Then, the motion planner runs the shortest path algorithm again to check if a shorter path exists that does not cause a new conflict. Finally, the motion controller uses a subset of future waypoints and velocities of corresponding edges to determine the desired velocity and control inputs (steering angle and acceleration) for the CAV. Alg. 8 shows the pseudo-code of our algorithm that is executed on each CAV. To have a better understanding of our algorithm, we have depicted different components of our approach and their relationship in Figure 6.4. Next, we will focus on explaining the functionality of each component of the algorithm.

6.2.2 Future Path Computation

Each CAV broadcasts its ID, position (x, y), velocity (v), and the corresponding timestamp (ts) as well as its future path $((x_1, y_1), ..., (x_n, y_n))$. Assuming the CAV's motion controller is tuned to have a short settling time, the CAV will track its path with a negligible error. As a result, we represent the future position of the CAV with a subset of its expected route (R). Given $R \subset M(N, E)$ is the route of the CAV, the future path of the CAV, $FP \subset R$ is calculated as follows which consists from n

Algorithm 8: CAVs algorithm

```
path = motionPlanner(map);
while has not reached the destination do
    FP = compute_future_path();
    CAV_info = [x, y, v, ts, FP, ID];
    broadcast(CAV_info);
    others_info = receive_other_CAVs_info();
    for each member of other_CAVs_info do
        [C, PDG] = find_conflict_zones(CAV_info, others_info);
    end
    broadcast(PDG);
    others_PDG = receive_other_PDGs();
    CDG = construct_CDG(PDG, other_PDGs);
    C = deadlock_resolution(C, CDG);
    if ego CAV has disadvantage over a conflict zone then
        [FP, velocity] = motion_planner(C, Map);
    end
    motionController(FP, Velocity);
end
```

points:

$$FP = \left\{ (x_i, y_i) \in R \;\middle|\; \left(\sum_{i=2}^{i=n} \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2} \right) < d_{max} \right\} \tag{6.4}$$

where d_{max} is the fixed length of the future path calculated as:

$$d_{max} = v_{max}(\rho + t_b) \tag{6.5}$$

ρ represents the worst-case end-to-end delay from one CAV capturing its information and broadcasting it, to another CAV's actuation based on the received information (see Figure 6.5) and t_b is the worst-case brake time which can be calculated as $t_b = \frac{v_{max}}{|a_{brake}|}$. Figure 6.5 shows the execution profile of our algorithm on two CAVs (i and j). Let us assume that CAVs i and j have a conflict and CAV i (top) has the advantage. If

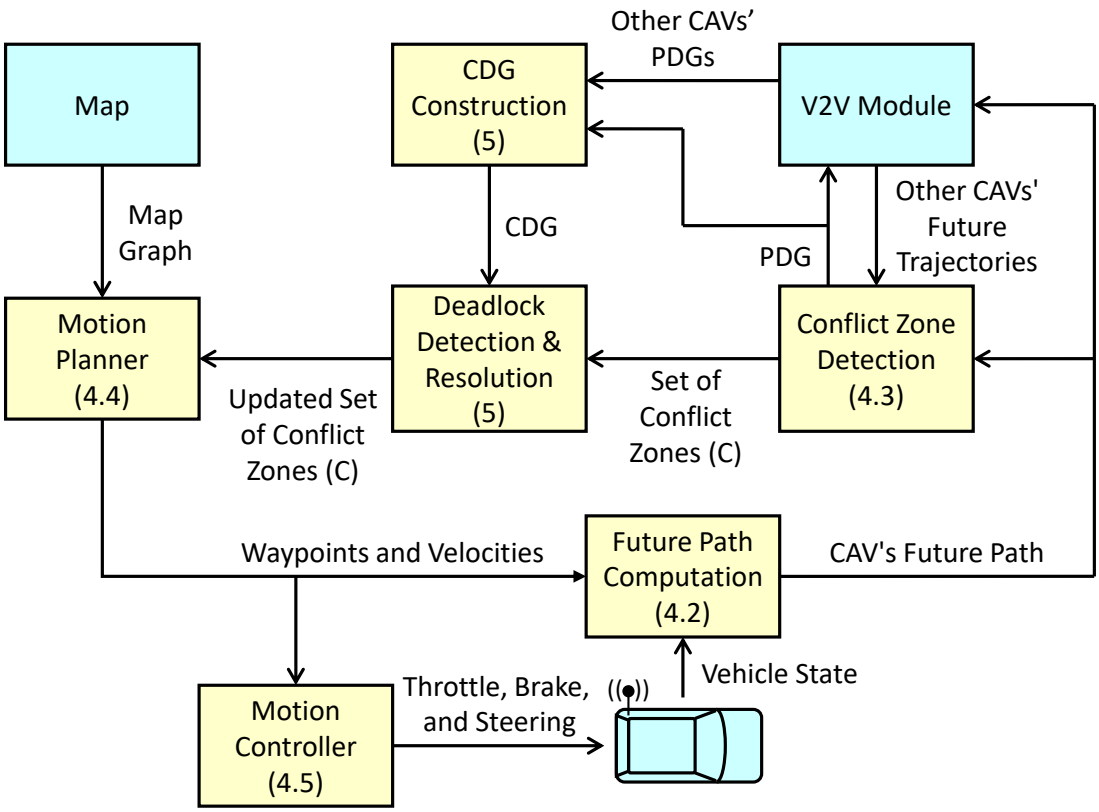


Figure 6.4: Overview of Our Approach. Details of Each Component –Except V2X Module and Map– Are Explained Later.

CAV i slows down due to any reason right after sensing and broadcasting its info, the CAV j will not be notified until receiving the next broadcast. As a result, the worst-case end-to-end delay (ρ) is bounded by $2T$ as depicted in Figure 6.5. By computing the d_{max} based on the worst-case info sharing delay and worst-case braking time, we ensure that for the first time that two CAVs detect that they have a conflict, the CAV with the disadvantage have enough distance to safely stop without entering the conflict zone, even in the worst-case scenario.

6.2.3 Conflict Zone Detection

Despite existing approaches that use fixed conflict zones, we use CAV's expected trajectory to detect a conflict zone. As mentioned before, CAVs' future paths (FP) are used to represents their expected future position. First, CAVs computes the

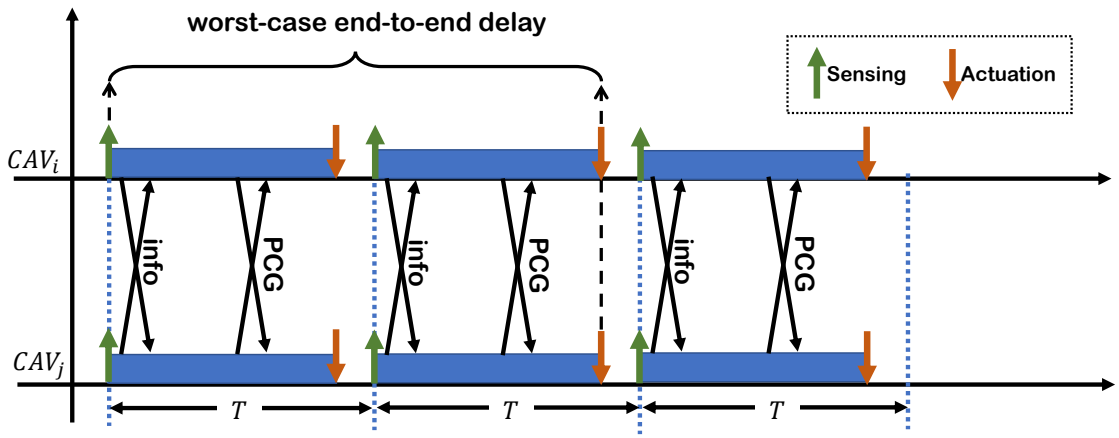


Figure 6.5: CAVs Perform Computation and Communication in a Synchronized Manner. The Worst-case Sensing to Actuation Delay Corresponds to the Case That CAV_i Breaks down Right after Sensing.

distance between the mid point of edges on their path. All contiguous edges that have a distance less than d_{th} are considered to be a part of the same conflict zone. Two CAVs may have multiple conflicts on their path as depicted in Fig. 6.6. Each

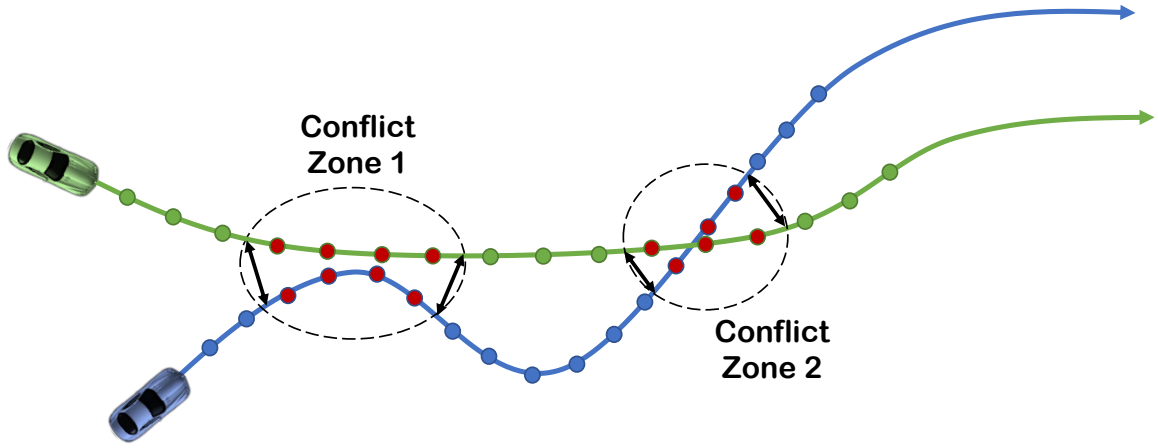


Figure 6.6: An Example of Two CAVs with Arbitrary Paths and Two Conflict Zones. The Conflict Zone Includes Parts of the CAV Path (Waypoints) Where the Distance Between Paths of CAVs Is Less than a Threshold.

conflict zone C_i is a data structure that includes waypoints that are inside the conflict zones, distance of CAVs from the the beginning and end of the conflict zone, their expected arrival time at the conflict zone (Equation 6.6) and the ID of the CAV that has the advantage. We compute the arrival time assuming the CAV drives at a

constant velocity.

$$TOA_i = \frac{d^i_{begin}}{v_i} \tag{6.6}$$

where d^i_{begin} is the distance of the CAV i from the conflict zone and v_i is the velocity of the CAV i. Since the algorithm is executed periodically (every T ms), the value of TOA_i is updated as the velocity of the CAV changes. If a CAV is stopped inside a conflict zone, its arrival time is set to zero. By default, the CAV with the earliest arrival time will have the advantage unless it is changed to resolve a deadlock (explained in the next section) or the other CAV has a priority (e.g. opposite direction). If two CAVs have the same arrival time, the CAV with the lower ID will have the advantage to break the tie. In addition, if the difference between the arrival times of two CAVs is within the accuracy of the clock synchronization (± 10 nanoseconds for GPS), they use CAVs ID to determine who has the advantage.

6.2.4 Motion Planner

If a CAV has disadvantage over a conflict zone, it first checks if an alternative path exists such that it avoids all the conflicts. If such a path exists, the CAV selects that path and if not, the CAV calculates a safe velocity (v_{SAFE}) to be maintained so that the CAV is always safe. The safe velocity, v_{SAFE}, is determined based on the minimum safe distance that the ego CAV must maintain from the conflict zone given that other CAV –which has the advantage– may slow down at any point in time and stop inside the conflict zone.

Maximum Safe Velocity: For each segment of the road that has a distance of d_C from the conflict zone, the maximum safe velocity is computed using Equation 6.7.

$$v_{SAFE} = \frac{-(2\rho a_{ACC} + 2|a_{brake}|) + \sqrt{\Delta}}{2} \tag{6.7}$$

where $\Delta = 4(a_{brake}^2 + 2a_{ACC}\rho a_{brake} - a_{ACC}\rho^2 a_{brake} - 2d_C|a_{brake}|$. Equation 6.7 is determined by solving Equation 6.2 for v_D when the distance from the conflict zone is d_C. d_C can be calculated using Equation 6.10. Equation (6.7) ensures that the CAV with disadvantage has always a minimum distance of d_{SAFE}^D from the conflict zone.

Once the safe velocity is determined for each conflict zone (C_i), the motion planner updates weights of the map $M(N, E)$, to account for the presence of other CAVs and generates safe velocities for the motion controller. To account for the presence of other CAVs, the motion planner determines, U the set of all edges (e_i) that are connected to waypoints that are on the future path of other CAVs

$$U = \{e_i \in E | e_i \in connected(FP)\} \tag{6.8}$$

where $connected(FP)$ is the set of all edges that are connected to waypoints in the set FP. To account for the safe RSS distance, the motion planner determines U_D, the set of all edges that are connected to the waypoints that are on the future path of the CAV with disadvantage (FP_D) and are either a member of the conflict zone set (C) or within the safe distance (d_{SAFE}^D) of the conflict zone.

$$U_D = \{e_i \in E | e_i \in connected(FP_D^C)\} \tag{6.9}$$

where $connected(FP_D^C)$ is the set of all edges that are connected to waypoints in the set FP_D^C. Figure 6.7 shows a merge scenario and CAV's future paths. Weights of all edges connected to nodes that are on the path of the CAV with advantage (depicted in green) and all edges that are on the path of the ego CAV and are either within the safe distance of or inside the conflict zone are updated. The set U_D and U are highlighted on the path of CAVs. The subset of future point, FP_D^C, is determined as:

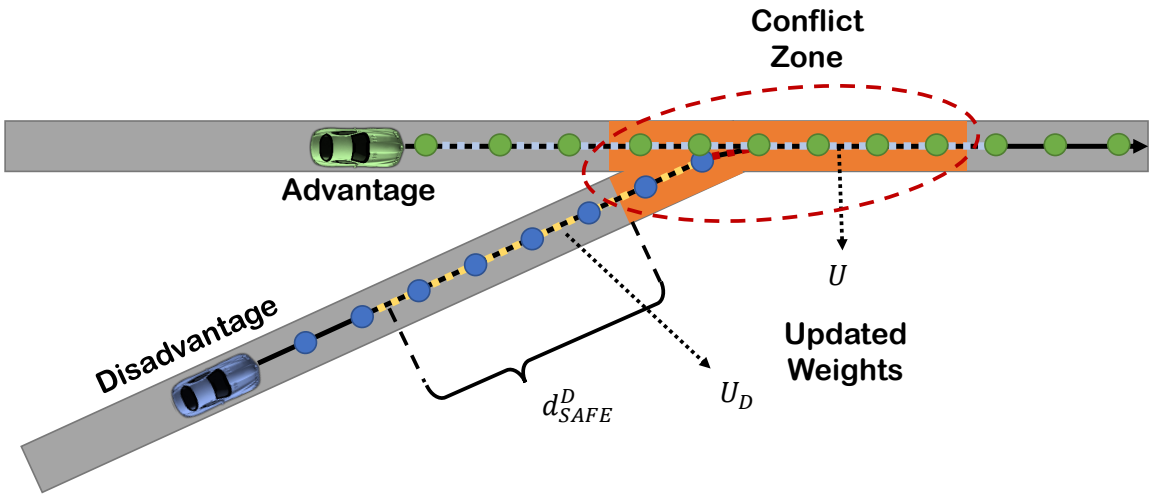


Figure 6.7: Weights of the Edges on the Path of the Other CAV and Edges on Path of the Ego CAV Are Updated to Account for the Presence of Other CAVs as Well as the Conflict Zone and the Required Safe Distance.

$$FP^C_D = \{n_i \in N | n_i \in FP_D \; and \; n_i \in C \; or \; n_i \in within(C_j)\}$$

$within(C_j)$ is the set of all waypoints that where their distance from the conflict zone j is less than d_{SAFE}. To calculate the distance between two waypoints, we use the following recursive equation:

$$distance = \sum_{i=2}^{N} \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2} \tag{6.10}$$

where N is the number of waypoints including the first and last waypoints. Finally, weights of each edge in set U and U_D are updated based on their distance from the conflict zone using Equation 6.7:

$$w_i = \frac{l}{v^i_{SAFE}} \tag{6.11}$$

where i refers to each segment of the road, l is the length of the corresponding edge and v^i_{SAFE} is the safe velocity calculated for each segment of the road (edge). Since the weight of an edge may be updated multiple times –as it may be involved in more

than one conflict–, the maximum weight is considered (the slowest safe velocity) for an edge. If the safe velocity (v_{SAFE} is equal to zero, instead of infinity, the weight is set to be a large constant number. After updating the weights, the motion planner uses the Dijkstra algorithm to find the shortest path to the destination. The summation of weights ($\sum w_i$) from the source to the destination corresponds to the travel duration.

6.2.5 Motion Controller

We assume the following vehicle dynamics for the CAV.

$$\begin{cases} \dot{x} = v\cos(\phi) \\ \dot{y} = v\sin(\phi) \\ \dot{\phi} = \frac{v}{L}\tan(\psi) \\ \dot{v} = a \end{cases} \tag{6.12}$$

where x and y represents the position of the ego CAV in Cartesian coordinates, ϕ is the CAV's heading angle from the x-axis, v and a are linear velocity and acceleration of the CAV respectively, L is CAV's wheelbase distance and ψ is steering angle of front wheels with respect to the heading of the CAV. In order to make the model more realistic, we consider an upper bound and a lower bound on the acceleration rate and steering angle of a CAV as: $a \in [a_{min}, a_{max}]$ and $\psi \in [\psi_{min}, \psi_{max}]$ where a_{max} and a_{min} are the maximum acceleration and deceleration rates and ψ_{max} and ψ_{min} are the maximum and minimum steering angles of the vehicle.

The motion controller uses the future waypoints and safe velocities to calculate the reference heading angle θ_{ref} and the safe velocity v_{ref} for the CAV. For the desired heading angle (θ_{ref}), the motion controller selects a look-ahead point similar to the pure pursuit algorithm Coulter (1992) and calculates the bearing angle from

its current location (x, y) to the look-ahead point:

$$\theta_{ref} = atan2(x - x_l, y - y_l) \tag{6.13}$$

where x_l and y_l correspond to the x-y coordinate of the look-ahead point. We assume that each vehicle has an initial desired velocity of v_0 and never drives faster than that. The motion controller uses the weight of the next edge to determine the reference velocity ($v_{ref} = \frac{d_i}{w_i}$). If the calculated velocity is greater than CAV's initial desired velocity (v_0), it sets the reference velocity to be v_0. If the reference velocity is close to zero, ($v < \epsilon$), it is set to zero. Once the reference heading and velocity are calculated, they are passed to two Proportional Integral Derivative (PID) controllers to calculate the steering angle and acceleration for the vehicle:

$$\begin{cases} \psi = k_P e_\theta + k_I \int e_\theta + k_D \dot{e}_\theta \\ a = k'_P e_v + k'_I \int e_v + k'_D \dot{e}_v \end{cases} \tag{6.14}$$

where k_P, k_I, k_D and k'_P, k'_I, k'_D are constant (controller gains) that are tuned to achieve a fast response while the overshoot is small (short settling time), $e_\theta = \theta_r - \theta$ and $e_v = v_{SAFE} - v$, and $\dot{e}_v$ and $\dot{e}_\theta$ are the derivative of e_v and e_θ, respectively.

6.3 Deadlock Detection and Resolution

In order to detect and resolve deadlocks, all CAVs create a directed graph called the *dependency graph*. Nodes of the dependency graph are vehicle IDs and edges represent that if a CAV is yielding to another CAV over a conflict zone. There will be a directed edge from node V_i to node V_j if CAV V_i is yielding to the CAV V_j over a conflict zone. Since a CAV determines only the conflicts between itself and other CAVs –and not the conflicts between other CAVs, the constructed dependency

graph is not complete. We refer to the dependency graph of each CAV as the "partial dependency graph" or PDG. To compute the complete graph, each CAV broadcasts its PDG to inform other CAVs about its conflict zones with other CAVs and to receive other CAVs' PDG. From the received PDGs of other CAVs and the PDG of the ego CAV, the complete dependency graph (CDG) is constructed. To build the CDG, the PDG is incrementally updated by adding nodes and edges for each received PDG. Finally, all edges between two nodes are merged into one. Figure 6.8 shows a scenario with 5 CAVs that have determined their PDG and the final consensual CDG.

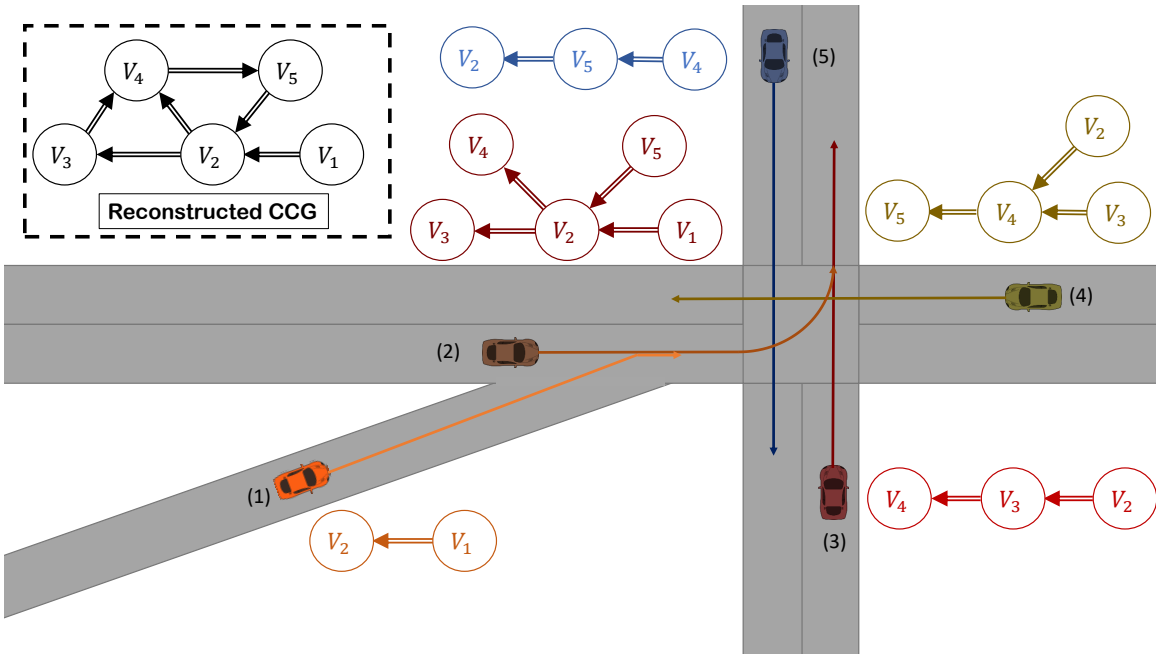


Figure 6.8: Each CAV Determines and Broadcasts Its PDG. After Receiving Other CAVs' PDG, CAVs Construct the CDG and Can Resolve Deadlocks.

After constructing the CDG, each CAV checks if the CDG has a cycle. We use the Depth-First Search (DFS) algorithm to detect cycles. If a deadlock is detected, each CAV calculates a score for each CAV that is involved in the cycle based on its average time of arrival at corresponding conflict zones. If a CAV has m conflicts, its

score is calculated as:

$$S = \frac{\sum_{i=1}^{m} TOA_i}{m}$$

where TOA_i is the time of arrival of the CAV at its ith conflict zone. We select the CAV with the least average time of arrival to have the advantage over all of its conflict zones because on average, it can reach its conflict zone earlier than others. We refer to this CAV as the leader. Once the leader is determined, the direction of all incoming edges to the leader's node is reversed. If two CAVs have the same score, the CAV with the lower ID number will be selected as the leader. Since there can be more than one cycle in a graph, this process is repeated until all cycles are removed.

Lemma 2 *If the CDG has no cycles, then there is no deadlock involving the ego CAV.*

Proof 2 *Once the CDG is modified to be acyclic, there is no path (set of sequential edges) starting at node V_{ego} that eventually loops back to node V_{ego} again, which means the ego CAV never yields to other CAVs that are yielding to the ego CAV and therefore, there is no deadlock involving the ego CAV.*

It takes some time to resolve a deadlock due to the vehicle's dynamic –CAVs cannot change their velocity and expected arrival time instantly. As a result, CAVs may face the same deadline again when they compute the CDG after T. It can be shown that the result of deadlock resolution will be the same (the same CAV will be selected as the leader) until the deadlock is resolved. Since the leader has the least average time of arrival in the first iteration, it does not yield to any other CAV while other CAVs involved in the deadlock slow down to yield to at least one CAV. Therefore, the average time of arrival of the leader will be less than other CAVs in the second iteration and so on.

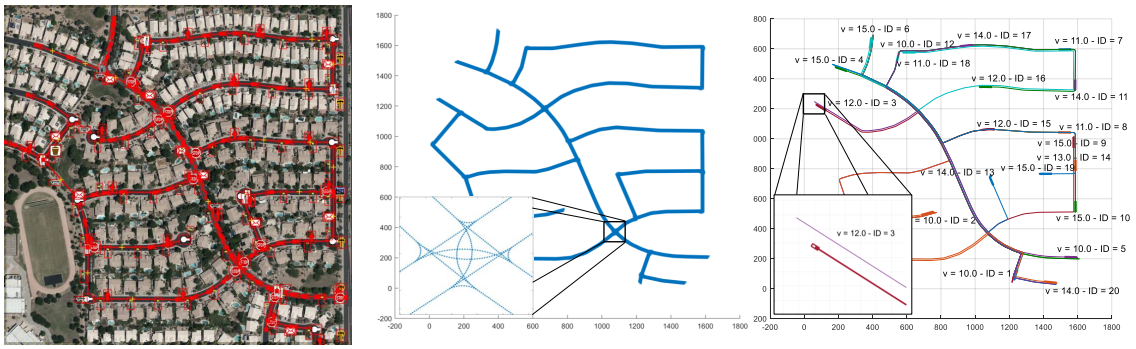


Figure 6.9: A Snapshot of a Map Retrieved from the OpenStreetMap (Left), Its Corresponding Directed Graph in Matlab (Middle) and a Scenario with Randomly Spawned Vehicles on the Map (Right).

6.4 Testbed 4 - City-Wide Traffic Simulation using OpenStreetMap

We evaluated our algorithm on a simulator that is developed in Matlab. We created a tool in Python to automatically extract a desired map from the OpenStreetMap[1] (OSM format) and then generate the world map graph for it. Once the map is generated, a driving scenario is created where initial position and velocity and the destination of CAVs are randomly selected. We used differential equations represented in (3.7) to model vehicle's dynamics. The size of each vehicle is 5x2 m, the lane width is 5 m and the distance between waypoints of the map is 0.5 m. Gains of the controller for both heading and velocity control are $K_P = 5$ and $K_D = 0.1$. Other parameters of the vehicle are listed in Table 6.1. CAVs communication delay

v_{min}	v_{max}	a_{min}	a_{max}	ψ_{min}	ψ_{max}	T	ρ
$0\frac{m}{s}$	$23\frac{m}{s}$	$-8\frac{m}{s^2}$	$5\frac{m}{s^2}$	$-\frac{\pi}{3}$rad	$\frac{\pi}{3}$rad	$0.1s$	$0.2s$

Table 6.1: Parameters of the CAVs for Simulation.

is modeled by queuing the broadcast packets. In Figure 6.9, a randomly generated map from openStreetMap, its corresponding map graph and a random scenario with 20 CAVs are depicted.

[1]https://www.openstreetmap.org/

6.5 Results

We performed three experiment to demo our RSS-based conflict resolution algorithm: 1) Safety evaluation for corner cases, 2) Deadlock detection and resolution and 3) Performance evaluation.

6.5.1 Safety Evaluation

To demonstrate the safety of the proposed algorithm, we created a merge and an intersection scenario where two CAVs have a conflict on their future path as it is depicted in Figure 6.10.

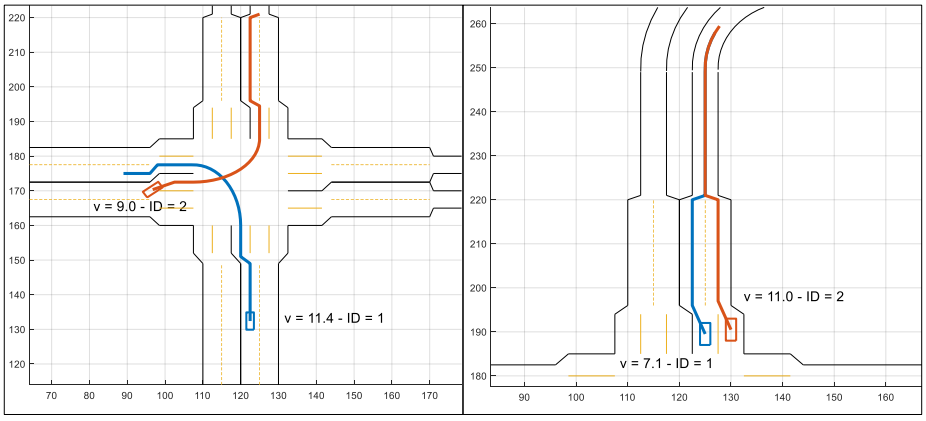


Figure 6.10: An Intersection and a Merge Scenario Are Created. The CAV with Advantage Suddenly Decelerates and Stops.

To verify that CAVs are always safe, we force the CAV with the advantage to suddenly decelerate at different times. We show that no accident will happen regardless of the deceleration time and CAVs maintain a minimum safe distance of 5 meters. Using brute-force testing, the deceleration time of the CAV with the advantage varies in a 30-second interval with a 0.1 s increment that includes critical times that stops inside the conflict zone. Figure 6.11 and 6.12 show the distance between CAVs for the intersection and merge scenarios, respectively. In the intersection scenario, the CAV

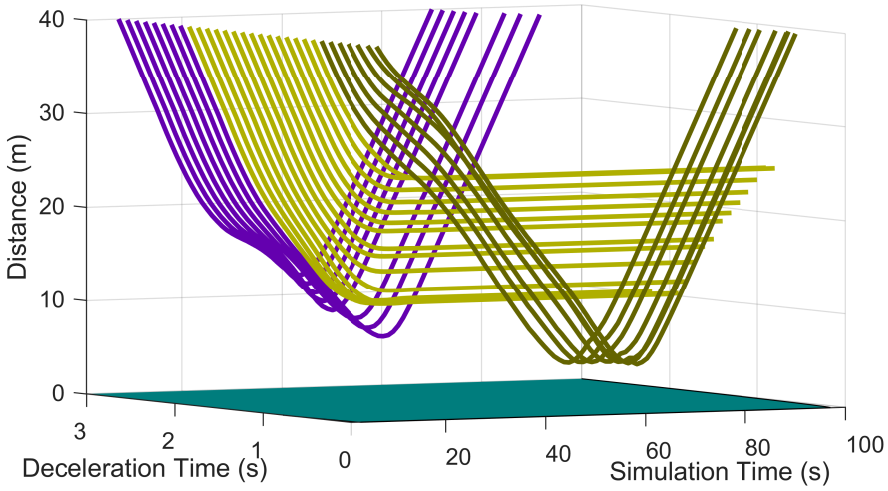


Figure 6.11: Brute-force Evaluation of an Intersection Scenario. CAVs Distance Is Always Greater than a Threshold Regardless of the Deceleration Time of the CAV with the Advantage –if It Stops Before Entering the Conflict Zone (Dark Green), Inside the Conflict Zone (Yellow), or after the Conflict Zone (Blue).

with the advantage may stop before, inside, or after the conflict zone where distances between CAVs are depicted in dark green, yellow, and blue colors, respectively. For cases that the CAV with advantage stops before or after the conflict zone, the CAV with disadvantage continues while in cases that the CAV with advantage stops inside the conflict zone, the CAV with advantage slows down and stops (depicted in yellow). In the merge scenario, the conflict zone moves with the CAV with the advantage after it reaches the merging point. As a result, the CAV with the advantage either stops before the conflict zone or inside it. For cases that the CAV with the advantage stops before the merging point, the CAV with the advantage continues and enters the merge (depicted in dark green) and for the rest of the cases, the CAV with the disadvantage slows down and stops (depicted in yellow).

6.5.2 Deadlock Resolution Demonstration

To evaluate our deadlock detection and resolution approach, we created a deadlock situation at the intersection (Figure 6.13). The right part of Figure 6.13 shows

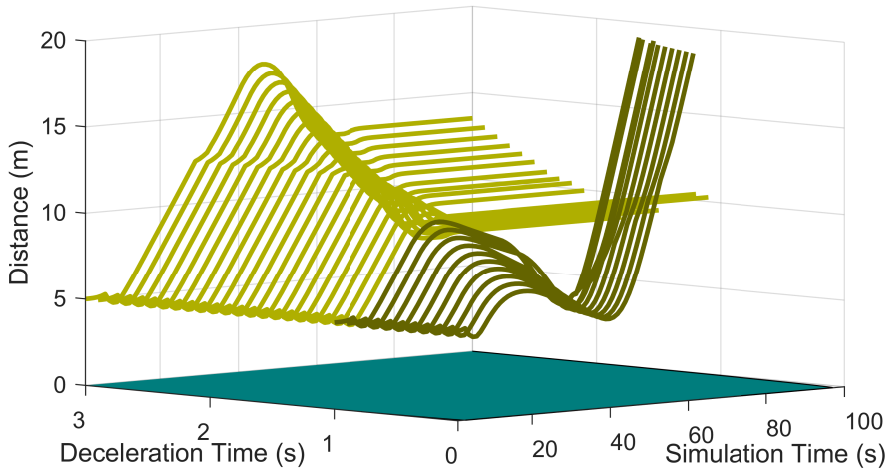


Figure 6.12: Merge Scenario - CAVs Distance Remains Greater than a Threshold for All Cases Where the CAV with Advantage Stops Before Entering the Merge (Yellow) or after the Merge (Dark Green).

the CCG for the scenario. We fixed the paths of CAVs to make a left turn at the

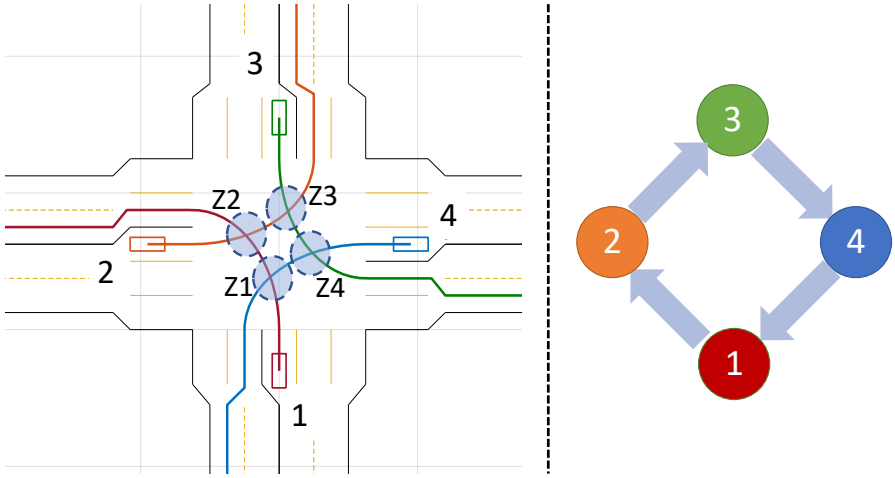


Figure 6.13: A Deadlock Scenario Where 4 CAVs Approach the Intersection with Same Velocity (Left) and the Corresponding CDG (Right).

intersection while having the same distance from the intersection and the same velocity. We simulated CAVs' behavior with and with our deadlock detection. Figure 6.14 shows the velocities of CAVs for both cases. For the case that no deadlock resolution is done, CAVs slow down to yield to other CAVs and eventually stop and will wait forever. For the case with deadlock resolution, CAVs slow down at first but speed up

when their conflict zone is cleared. We can observe that after 7s, all CAVs reach their

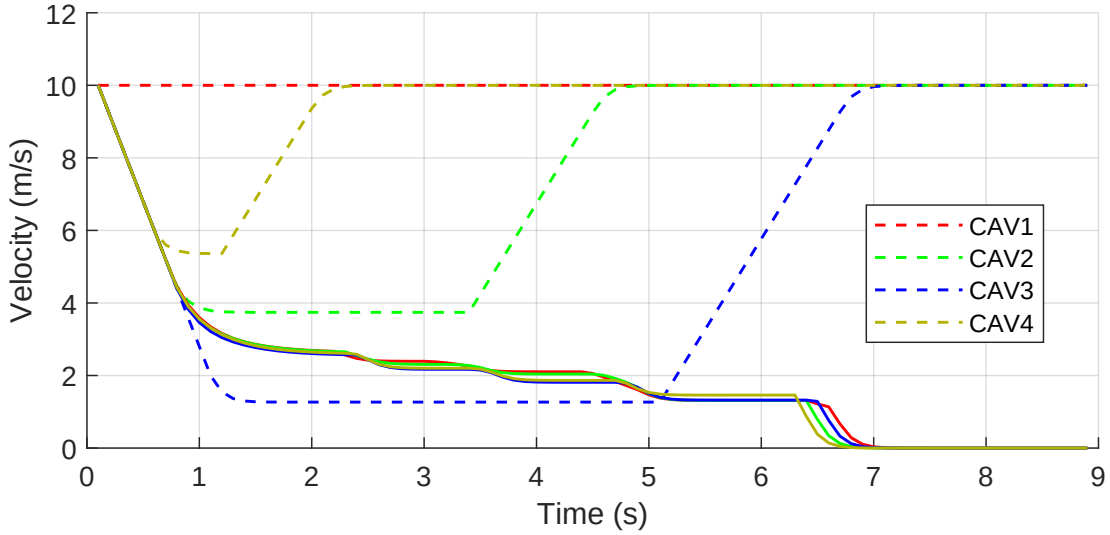


Figure 6.14: Velocity Profiles of CAVs with and Without Deadlock Resolution for the Scenario in Figure 6.13

desired velocity (10m/s) while in no deadlock detection case, their velocity converges to zero.

6.5.3 Efficiency Evaluation

To evaluate the efficiency of our approach, we compared the performance of our approach with the case that vehicles are autonomous but not connected. For the non-connected case, the intersections are managed by stop signs and all other conflicts among CAVs are handled by the AV's perception system e.g. adaptive cruise control (ACC) system. We extracted a map from the OpenStreetMap (Figure 6.9) and simulated three scenarios, i) light traffic with 5 vehicles, ii) moderate traffic with 10 vehicles, and iii) heavy traffic with 20 vehicles being present at the same time. When a vehicle exits the map boundary, a new vehicle is spawned. We measured the average velocities of CAVs and reported them in Table 6.2. We also computed

the fuel consumption of CAVs using the following model Akçelik *et al.* (2012) and reported them in Table 6.2:

$$f = \begin{cases} 0 & \text{if } P_T > 0 \\ \frac{f_i}{3600} + \beta_1 P_T + \beta_2 a P_I & \text{otherwise} \end{cases} \tag{6.15}$$

where $P_T = \min(P_{max}, P_C + P_I)$ is the total tractive power (kW), $P_C = b_1 v + b_2 v^3$ is the cruise component of total power (kW), $P_I = \frac{mav}{1000}$ is the inertia component of the total power (kW), $f_i = 888.8mL/h$ is the instantaneous fuel consumption rate (mL/s), P_{max} is the maximum engine power (kW), m is the vehicle mass, a and v are the instantaneous acceleration and velocity, b_1 is rolling resistant factor (kN), and b_2 is the aerodynamic drag factor (kN/(m/s)^2), β_1 and β_2 are the efficiency factors for non-accelerating and accelerating cases.

	Light Traffic		Moderate Traffic		Heavy Traffic	
	AVs	CAVs	AVs	CAVs	AVs	CAVs
Average Velocity (m/s)	11.21	11.96	10.91	11.83	10.51	11.55
Average Fuel Consumption (mL/s)	1.017	0.485	1.089	0.479	1.271	0.495

Table 6.2: Comparing the Average Velocity and Fuel Consumption of Vehicles When They Navigate Autonomously (Non-connected) and Cooperatively (Connected).

With the help of shared information, CAVs not only drive at higher velocities, they drive smoother than non-connected case because they slow down and stop less frequently and therefore, their fuel consumption is less than the connected case.

Chapter 7

CONTRIBUTIONS

Title: Crossroads: Time-Sensitive Autonomous Intersection Management Technique

Authors: Andert, Edward, Mohammad Khayatian, and Aviral Shrivastava

Contributions: developed an intersection management algorithm for CAVs that is resilient against roundtrip delay

Title: Crossroads+: A Time-aware Approach for Intersection Management of Connected Autonomous Vehicles

Authors: Khayatian, Mohammad, Yingyan Lou, Mohammadreza Mehrabian, and Aviral Shirvastava

Contributions: Develop an algorithm that compensates for the response time of the controller presented in Crossroads work

Title: RIM: Robust Intersection Management for Connected Autonomous Vehicles

Authors: Khayatian, Mohammad, Mohammadreza Mehrabian, and Aviral Shrivastava

Contributions: Developed an algorithm that is robust to bounded model mismatches and external disturbances

Title: A Dependable Detection Mechanism for Intersection Management of Connected Autonomous Vehicles

Authors: Dedinsky, Rachel, Mohammad Khayatian, Mohammadreza Mehrabian, and Aviral Shrivastava

Contributions: Developed an algorithm to detect rogue vehicles at intersection

Title: A Survey on Intersection Management of Connected Autonomous Vehicles
Authors: Khayatian, Mohammad, Mohammadreza Mehrabian, Edward Andert, Rachel Dedinsky, Sarthake Choudhary, Yingyan Lou, and Aviral Shirvastava
Contributions: studied more than 100 papers that are published on intersection management of CAVs and analyzed their pros and cons from different perspectives

Title: R^2IM – Robust and Resilient Intersection Management of Connected Autonomous Vehicles.
Authors: Khayatian, Mohammad, Rachel Dedinsky, Sarthake Choudhary, Mohammadreza Mehrabian, and Aviral Shrivastava
Contributions: Developed an algorithm for robust intersection management of CAVs against rogue vehicles

Title: Cooperative driving of connected autonomous vehicles using responsibility-sensitive safety (RSS) rules.
Authors: Mohammad Khayatian, Mohammadreza Mehrabian, Harshith Allamsetti, Kai-Wei Liu, Po-Yu Huang, Chung-Wei Lin, Aviral Shrivastava
Contributions: Developed a generic V2V algorithm for safe and deadlock-free conflict resolution of connected autonomous vehicle

REFERENCES

Abdelgader, A. and W. Lenan, "The physical layer of the ieee 802.11 p wave communication standard: the specifications and challenges", in "Proceedings of the world congress on engineering and computer science", vol. 2, p. 71 (2014).

Administration, U. F. H., "Statistics on intersection accidents", `https://www.autoaccident.com/statistics-on-intersection-accidents.html`, [Online; accessed 30-May-2019] (2019).

Ahn, H. and D. Del Vecchio, "Safety Verification and Control for Collision Avoidance at Road Intersections", arXiv preprint arXiv:1612.02795 (2016).

Akçelik, R., R. Smit and M. Besley, "Calibrating fuel consumption and emission models for modern vehicles", in "IPENZ transportation group conference, Rotorua, New Zealand", (2012).

Andert, E., M. Khayatian and A. Shrivastava, "Crossroads: Time-sensitive autonomous intersection management technique", in "Proceedings of the 54th Annual Design Automation Conference 2017", p. 50 (ACM, 2017).

Aoki, S. and R. Rajkumar, "A merging protocol for self-driving vehicles", in "2017 ACM/IEEE 8th International Conference on Cyber-Physical Systems (ICCPS)", pp. 219–228 (IEEE, 2017).

Aoki, S. and R. R. Rajkumar, "Dynamic intersections and self-driving vehicles", in "Proceedings of the 9th ACM/IEEE International Conference on Cyber-Physical Systems", pp. 320–330 (IEEE Press, 2018).

Ashtiani, F., S. A. Fayazi and A. Vahidi, "Multi-intersection traffic management for autonomous vehicles via distributed mixed integer linear programming", in "2018 Annual American Control Conference (ACC)", pp. 6341–6346 (IEEE, 2018).

Au, T.-C. and P. Stone, "Motion planning algorithms for autonomous intersection management.", in "Bridging the gap between task and motion planning", (2010).

AutoSIM, "Driving simulators and simulator software", `https://www.autosim.no/`, [Online; accessed 03-June-2019] (2019).

Azimi, R., G. Bhatia, R. Rajkumar and P. Mudalige, "Intersection management using vehicular networks", Tech. rep., SAE Technical Paper (2012).

Azimi, R., G. Bhatia, R. Rajkumar and P. Mudalige, "V2v-intersection management at roundabouts", SAE International Journal of Passenger Cars-Mechanical Systems **6**, 2013-01-0722, 681–690 (2013a).

Azimi, R., G. Bhatia, R. Rajkumar and P. Mudalige, "Ballroom intersection protocol: Synchronous autonomous driving at intersections", in "2015 IEEE 21st International Conference on Embedded and Real-Time Computing Systems and Applications", pp. 167–175 (IEEE, 2015).

Azimi, R., G. Bhatia, R. R. Rajkumar and P. Mudalige, "Stip: Spatio-temporal intersection protocols for autonomous vehicles", in "ICCPS'14: ACM/IEEE 5th International Conference on Cyber-Physical Systems (with CPS Week 2014)", pp. 1–12 (IEEE Computer Society, 2014).

Azimi, S. R., G. Bhatia, R. R. Rajkumar and P. Mudalige, "Reliable intersection protocols using vehicular networks", in "Proceedings of the ACM/IEEE 4th International Conference on Cyber-Physical Systems", pp. 1–10 (ACM, 2013b).

Bailey, T. and H. Durrant-Whyte, "Simultaneous localization and mapping (slam): Part ii", IEEE Robotics & Automation Magazine **13**, 3, 108–117 (2006).

Bashiri, M. and C. H. Fleming, "A platoon-based intersection management system for autonomous vehicles", in "Intelligent Vehicles Symposium (IV), 2017 IEEE", pp. 667–672 (IEEE, 2017).

Bashiri, M., H. Jafarzadeh and C. H. Fleming, "Paim: Platoon-based autonomous intersection management", in "2018 21st International Conference on Intelligent Transportation Systems (ITSC)", pp. 374–380 (IEEE, 2018).

Beaver, L. E., B. Chalaki, A. Mahbub, L. Zhao, R. Zayas and A. A. Malikopoulos, "Demonstration of a time-efficient mobility system using a scaled smart city", arXiv preprint arXiv:1903.01632 (2019).

Behrisch, M., L. Bieker, J. Erdmann and D. Krajzewicz, "Sumo–simulation of urban mobility: an overview", in "Proceedings of SIMUL 2011, The Third International Conference on Advances in System Simulation", (ThinkMind, 2011).

Bekris, K. E., K. I. Tsianos and L. E. Kavraki, "A decentralized planner that guarantees the safety of communicating vehicles with complex dynamics that replan online", in "2007 IEEE/RSJ International Conference on Intelligent Robots and Systems", pp. 3784–3790 (IEEE, 2007).

Belkhouche, F., "Collaboration and optimal conflict resolution at an unsignalized intersection", IEEE Transactions on Intelligent Transportation Systems (2018).

Bentjen, K. C., "Mitigating the effects of cyber attacks and human control in an autonomous intersection", Tech. rep., Air Force Institute Of Technology Wright-Patterson AFB OH (2018).

Bento, L. C., R. Parafita and U. Nunes, "Intelligent traffic management at intersections supported by v2v and v2i communications", in "2012 15th International IEEE Conference on Intelligent Transportation Systems", pp. 1495–1502 (IEEE, 2012).

Bian, Y., S. E. Li, W. Ren, J. Wang, K. Li and H. Liu, "Cooperation of multiple connected vehicles at unsignalized intersections: Distributed observation, optimization, and control", IEEE Transactions on Industrial Electronics (2019).

Bichiou, Y. and H. A. Rakha, "Developing an optimal intersection control system for automated connected vehicles", IEEE Transactions on Intelligent Transportation Systems **20**, 5, 1908–1916 (2018).

Briefs, U., "Mcity grand opening", Research Review **46**, 3 (2015).

Bruni, L., A. Colombo and D. Del Vecchio, "Robust multi-agent collision avoidance through scheduling", in "52nd IEEE Conference on Decision and Control", pp. 3944–3950 (IEEE, 2013).

Buckman, N., A. Pierson, W. Schwarting, S. Karaman and D. Rus, "Sharing is caring: Socially-compliant autonomous intersection negotiation", in "2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)", pp. 6136–6143 (IEEE, 2019).

Center, G. A., "Simulation of urban mobility (sumo)", `http://sumo.sourceforge.net/`, [Online; accessed 03-June-2019] (2019).

Chattopadhyay, S. and A. Roychoudhury, "Unified Cache Modeling for WCET Analysis and Layout Optimizations", in "Real-Time Systems Symposium, 2009, RTSS 2009. 30th IEEE", pp. 47–56 (IEEE, 2009).

Checkoway, S., D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, T. Kohno *et al.*, "Comprehensive experimental analyses of automotive attack surfaces.", in "USENIX Security Symposium", vol. 4, pp. 447–462 (San Francisco, 2011).

Chen, L. and C. Englund, "Cooperative intersection management: A survey", IEEE Transactions on Intelligent Transportation Systems **17**, 2, 570–586 (2015).

Chen, Q., D. Jiang and L. Delgrossi, "Ieee 1609.4 dsrc multi-channel operations and its implications on vehicle safety communications", in "Vehicular Networking Conference (VNC), 2009 IEEE", pp. 1–8 (IEEE, 2009).

Chen, Q. A., Y. Yin, Y. Feng, Z. M. Mao and H. X. Liu, "Exposing congestion attack on emerging connected vehicle based traffic signal control", in "Network and Distributed Systems Security (NDSS) Symposium 2018", (2018).

Choi, M., A. Rubenecia and H. H. Choi, "Reservation-based traffic management for autonomous intersection crossing", International Journal of Distributed Sensor Networks **15**, 12, 1550147719895956 (2019).

CNBC, "California now allows driverless truck and cargo van testing on public roads", `https://tinyurl.com/wkoqpj4`, [Online; accessed 16-March-2020] (2020).

Colombo, A. and D. Del Vecchio, "Efficient algorithms for collision avoidance at intersections", in "Proceedings of the 15th ACM international conference on Hybrid Systems: Computation and Control", pp. 145–154 (ACM, 2012).

Cosgun, A., L. Ma, J. Chiu, J. Huang, M. Demir, A. M. Anon, T. Lian, H. Tafish and S. Al-Stouhi, "Towards full automated drive in urban environments: A demonstration in gomentum station, california", in "2017 IEEE Intelligent Vehicles Symposium (IV)", pp. 1811–1818 (IEEE, 2017).

Coulter, R. C., "Implementation of the pure pursuit path tracking algorithm", Tech. rep., Carnegie-Mellon UNIV Pittsburgh PA Robotics INST (1992).

Dedinsky, R., M. Khayatian, M. Mehrabian and A. Shrivastava, "A dependable detection mechanism for intersection management of connected autonomous vehicles (interactive presentation)", in "Workshop on Autonomous Systems Design (ASD 2019)", (Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2019).

Douceur, J. R., "The sybil attack", in "International workshop on peer-to-peer systems", pp. 251–260 (Springer, 2002).

Dresner, K. and P. Stone, "Mitigating catastrophic failure at intersections of autonomous vehicles", in "Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 3", pp. 1393–1396 (International Foundation for Autonomous Agents and Multiagent Systems, 2008a).

Dresner, K. and P. Stone, "A multiagent approach to autonomous intersection management", Journal of artificial intelligence research **31**, 591–656 (2008b).

Dresner, K. M. and P. Stone, "Sharing the road: Autonomous vehicles meet human drivers.", in "IJCAI", vol. 7, pp. 1263–1268 (2007).

Durrant-Whyte, H. and T. Bailey, "Simultaneous localization and mapping: part i", IEEE robotics & automation magazine **13**, 2, 99–110 (2006).

Elhadef, M., "An adaptable invanets-based intersection traffic control algorithm", in "Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM), 2015 IEEE International Conference on", pp. 2387–2392 (IEEE, 2015).

Elhenawy, M., A. A. Elbery, A. A. Hassan and H. A. Rakha, "An intersection game-theory-based traffic control algorithm in a connected vehicle environment", in "Intelligent Transportation Systems (ITSC), 2015 IEEE 18th International Conference on", pp. 343–347 (IEEE, 2015).

Emami, P., M. Pourmehrab, M. Martin-Gasulla, S. Ranka and L. Elefteriadou, "A comparison of intelligent signalized intersection controllers under mixed traffic", in "2018 21st International Conference on Intelligent Transportation Systems (ITSC)", pp. 341–348 (IEEE, 2018).

Erdmann, M. and T. Lozano-Perez, "On multiple moving objects", Algorithmica **2**, 1-4, 477 (1987).

Fajardo, D., T.-C. Au, S. Waller, P. Stone and D. Yang, "Automated intersection control: Performance of future innovation versus current traffic signal control", Transportation Research Record: Journal of the Transportation Research Board **0**, 2259, 223–232 (2011).

Farhadi, M. *et al.*, "A novel design of adaptive and hierarchical convolutional neural networks using partial reconfiguration on fpga", in "High Performance Extreme Computing Conference", (IEEE, 2019).

Fayazi, S. A. and A. Vahidi, "Vehicle-in-the-loop (vil) verification of a smart city intersection control scheme for autonomous vehicles", in "2017 IEEE Conference on Control Technology and Applications (CCTA)", pp. 1575–1580 (IEEE, 2017).

Fayazi, S. A. and A. Vahidi, "Mixed-integer linear programming for optimal scheduling of autonomous vehicle intersection crossing", IEEE Transactions on Intelligent Vehicles **3**, 3, 287–299 (2018).

Fayazi, S. A., A. Vahidi and A. Luckow, "Optimal scheduling of autonomous vehicle arrivals at intelligent intersections via milp", in "American Control Conference (ACC), 2017", pp. 4920–4925 (IEEE, 2017).

Feng, Y., C. Yu and H. X. Liu, "Spatiotemporal intersection control in a connected and automated vehicle environment", Transportation Research Part C: Emerging Technologies **89**, 364–383 (2018).

Filocamo, B., J. A. Ruiz and M. A. Sotelo, "Efficient management of road intersections for automated vehicles—the frfp system applied to the various types of intersections and roundabouts", Applied Sciences **10**, 1, 316 (2020).

Fitzpatrick, K., W. Schneider and H. William, "Turn Speeds and Crashes within Right-turn Lanes", Tech. rep., Texas Transportation Institute, Texas A & M University System (2005).

Fok, C.-L., M. Hanna, S. Gee, T.-C. Au, P. Stone, C. Julien and S. Vishwanath, "A platform for evaluating autonomous intersection management policies", in "Cyber-Physical Systems (ICCPS), 2012 IEEE/ACM Third International Conference on", pp. 87–96 (IEEE, 2012).

Fuentes-Pacheco, J., J. Ruiz-Ascencio and J. M. Rendón-Mancha, "Visual simultaneous localization and mapping: a survey", Artificial Intelligence Review **43**, 1, 55–81 (2015).

Gazebo, "Robot simulator", `http://gazebosim.org/`, [Online; accessed 03-June-2019] (2019).

Gelfand, I. M., R. A. Silverman *et al.*, *Calculus of Variations* (Courier Corporation, 2000).

Gregoire, J., S. Bonnabel and A. de La Fortelle, "Optimal cooperative motion planning for vehicles at intersections", arXiv preprint arXiv:1310.7729 (2013).

Gross, J., Y. Gu, S. Gururajan, B. Seanor and M. Napolitano, "A comparison of extended kalman filter, sigma-point kalman filter, and particle filter in gps/ins sensor fusion", in "AIAA guidance, navigation, and control conference", p. 8332 (2010).

Guler, S. I., M. Menendez and L. Meier, "Using connected vehicle technology to improve the efficiency of intersections", Transportation Research Part C: Emerging Technologies **46**, 121–131 (2014).

Guney, M. A. and I. A. Raptis, "Scheduling-based optimization for motion coordination of autonomous vehicles at multilane intersections", Journal of Robotics **2020** (2020).

Gustafsson, J., A. Ermedahl, C. Sandberg and B. Lisper, "Automatic Derivation of Loop Bounds and Infeasible Paths for WCET Analysis using Abstract Execution", in "Real-Time Systems Symposium, 2006. RTSS'06. 27th IEEE International", pp. 57–66 (IEEE, 2006).

Hadjigeorgiou, A. and S. Timotheou, "Optimizing the trade-off between fuel consumption and travel time in an unsignalized autonomous intersection crossing", in "2019 IEEE Intelligent Transportation Systems Conference (ITSC)", pp. 2443–2448 (IEEE, 2019).

Hafizulazwan, B. M. N. M., "Optimal scheduling of connected and automated vehicles at urban intersections via milp", in "Proceedings of Joint Conference on Automatic Control", pp. 160–165 (J-Stage, 2018).

Hardy, D. and I. Puaut, "WCET Analysis of Multi-level non-inclusive Set-associative Instruction Caches", in "Real-Time Systems Symposium, 2008", pp. 456–466 (IEEE, 2008).

Hausknecht, M., T.-C. Au and P. Stone, "Autonomous intersection management: Multi-intersection optimization", in "Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on", pp. 4581–4586 (IEEE, 2011).

Hubmann, C., M. Becker, D. Althoff, D. Lenz and C. Stiller, "Decision making for autonomous driving considering interaction and uncertain prediction of surrounding vehicles", in "2017 IEEE Intelligent Vehicles Symposium (IV)", pp. 1671–1678 (IEEE, 2017).

Jin, Q., G. Wu, K. Boriboonsomsin and M. Barth, "Advanced intersection management for connected vehicles using a multi-agent systems approach", in "Intelligent Vehicles Symposium (IV), 2012 IEEE", pp. 932–937 (IEEE, 2012a).

Jin, Q., G. Wu, K. Boriboonsomsin and M. Barth, "Multi-agent intersection management for connected vehicles using an optimal scheduling approach", in "Connected Vehicles and Expo (ICCVE), 2012 International Conference on", pp. 185–190 (IEEE, 2012b).

Jin, Q., G. Wu, K. Boriboonsomsin, M. J. Barth *et al.*, "Platoon-based multi-agent intersection management for connected vehicle.", in "ITSC", pp. 1462–1467 (2013).

Kant, K. and S. W. Zucker, "Toward efficient trajectory planning: The path-velocity decomposition", The international journal of robotics research **5**, 3, 72–89 (1986).

Karlsson, J., J. Sjöberg, N. Murgovski, L. Hanning, S. Luu, V. Olsson and A. Rasch, "Intersection crossing with reduced number of conflicts", in "2018 21st International Conference on Intelligent Transportation Systems (ITSC)", pp. 1993–1999 (IEEE, 2018).

Katriniok, A., S. Kojchev, E. Lefeber and H. Nijmeijer, "Distributed scenario model predictive control for driver aided intersection crossing", in "2018 European Control Conference (ECC)", pp. 1746–1752 (IEEE, 2018).

Katriniok, A., P. Sopasakis, M. Schuurmans and P. Patrinos, "Nonlinear model predictive control for distributed motion planning in road intersections using panoc", arXiv preprint arXiv:1903.12091 (2019).

Khayatian, M., R. Dedinsky, S. Choudhary, M. Mehrabian and A. Shrivastava, "R2im – robust and resilient intersection management of connected autonomous vehicles", in "2020 IEEE International Conference on Intelligent Transportation Systems (ITSC)", (IEEE, 2020a).

Khayatian, M., Y. Lou, M. Mehrabian and A. Shirvastava, "Crossroads+: A time-aware approach for intersection management of connected autonomous vehicles", ACM Transactions on Cyber-Physical Systems **4**, 2, 20 (2019).

Khayatian, M., M. Mehrabian, H. Allamsetti, K.-W. Liu, P.-Y. Huang, C.-W. Lin and A. Shrivastava, "Cooperative driving of connected autonomous vehicles using responsibility-sensitive safety (rss) rules", in "Proceedings of the ACM/IEEE 12th International Conference on Cyber-Physical Systems", pp. 11–20 (2021).

Khayatian, M., M. Mehrabian, E. Andert, R. Dedinsky, S. Choudhary, Y. Lou and A. Shirvastava, "A survey on intersection management of connected autonomous vehicles", ACM Transactions on Cyber-Physical Systems **4**, 4, 1–27 (2020b).

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

Khoury, J. and J. Khoury, "Passive, decentralized, and fully autonomous intersection access control", in "Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on", pp. 3028–3033 (IEEE, 2014).

Kloock, M., P. Scheffe, S. Marquardt, J. Maczijewski, B. Alrifaee and S. Kowalewski, "Distributed model predictive intersection control of multiple vehicles", in "2019 IEEE Intelligent Transportation Systems Conference (ITSC)", pp. 1735–1740 (IEEE, 2019).

Koscher, K., A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham *et al.*, "Experimental security analysis of a modern automobile", in "2010 IEEE Symposium on Security and Privacy", pp. 447–462 (IEEE, 2010).

Le Vine, S., A. Zolfaghari and J. Polak, "Autonomous cars: The tension between occupant experience and intersection capacity", Transportation Research Part C: Emerging Technologies **52**, 1–14 (2015).

Lee, J. and B. Park, "Development and evaluation of a cooperative vehicle intersection control algorithm under the connected vehicles environment", IEEE Transactions on Intelligent Transportation Systems **13**, 1, 81–90 (2012).

Lee, J., B. B. Park, K. Malakorn and J. J. So, "Sustainability assessments of cooperative vehicle intersection control at an urban corridor", Transportation Research Part C: Emerging Technologies **32**, 193–206 (2013).

Li, L. and F.-Y. Wang, "Cooperative driving at blind crossings using intervehicle communication", IEEE Transactions on Vehicular technology **55**, 6, 1712–1724 (2006).

Li, N., Y. Yao, I. Kolmanovsky, E. Atkins and A. Girard, "Game-theoretic modeling of multi-vehicle interactions at uncontrolled intersections", arXiv preprint arXiv:1904.05423 (2019a).

Li, Y. J., "An overview of the dsrc/wave technology", in "International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness", pp. 544–558 (Springer, 2010).

Li, Z., Q. Wu, H. Yu, C. Chen, G. Zhang, Z. Z. Tian and P. D. Prevedouros, "Temporal-spatial dimension extension-based intersection control formulation for connected and autonomous vehicle systems", Transportation Research Part C: Emerging Technologies **104**, 234–248 (2019b).

Lin, P., J. Liu, P. J. Jin and B. Ran, "Autonomous vehicle-intersection coordination method in a connected vehicle environment", IEEE Intelligent Transportation Systems Magazine **9**, 4, 37–47 (2017).

Lin, S.-H. and T.-Y. Ho, "Autonomous vehicle routing in multiple intersections", in "Proceedings of the 24th Asia and South Pacific Design Automation Conference", ASPDAC '19, pp. 585–590 (ACM, New York, NY, USA, 2019), URL `http://doi.acm.org/10.1145/3287624.3287723`.

Lin, Y.-T. *et al.*, "Graph-based modeling, scheduling, and verification for intersection management of intelligent vehicles", ACM Transactions on Embedded Computing Systems (TECS) **18**, 5s, 1–21 (2019).

Liu, B., Q. Shi, Z. Song and A. El Kamel, "Trajectory planning for autonomous intersection management of connected vehicles", Simulation Modelling Practice and Theory **90**, 16–30 (2019a).

Liu, C., C.-W. Lin, S. Shiraishi and M. Tomizuka, "Distributed conflict resolution for connected autonomous vehicles", IEEE Transactions on Intelligent Vehicles **3**, 1, 18–29 (2017).

Liu, C., C.-W. Lin, S. Shiraishi and M. Tomizuka, "Distributed conflict resolution for connected autonomous vehicles", IEEE Transactions on Intelligent Vehicles **3**, 1, 18–29 (2018).

Liu, S., W. Zhang, X. Wu, S. Feng, X. Pei and D. Yao, "A simulation system and speed guidance algorithms for intersection traffic control using connected vehicle technology", Tsinghua Science and Technology **24**, 2, 160–170 (2019b).

Lu, Q. and K.-D. Kim, "Intelligent intersection management of autonomous traffic using discrete-time occupancies trajectory", Journal of Traffic and Logistics Engineering Vol **4**, 1, 1–6 (2016).

Lu, Q. and K.-D. Kim, "A mixed integer programming approach for autonomous and connected intersection crossing traffic control", in "2018 IEEE 88th Vehicular Technology Conference (VTC-Fall)", pp. 1–6 (IEEE, 2019).

Lu, X.-Y. and J. K. Hedrick, "Longitudinal control algorithm for automated vehicle merging", International Journal of Control **76**, 2, 193–202 (2003).

Mahbub, A., L. Zhao, D. Assanis and A. A. Malikopoulos, "Energy-optimal coordination of connected and automated vehicles at multiple intersections", arXiv preprint arXiv:1903.03169 (2019).

Makarem, L. and D. Gillet, "Decentralized coordination of autonomous vehicles at intersections", IFAC Proceedings Volumes **44**, 1, 13046–13051 (2011).

Malikopoulos, A. A. and L. Zhao, "A closed-form analytical solution for optimal coordination of connected and automated vehicles", in "2019 American Control Conference (ACC)", pp. 3599–3604 (IEEE, 2019a).

Malikopoulos, A. A. and L. Zhao, "Decentralized optimal path planning and coordination for connected and automated vehicles at signalized-free intersections", arXiv preprint arXiv:1903.04013 (2019b).

Malikopoulos, A. A. *et al.*, "Optimal control for speed harmonization of automated vehicles", IEEE Transactions on Intelligent Transportation Systems **20**, 7, 2405–2417 (2018).

Mazloom, S., M. Rezaeirad, A. Hunter and D. McCoy, "A security analysis of an in-vehicle infotainment and app platform", in "10th {USENIX} Workshop on Offensive Technologies ({WOOT} 16)", (2016).

Mehrabian, M., M. Khayatian, A. Shrivastava, J. C. Eidson, P. Derler, H. A. Andrade, Y.-S. Li-Baboud, E. Griffor, M. Weiss and K. Stanton, "Timestamp temporal logic (ttl) for testing the timing of cyber-physical systems", ACM Transactions on Embedded Computing Systems (TECS) **16**, 5s, 169 (2017).

Mehrabian, M. *et al.*, "An Efficient Timestamp-based Monitoring Approach to Test Timing Constraints of Cyber-physical Systems", in "Proceedings of the 55th Annual Design Automation Conference", p. 144 (ACM, 2018).

Mills, D. *et al.*, "Network time protocol", Tech. rep., RFC 958, M/A-COM Linkabit (1985).

Mirheli, A., M. Tajalli, L. Hajibabai and A. Hajbabaie, "A consensus-based distributed trajectory control in a signal-free intersection", Transportation Research Part C: Emerging Technologies **100**, 161–176 (2019).

Misener, J., "Sae connected vehicle standards", (2016).

Mohanan, M. and A. Salgoankar, "A survey of robotic motion planning in dynamic environments", Robotics and Autonomous Systems **100**, 171–185 (2018).

Murgovski, N., G. R. de Campos and J. Sjöberg, "Convex modeling of conflict resolution at traffic intersections", in "2015 54th IEEE conference on decision and control (CDC)", pp. 4708–4713 (IEEE, 2015).

Neuendorf, N. and T. Bruns, "The vehicle platoon controller in the decentralised, autonomous intersection management of vehicles", in "Mechatronics, 2004. ICM'04. Proceedings of the IEEE International Conference on", pp. 375–380 (Ieee, 2004).

Niels, T., N. Mitrovic, K. Bogenberger, A. Stevanovic and R. L. Bertini, "Smart intersection management for connected and automated vehicles and pedestrians", in "2019 6th International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)", pp. 1–10 (IEEE, 2019).

Onieva, E., U. Hernández-Jayo, E. Osaba, A. Perallos and X. Zhang, "A multiobjective evolutionary algorithm for the tuning of fuzzy rule bases for uncoordinated intersections in autonomous driving", Information Sciences **321**, 14–30 (2015).

Paull, L., J. Tani, H. Ahn, J. Alonso-Mora, L. Carlone, M. Cap, Y. F. Chen, C. Choi, J. Dusek, Y. Fang *et al.*, "Duckietown: an open, inexpensive and flexible platform for autonomy education and research", in "2017 IEEE International Conference on Robotics and Automation (ICRA)", pp. 1497–1504 (IEEE, 2017).

Peng, J. and S. Akella, "Coordinating multiple robots with kinodynamic constraints along specified paths", The International Journal of Robotics Research **24**, 4, 295–310 (2005).

Perronnet, F., J. Buisson, A. Lombard, A. Abbas-Turki, M. Ahmane and A. El Moudni, "Deadlock prevention of self-driving vehicles in a network of intersections", IEEE Transactions on Intelligent Transportation Systems **20**, 11, 4219–4233 (2019).

Perry, F., "Overview of dsrc messages and performance requirements", `https://tinyurl.com/yb4vlvw6`, [Online; accessed 28-Jan-2019] (2019).

Philippe, C., L. Adouane, A. Tsourdos, H.-S. Shin and B. Thuilot, "Probability collectives algorithm applied to decentralized intersection coordination for connected autonomous vehicles", in "2019 IEEE Intelligent Vehicles Symposium (IV)", pp. 1928–1934 (IEEE, 2019).

Pourmehrab, M., L. Elefteriadou, S. Ranka and M. Martin-Gasulla, "Optimizing signalized intersections performance under conventional and automated vehicles traffic", arXiv preprint arXiv:1707.01748 (2017).

Pruekprasert, S., X. Zhang, J. Dubut, C. Huang and M. Kishida, "Decision making for autonomous vehicles at unsignalized intersection in presence of malicious vehicles", arXiv preprint arXiv:1904.10158 (2019).

PTVAG, "Vissim - simulating driving behaviour", `https://tinyurl.com/yd7ycna6`, [Online; accessed 03-June-2019] (2019).

Qian, B., H. Zhou, F. Lyu, J. Li, T. Ma and F. Hou, "Toward collision-free and efficient coordination for automated vehicles at unsignalized intersection", IEEE Internet of Things Journal **6**, 6, 10408–10420 (2019).

Qiao, J., D. Zhang and D. de Jonge, "Virtual roundabout protocol for autonomous vehicles", in "Australasian Joint Conference on Artificial Intelligence", pp. 773–782 (Springer, 2018).

Quinlan, M., T.-C. Au, J. Zhu, N. Stiurca and P. Stone, "Bringing simulation to life: A mixed reality autonomous intersection", in "Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on", pp. 6083–6088 (IEEE, 2010).

Rios-Torres, J. and A. A. Malikopoulos, "Automated and cooperative vehicle merging at highway on-ramps", Transactions on Intelligent Transportation Systems **18**, 4, 780–789 (2016a).

Rios-Torres, J. and A. A. Malikopoulos, "A survey on the coordination of connected and automated vehicles at intersections and merging at highway on-ramps", IEEE Transactions on Intelligent Transportation Systems **18**, 5, 1066–1077 (2016b).

Romero, M. C. *et al.*, "Carma3 system architecture", `https://tinyurl.com/y6ws98h4`, [Online; accessed 16-March-2020] (2020).

Rossi, F., S. Bandyopadhyay, M. Wolf and M. Pavone, "Review of multi-agent algorithms for collective behavior: a structural taxonomy", IFAC-PapersOnLine **51**, 12, 112–117 (2018).

Sayin, M. O., C.-W. Lin, S. Shiraishi, J. Shen and T. Başar, "Information-driven autonomous intersection control via incentive compatible mechanisms", IEEE Transactions on Intelligent Transportation Systems **0**, 99, 1–13 (2018).

Setoodeh, P., A. Khayatian and E. Farjah, "Attitude estimation by divided difference filter-based sensor fusion", The journal of navigation **60**, 1, 119–128 (2007).

Setoodeh, P., A. Khayatian and E. Frajah, "Attitude estimation by separate-bias kalman filter-based data fusion", The Journal of Navigation **57**, 2, 261–273 (2004).

Shalev-Shwartz, S., S. Shammah and A. Shashua, "On a Formal Model of Safe and Scalable Self-driving Cars", arXiv preprint arXiv:1708.06374 (2017).

Sharon, G., S. D. Boyles and P. Stone, "Intersection management protocol for mixed autonomous and human-operated vehicles", Transportation Research Part C: Emerging Technologies (Under submission TRC-D-17-00857) (2017).

Sharon, G. and P. Stone, "A protocol for mixed autonomous and human-operated vehicles at intersections", in "International Conference on Autonomous Agents and Multiagent Systems", pp. 151–167 (Springer, 2017).

Shen, Z., A. Mahmood, Y. Wang and L. Wang, "Coordination of connected autonomous and human-operated vehicles at the intersection", in "2019 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)", pp. 1391–1396 (IEEE, 2019).

Shi, J., Y. Zheng, Y. Jiang, M. Zanon, R. Hult and B. Houskal, "Distributed control algorithm for vehicle coordination at traffic intersections", in "2018 European Control Conference (ECC)", pp. 1166–1171 (IEEE, 2018).

Shrivastava, A., P. Derler, Y.-S. L. Baboudr, K. Stanton, M. Khayatian, H. A. Andrade, M. Weiss, J. Eidson and S. Chandhoke, "Time in cyber-physical systems", in "Hardware/Software Codesign and System Synthesis (CODES+ ISSS), 2016 International Conference on", pp. 1–10 (IEEE, 2016).

Shrivastava, A., M. Mehrabian, M. Khayatian, P. Derler, H. Andrade, K. Stanton, Y.-S. Li-Baboud, E. Griffor, M. Weiss and J. Eidson, "A testbed to verify the timing behavior of cyber-physical systems", in "Proceedings of the 54th Annual Design Automation Conference 2017", p. 69 (ACM, 2017).

Steinmetz, E., R. Hult, Z. Zou, R. Emardson, F. Brännström, P. Falcone and H. Wymeersch, "Collision-aware communication for intersection management of automated vehicles", IEEE Access **6**, 77359–77371 (2018).

Stevanovic, A. and N. Mitrovic, "Combined alternate-direction lane assignment and reservation-based intersection control", in "2018 21st International Conference on Intelligent Transportation Systems (ITSC)", pp. 14–19 (IEEE, 2018).

Stone, P., S. Zhang and T.-C. Au, "Autonomous intersection management for semi-autonomous vehicles", in "Routledge Handbook of Transportation", pp. 116–132 (Routledge, 2015).

Tachet, R., P. Santi, S. Sobolevsky, L. I. Reyes-Castro, E. Frazzoli, D. Helbing and C. Ratti, "Revisiting street intersections using slot-based systems", PloS one **11**, 3, e0149607 (2016).

Tay, T.-T., I. Mareels and J. B. Moore, *High performance control* (Springer Science & Business Media, 2012).

Timmerman, R. and M. A. Boon, "Platoon forming algorithms for intelligent street intersections", arXiv preprint arXiv:1901.04583 (2019).

Toukabri, T., A. M. Said, E. Abd-Elrahman and H. Afifi, "Cellular vehicular networks (cvn): Prose-based its in advanced 4g networks", in "2014 IEEE 11th International Conference on Mobile Ad Hoc and Sensor Systems", pp. 527–528 (IEEE, 2014).

Tsai, J. S.-H., J.-C. Juang, C.-H. Tu, T.-Y. Tsai, P.-C. Chung, C.-C. Hsu, C.-Y. Lee and C.-F. Lin, "Development of key technologies for autonomous driving vehicles", in "2019 International Automatic Control Conference (CACS)", pp. 1–6 (IEEE, 2019).

Vasirani, M. and S. Ossowski, "A market-inspired approach for intersection management in urban road traffic networks", Journal of Artificial Intelligence Research **43**, 621–659 (2012).

Wang, X., S. Mao and M. X. Gong, "An overview of 3gpp cellular vehicle-to-everything standards", GetMobile: Mobile Computing and Communications **21**, 3, 19–25 (2017).

Wang, Y., P. Cai and G. Lu, "Cooperative autonomous traffic organization method for connected automated vehicles in multi-intersection road networks", Transportation Research Part C: Emerging Technologies **111**, 458–476 (2020).

Wei, H., L. Mashayekhy and J. Papineau, "Intersection management for connected autonomous vehicles: A game theoretic framework", in "2018 21st International Conference on Intelligent Transportation Systems (ITSC)", pp. 583–588 (IEEE, 2018).

Won, S.-h. P., W. W. Melek and F. Golnaraghi, "A kalman/particle filter-based position and orientation estimation method using a position sensor/inertial measurement unit hybrid system", IEEE Transactions on Industrial Electronics **57**, 5, 1787–1798 (2010).

Wu, J., A. Abbas-Turki and A. El Moudni, "Cooperative driving: an ant colony system for autonomous intersection management", Applied Intelligence **37**, 2, 207–222 (2012).

Wu, Y., H. Chen and F. Zhu, "Dcl-aim: Decentralized coordination learning of autonomous intersection management for connected and automated vehicles", Transportation Research Part C: Emerging Technologies **103**, 246–260 (2019).

Xu, H., Y. Zhang, L. Li and W. Li, "Cooperative driving at unsignalized intersections using tree search", IEEE Transactions on Intelligent Transportation Systems (2019).

Xu, Z., X. Li, X. Zhao, M. H. Zhang and Z. Wang, "Dsrc versus 4g-lte for connected vehicle applications: A study on field experiments of vehicular communication performance", Journal of Advanced Transportation **2017** (2017).

Yang, K., S. I. Guler and M. Menendez, "Isolated intersection control for various levels of vehicle technology: Conventional, connected, and automated vehicles", Transportation Research Part C: Emerging Technologies **72**, 109–129 (2016).

Zhang, Y., A. A. Malikopoulos and C. G. Cassandras, "Decentralized optimal control for connected automated vehicles at intersections including left and right turns", in "2017 IEEE 56th Annual Conference on Decision and Control (CDC)", pp. 4428–4433 (IEEE, 2017).

Zhang, Y. J., A. A. Malikopoulos and C. G. Cassandras, "Optimal control and coordination of connected and automated vehicles at urban traffic intersections", in "2016 American Control Conference (ACC)", pp. 6227–6232 (IEEE, 2016).

Zhao, L. and A. A. Malikopoulos, "Decentralized optimal control of connected and automated vehicles in a corridor", in "2018 21st International Conference on Intelligent Transportation Systems (ITSC)", pp. 1252–1257 (IEEE, 2018).

Zhao, X., J. Wang, Y. Chen and G. Yin, "Multi-objective cooperative scheduling of cavs at non-signalized intersection", in "2018 21st International Conference on Intelligent Transportation Systems (ITSC)", pp. 3314–3319 (IEEE, 2018).

Zheng, B., C.-W. Lin, S. Shiraishi and Q. Zhu, "Design and analysis of delay-tolerant intelligent intersection management", ACM Transactions on Cyber-Physical Systems **4**, 1, 1–27 (2019).

Zhu, F. and S. V. Ukkusuri, "A linear programming formulation for autonomous intersection control within a dynamic traffic assignment and connected vehicle environment", Transportation Research Part C: Emerging Technologies **55**, 363–378 (2015).

Zohdy, I. H., R. K. Kamalanathsharma and H. Rakha, "Intersection management for autonomous vehicles using icacc", in "Intelligent Transportation Systems (ITSC), 2012 15th International IEEE Conference on", pp. 1109–1114 (IEEE, 2012).

APPENDIX A

CONTROLLER DESIGN OF CROSSROADS+

For controller design, there are three gains to be determined: K_P, K_I, and K_D. Different values of K_P, K_I, and K_D will lead to different response times to achieve the target velocity. It is desired that the PID gains are such that the target velocity is achieved quickly. We use settling time as a metric for the CAV response time, which is defined as "the time it takes to reach and stay within 2% of the final value" and can be calculated as Tay *et al.* (2012):

$$T_s = \frac{3.9}{\zeta \omega_n} \tag{A.1}$$

where ζ is the damping ratio and ω_n is the natural frequency of the system Tay *et al.* (2012):

$$\zeta = \frac{K_P}{\sqrt{K_I(1+K_D)}}$$

$$\omega_n = \sqrt{\frac{K_I}{1+K_D}}$$

We define a cost function to minimize the settling time of the system:

$$min \ \ \frac{1}{\zeta \omega_n} \tag{A.2}$$

Based on the values of the PID controller (K_P, K_I, K_D), roots of the characteristic Equation (3.11) can be pure real or complex that correspond to overdamped and underdamped responses respectively. An underdamped response is not suitable for controlling the velocity because CAV's velocity will oscillate (around the set target velocity) before reaching the steady state. As a result, we only consider an overdamped situation (no oscillation). We rewrite the this requirement as a constraint for the optimization problem:

$$K_P^2 - 4K_I(1+K_D) > 0$$

Also, to ensure the stability of the closed-loop system, we have:

$$K_P > \sqrt{K_P^2 - 4K_I(1+K_D)}$$

A.1 Position Discrepancy for Saturated Case

The acceleration at time zero for the saturate case will be:

$$\dot{a}(0) = A^2 c_1 + B^2 c_2$$

$$= \frac{(v_T - v_0)}{A-B}(-BA^2 + AB^2)$$

$$= (v_0 - v_T)(A^2 + B^2)$$

Since the value of A or B is selected to be large, the initial acceleration will be large enough.

We can calculate the position error for the actual behavior with saturated acceleration:

$$e = \int v_T dt - \int v'(t) dt$$
$$= \int_0^{t_s} v_T dt -$$
$$\left(\int_0^{t_{SAT}} (v_0 + a_{max} t) dt + \int_{t_{SAT}}^{t_s} (c_1' e^{At} + c_2' e^{Bt} + v_T) dt \right) \tag{A.3}$$

We can use a change of variable ($t' = t - t_{SAT}$) in the last integral as:

$$\int_0^{t_s - t_{SAT}} (c_1' e^{At'} + c_2' e^{Bt'} + v_T) dt'$$

Finally, we can calculate the error as:

$$e = v_T t_s -$$
$$\left(\frac{a_{max} (t_{SAT})^2}{2} + v_0 t_{SAT} + v_T (t_c - t_{SAT} - \frac{c_1'}{A} - \frac{c_2'}{B}) \right) \tag{A.4}$$

A.2 Real-life Projection

In this section, we elaborate on details of how Crossroads+ can be implemented in real-life.

A.2.1 Slow-down Line

Slow-down lines are computed according to the max velocity and maximum deceleration rate of each CAV. Assuming a CAV is driving at velocity v and has maximum deceleration a_{min}, it will take

$$t_{SD} = \frac{v}{a_{min}} \tag{A.5}$$

seconds to come to a complete stop if the settling time of the response is short. Plugging in the deceleration time into the equation of motion of CAV, we can calculate the travel distance:

$$D_{SDL} = \frac{1}{2} a_{min} \left(\frac{v}{a_{min}} \right)^2 + v \left(\frac{v}{a_{min}} \right) \tag{A.6}$$

Simplifying, we can obtain the distance of the slow-down line (D_{SDL}) from the stop lines (transmit line and edge of the intersection):

$$D_{SDL} = \frac{3v^2}{2a_{min}}$$

A.2.2 Transmit Line

As we have discussed in previous sections, a key assumption is that CAVs are able to reach the assigned velocity before they reach the conflict point (x_c). As a result, we need to ensure that the transmit line is set sufficiently back from the intersection. By doing so, the IM would also be more flexible to assign a safe velocity to CAVs because not all CAVs are able to accelerate/decelerate quickly when the distance for the response is short.

According to Equation (3.3), the actuation time will be set Δt_{WCRTD} seconds after the request time. As a result, the CAV requires $v_0(\Delta t_{WCRTD})$ meters space while it's waiting for the response. Additionally, a CAV requires some time to maintain the target velocity which will be the summation of acceleration time under saturated acceleration (if any) and settling time of the controller. We can find the required distance as:

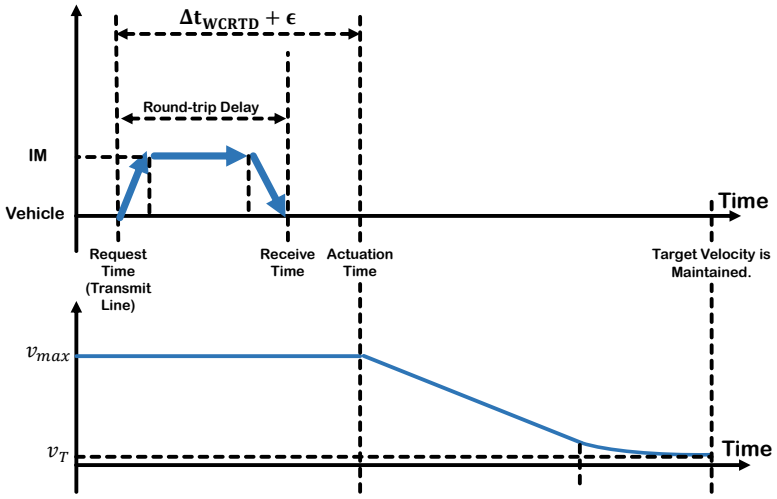


Figure A.1: In the Worst-case Behavior, a CAV Is Driving at the Maximum Velocity and the Assigned Target Velocity Is the Minimum Possible Velocity.

$$D_{TL} = v_0(\Delta t_{WCRTD}) + \frac{v_T - v_0}{a_{min}}\left(\frac{v_T - v_0}{2}\right) + T_s(v_S^{AVG}) \tag{A.7}$$

In this equation, D_{TL} is the required distance for the transmit line, v_0 is the initial velocity of the CAV, v_T is the target velocity assigned to the CAV, a_{max} is the maximum acceleration during the saturated behavior, T_s is the settling time of the controller (see Equations (A.1) and (A.2)) and v_s^{AVG} is the average velocity of the CAV during the settling time. We can find the worst-case scenario (maximum required distance) as:

$$D_{TL} = v_{max}(\Delta t_{WCRTD}) + \frac{v_{max}}{a_{min}^{slowest}}\left(\frac{v_{max}}{2}\right) + T_s(v_{max}) \tag{A.8}$$

$a_{min}^{slowest}$ is the slowest deceleration rate among all CAVs. We use an upper bound for v_S^{AVG} and replace it with v_{max} to make the calculation easier. Figure A.1 shows the scenario where the traveled distance is longest. Assuming that $v_{max} = 20m/s$ (≈ 45

mph), $a_{min}^{slowest} = 3m/s^2$, $\Delta t_{WCRTD} + \epsilon = 1s$ and $T_s = 0.1s$, we can find the distance of the transmit line from the edge of the intersection for a real scenario as $D_{TL} = 88.6m$.