Interaction between autonomous vehicles and other road users: a simulation study

Irtiza Rafid Khan

Follow this and additional works at: https://louis.uah.edu/uah-dissertations

Recommended Citation
https://louis.uah.edu/uah-dissertations/364

This Dissertation is brought to you for free and open access by the UAH Electronic Theses and Dissertations at LOUIS. It has been accepted for inclusion in Dissertations by an authorized administrator of LOUIS.
INTERACTION BETWEEN AUTONOMOUS VEHICLES AND OTHER ROAD USERS—A SIMULATION STUDY

Irtiza Rafid Khan

A DISSERTATION

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Civil and Environmental Engineering to The Graduate School of The University of Alabama in Huntsville December 2023

Approved by:

Dr. Rui Ma, Committee Chair
Dr. Michael D. Anderson, Committee Member
Dr. Abdullahi Salman, Committee Member
Dr. Sampson Gholston, Committee Member
Dr. Virginia Sisiopiku, Committee Member
Dr. Michael D. Anderson, Department Chair
Dr. Shankar Mahalingam, College Dean
Dr. Jon Hakkila, Graduate Dean
Abstract

INTERACTION BETWEEN AUTONOMOUS VEHICLES AND OTHER ROAD USERS—A SIMULATION STUDY

Irtiza Rafid Khan

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy

Civil and Environmental Engineering

The University of Alabama in Huntsville
December 2023

This dissertation evaluates the dynamics and conflicts between Autonomous Vehicles (AVs) and other road users, particularly focusing on the interactions at unsignalized crossings in urban settings and the implications of varying AV penetration rates on traffic performance. Utilizing a co-simulation approach which incorporates the autonomous vehicle simulator CARLA and the microscopic traffic simulation platform SUMO, this research comprehensively studied conflict scenarios at conflicting zones. Specifically, the influence of AV gap acceptance and waiting time thresholds on conflicts, such as collisions and lost time, was examined. Results showed that these thresholds significantly affect conflict performance indicators, highlighting a necessary tradeoff: designs with fewer collisions often result in higher lost times.

Extending this, the study also incorporated insights from the Chicken Game Theory (CGT) to delve deeper into the decision-making, risk assessment, and cooperative strategies of AVs in relation to Surrogate Safety Measures (SSM). Our simulations successfully quantified key SSM values, like Time to Collision (TTC) and Post Encroachment Time (PET), which are pivotal in determining collision likelihoods.
between AVs, bicycles, and other motor vehicles. For instance, an AV designed to initiate braking when reaching a TTC of 4.3 seconds or a PET of 1.7 seconds resulted in zero collisions, underscoring the effectiveness of this SSM-based strategy for AV safety. Lastly, by investigating AV penetration rates from 25% to 100% on traffic performance indicators, we identified a nuanced relationship between AV adoption and road safety and efficiency. Notably, even a 25% penetration rate significantly reduced near-miss incidents in some scenarios. However, the effect on queue length remained unchanged across all penetration levels, suggesting limitations in AV's ability to address certain congestion challenges, especially in saturated traffic systems. In summation, while AVs offer transformative potential for urban mobility, understanding their multifaceted interactions with current traffic systems is imperative. This research illuminates key areas of conflict and synergy, presenting insights for stakeholders ranging from policymakers and urban planners to technologists as we approach an era dominated by autonomous driving. Further investigations are essential to build upon these findings and fine-tune traffic systems for the future.
Acknowledgements

First and foremost, I express my profound gratitude to my advisor, Dr. Rui Ma, whose expertise, understanding, and patience, added considerably to my graduate experience. I appreciate the vast amount of time you have spent mentoring me, providing invaluable feedback, and encouraging me to believe in myself. I am also deeply thankful to my committee members- Dr. Michael Anderson, Dr. Abdullahi Salman, Dr. Sampson Gholston and Dr. Virginia Sisiopiku for their insightful comments and suggestions, and for challenging me to enhance my research from multiple perspectives.

A special thank you to my colleague and friend, Muhammad Usama, for the support and assistance that has been instrumental in the completion of this research. Your detailed advice and enthusiasm have been crucial during challenging periods.

I would like to acknowledge the contribution of Department of Civil and Environmental Engineering and the University of Alabama in Huntsville, for providing the necessary resources and an enabling environment that made this research possible.

I must express my appreciation to my family for their continuous support throughout this process. To my parents, Rashedul Islam Khan and Fouzia Khan, who have always believed in me and guided me, and to my wife, Faria, whose love and encouragement have been my pillars of strength.

Finally, a heartwarming thank you to my newborn son, Abrar, who brought a new perspective and joy into my life during the final stages of this journey. Your arrival has been the sweetest inspiration, reminding me of the beautiful possibilities that the future holds.
Dedication

To my beloved sisters- Nashra, Nahian, Eushra, Aisha, Rahmah and Sawdah.

My Ph.D. thesis is lovingly dedicated to you. Your unwavering support, encouragement, and love have been my guiding lights, making this journey not only possible but also joyful. This accomplishment is as much yours as it is mine. Thank you for everything.
Table of Contents

Abstract................................................................................................................................. ii

Acknowledgements ........................................................................................................... iv

Dedication ............................................................................................................................. v

Table of Contents ................................................................................................................ vi

List of Figures ....................................................................................................................... viii

List of Tables ......................................................................................................................... x

Epigraph ................................................................................................................................ xi

Chapter 1. Introduction........................................................................................................... 1

Chapter 2. Collision Avoidance Thresholds of Autonomous Vehicles at
Conflicts with Cyclists at Unsignalized Crossings – A Co-Simulation Study.. 3

  2.1 Introduction ...................................................................................................................... 3

  2.2 Literature Review ........................................................................................................... 4

  2.3 Materials and Methods ................................................................................................. 9

  2.4 Results ........................................................................................................................... 15

  2.5 Limitations and Safer Approach .................................................................................... 27

  2.6 Conclusions .................................................................................................................. 30

Chapter 3. A CARLA and SUMO Co-Simulation Study Between
Autonomous Vehicles and Other Road Users ................................................................. 32

  3.1 Introduction .................................................................................................................... 32
Chapter 4. Analyzing the Impact of Penetration Rates of Autonomous Vehicle in an Urban Environment—A SUMO Simulation Study

4.1 Introduction ........................................................................................................... 57
4.2 Literature Review ................................................................................................. 59
4.3 Methodology .......................................................................................................... 68
4.4 Data Collection & Analysis .................................................................................... 73
4.5 Conclusion .............................................................................................................. 86

Chapter 5. Conclusions .............................................................................................. 89

References .................................................................................................................. 92

Appendix A. Python Code for Collision Data (CARLA) .............................................. 103
Appendix B. Python Code for Calculating PET (CARLA) ........................................... 125
Appendix C. Python Code for Calculating TTC (CARLA) .......................................... 126
List of Figures

Figure 2.1 Layout of the scenario in SUMO (Left) and CARLA (Right) ..................... 11
Figure 2.2 Layout of the intersection in CARLA (Left) and SUMO (Right) ............... 12
Figure 2.3 Interaction between Autonomous Vehicle and Bicycle (Crossing) ............. 12
Figure 2.4 Flowchart of AV conflict-avoidance behavior at a bicycle crossing .......... 14
Figure 2.5 Average speed of ego vehicle for (a) “Aggressive” (b) “Conservative” approach.......................................................................................................................... 16
Figure 2.6 Collision Records for (a) “Aggressive” (b) “Conservative” approach....... 18
Figure 2.7 Time history of ego vehicle (Aggressive) ............................................ 21
Figure 2.8 Time history of ego vehicle (Conservative) .......................................... 22
Figure 2.9 Time history of ego vehicle (A44) ......................................................... 22
Figure 2.10 Collision Records for (a) Through movement ..................................... 23
Figure 2.11 Lost time for (b) Right-turn movement ............................................. 23
Figure 2.12 Contour Plots for Collisions between a through AV and cyclists (a) Collision Incident, (b) Collision Impulse with Bicycle, (c) Lost Time of Following Vehicles and (d) Lost Time of Ego Vehicle .................................................. 24
Figure 2.13 Contour Plots for Collisions between a left turning AV and cyclists (a) Collision Incident, (b) Collision Impulse with Bicycle, (c) Lost Time of Following Vehicles and (d) Lost Time of Ego Vehicle .................................................. 25
Figure 2.14 Contour Plots for Collisions between a right turning AV and cyclists (a) Collision Incident, (b) Collision Impulse with Bicycle, (c) Lost Time of Following Vehicles and (d) Lost Time of Ego Vehicle .................................................. 26
Figure 2. 15 Updated Flowchart of AV conflict-avoidance behavior at a bicycle crossing .......................................................... 28

Figure 3. 1 Different Scenarios in Chicken Game Theory ........................................ 41

Figure 4. 1 Huntsville Downtown Map in SUMO ........................................ 70

Figure 4. 2 Change in Average Speed for Different Penetration Rate of AV ............ 77

Figure 4. 3 Change in Near-Miss Incident for Different Penetration Rate of AV ........ 78
List of Tables

Table 2.1 Descriptive Statistics for Average Speed (Conservative) ........................................... 17
Table 2.2 Different Scenarios for Ego Vehicle .............................................................................. 19
Table 2.3 Autonomous Vehicle Interactions with Bicyclists at Varied Speeds ......................... 29
Table 3.1 Collision Impulse between Two AVs ........................................................................... 46
Table 3.2 Time to Collison (AV and Bicycle) .............................................................................. 48
Table 3.3 Time to Collison (AV and Manually Driven Vehicle) .................................................... 50
Table 3.4 PET (Post Encroachment Time) .................................................................................... 52
Table 4.1 T-Test Results for Average Speed for 10% AV Penetration ........................................ 78
Table 4.2 T-Test Results for Near-Miss Incident for 10% AV Penetration ................................. 79
Table 4.3 T-Test Results for Average Speed for 25% AV Penetration ........................................ 80
Table 4.4 T-Test Results for Near-Miss Incident for 25% AV Penetration ................................. 81
Table 4.5 T-Test Results for Average Speed for 50% AV Penetration ........................................ 82
Table 4.6 T-Test Results for Near-Miss Incident for 50% AV Penetration ................................. 82
Table 4.7 T-Test Results for Average Speed for 75% AV Penetration ........................................ 83
Table 4.8 T-Test Results for Average Speed for 100% AV Penetration ....................................... 83
Table 4.9 T-Test Results for Various AV Penetration (Dummy Data) ........................................... 84
The future is here – it's just not evenly distributed yet.

– William Gibson
Chapter 1. Introduction

The contemporary domain of urban transportation is witnessing a significant transformation with the advent of autonomous vehicles (AVs). These vehicles, equipped with state-of-the-art technologies, are poised to redefine the metrics of safety, efficiency, and overall transportation dynamics. However, their seamless integration into the existing transportation matrix presents multidimensional challenges. This dissertation synthesizes three pivotal studies, aiming to provide a holistic understanding of their implications. With the goal of offering a comprehensive grasp of their ramifications, this dissertation synthesizes three studies that outline the points at which AVs interface with the larger urban traffic milieu.

First, the discussion centers on the crucial comparison between efficiency and safety for autonomous vehicles. It is very important to thoroughly evaluate AV interactions, especially when they occur at unsignalized crossings with susceptible objects like bicycles. Using the co-simulation platforms, SUMO and CARLA, this research segment conducts an examination of gap acceptability and waiting time thresholds for autonomous vehicles. Finding the suitable tactics to balance safety concerns without sacrificing traffic flow is the aim.

The research narrative shifts from individual AV decisions to the complex web of multi-agent interactions on the roadways. The study utilizes the concepts of the Chicken Game Theory, to comprehend the dynamics of AV conflict resolution. Understanding that the conventional two-player situations in this theory have limits, particularly when it
comes to busy metropolitan junctions, the research shifts its focus to the use of surrogate safety measures. The goal of this methodological change is to provide a more nuanced understanding of potential conflicts and mitigation techniques in multi-agent systems by emphasizing microscopic measures like time-to-collision and post-encroachment time.

Lastly, the role of simulation studies becomes necessary when projecting the impact of increased AV penetration rates in urban settings. With only a handful of AVs currently on the road, simulation studies, particularly those employing tools like SUMO, allow us to anticipate the challenges and implications of a future with widespread AV adoption. Such simulations provide insights into the coexistence dynamics between human-driven and autonomous vehicles.

Connecting these studies is a central thread: the quest for a harmonious integration of AVs into our urban milieu. By journeying through immediate challenges, complex interactions, and future visions, our aim is to contribute to the chorus of voices shaping the next chapter of urban transportation. In essence, this dissertation, through a seamless amalgamation of immediate challenges, interaction complexities, and future-oriented analyses, seeks to contribute substantively to the academic and practical discourse on AV integration in urban landscapes.
Chapter 2. Collision Avoidance Thresholds of Autonomous Vehicles at Conflicts with Cyclists at Unsignalized Crossings—A Co-Simulation Study

2.1 Introduction

Emerging autonomous vehicles (AV) have changed the landscape of current research and practice on transportation. Safety concerns and traffic efficiency are among the significant consideration of the AVs applications. The interactions between AVs and other vulnerable road participants are among the heated topics regarding AV safety and efficiency concerns. Regarding safety concerns, although the public generally supposes AV is a safer alternative to interact with cyclists [1], the real-world performance of AV and semi-autonomous vehicles brings mixed opinions from the public. Even with advanced technologies in object detection, automated braking, and other technologies implemented on an AV, it is unlikely that AV can entirely avoid all conflicts between AVs and cyclists in real-world scenarios. For instance, Arizona had a recent fatality during an AV road test [2]. Regarding efficiency, one of the selling points to potential AV owners and riders is that AV can significantly enhance the driving experience and save travel time and effort. Researchers also assume that in the total AV or mixed AV and human driving traffic, the headway between vehicles will be shorter, which leads to higher traffic capacity and, thus a more efficient road system. However, various real-world AV applications, such as autonomous transit vehicles in dedicated environments (e.g., airport shuttles), have shown the opposite. When an AV applies a high safety standard and conservatively yields to conflicting objects often, the AV and the vehicular
traffic may severely suffer from stop-and-go patterns, longer travel time, and less desired riding experience.

The proper balance between safety and efficiency for AVs is thus critical for successful AV implementations. However, current research on balancing the safety and efficiency among AVs and cyclists are still sparse. This research aims to fill the research gap by investigating how the gap acceptance and waiting time thresholds of AVs affect the conflicts among autonomous vehicles and cyclists on urban streets in a co-simulation setup. Two simulation tools compose the co-simulation platform - the Simulation of Urban Mobility (SUMO), an open-source microscopic traffic simulation platform, and the Car Learning to Act (CARLA), an emerging autonomous vehicle simulation environment. Various strategies in terms of gap acceptance and waiting time thresholds of the autonomous vehicles on reacting against potential conflicts with the cyclists due to cyclist crossing at unsignalized intersections are developed and tested in this co-simulation platform.

2.2 Literature Review

Many studies on the AVs concentrate on technology associated with the hardware of the AVs. However, recent studies have started on the interactions between human road participants and AVs. A simulator study was conducted by Toffetti et al. to observe the difference in drivers' perception about transitioning from manual driving to autonomous driving [3] and tested the differences between a vocal and acoustic user interface in autonomous system. Another study was based on a survey based on a number of drivers' perceptions of various driver assistance systems [4]. Seppelt et al. [5] discussed a computational model that was utilized for assessing the interaction between the driver
and the Adaptive Cruise Control (ACC) system. Moreover, the implications of such a computational model for a higher level of automation were also discussed. Cunningham et al. studied drivers' attention and distraction and highlighted how it could adversely affect safety while driving in an AV [6]. Some studies also concentrated on users' attitudes toward AVs and their concerns about safety and reliability [7-9]. Despite the recent tragedy involving an Uber testing AV and the death of a cyclist in Tempe, Arizona, on March 18, 2018 [2], the interactions between cyclists and AVs are not thoroughly investigated in the literature. On the AV side, manufacturers have implemented collision avoidance as one of the critical elements in the Advanced Driver Assistance System, which has improved perception systems and aims to avoid crashes with nonmotorized vehicles like cyclists' communication and safety techniques [10]. For example, complete auto-brake methods have the ability to discover cyclists, pedestrians, and other roadway users and promote an emergency brake if necessary.

Studies show that the general road participants tend to accept that AVs may be a safer option for vulnerable groups such as cyclists. For instance, semi-autonomous cars on city roadways have been experimented with by Google since 2009 and Uber since the year 2016 [11]. A survey on the semi-autonomous vehicles deployed in Pittsburgh demonstrates that cyclists typically feel safer when they are around autonomous cars compared to manually driven vehicles [12]. Thus far, the AV industry and research teams have been focusing on the technical challenge of detecting cyclists. Pammer et al. [1] surveyed the level of trust, comfort, and feeling of safety of motorcyclists and cyclists in human drivers and AVs. In that study, questionnaires were conducted to examine the dissimilarity between their confidence generally and confidence in their safety. Results
indicated that their confidence level to interact with manually driven and autonomous vehicles is medium to inferior for both typical human drivers and AVs. However, while considering their safety, they are notably more likely to acknowledge that AVs are a safer option than human-driven cars. It shows that, in general, cyclists have an increased feeling of defenselessness while they are on the street, while they tend to accept the introductions of AVs to reduce safety concerns.

Currently, drivers and cyclists intercommunicate through unspoken signals, such as the movement of the automobile, horns, sign language including hand gestures, and even somewhat ambiguous eye contact. However, it is still very challenging for state-of-art AVs to fully understand and communicate with cyclists nearby. Instead, current technology is better at detecting the location, speed, size, and categories of the encountered cyclists by utilizing sensor data and their established testing database. Hou et al. [12] demonstrated an investigation of the exchange between AVs and cyclists, initiating introductory design analyses that disclosed the execution of an engaging virtual reality (VR) simulator and the configuration and assessment of several cyclist-AV interfaces. The results indicate that interaction between cyclists and AVs possesses the capacity to enhance the conviction of riders in the scenarios of merging with lanes. In addition, this research came up with a cyclist-AV simulator, an understanding of trade-offs of different factors of the format of interaction between AVs and cyclists, as well as location, modalities, and sophistication. Positive results suggest developed rider conviction due to the cyclist-AV relationship.

It is generally convinced that mature AV technologies will help lowering street fatalities. A recent study [13] has been conducted on their impact on the safety of
vulnerable roadway users, e.g., cyclists. Their study focused on the relations between AVs and bicyclists and discussed whether AVs' predicted execution could better protect cyclists. The PTV Vissim [14], a microscopic traffic flow simulation software, was utilized in their study, integrated with the surrogate safety assessment model (SSAM) [15]. Furthermore, the roadway network was developed based on a medium-sized city in Belgium as a case study, where cyclists and vehicles share narrow streets in the city center on numerous occasions. The analysis conclusions demonstrate a significant reduction in the number of altercations between cars and between cyclists and vehicles compared to the existing circumstances. Moreover, the intensity of conflicts also declined due to the lack of typical human-driven automobiles in the traffic streams. In that particular study [13], it was assumed that besides bicycles, all other vehicles are autonomous, which might not be the case in real life. Therefore, there can be a mixture of human-driven and autonomous vehicles on the urban roadway. This study focused mainly on the rear-end collision between bicycles and AVs. Hence, a simulation study on AVs in which there is an unsignalized crossing of bikes is available should be investigated.

Instead of the costly experiments in the real world, testing the behavior and responses of road participators, including AVs and cyclists in simulations can be an affordable and realistic alternative. In many scenarios regarding potential conflicts, simulations are a much more economical approach than real-world experiments. Simulation has been applied for training driving models [16]. Traditional simulation designs were typically for guiding and benchmarking robotic assessing procedures [17-24]. Lately, racing simulators have been used to evaluate novel techniques for
autonomous driving [25, 26]. Moreover, popular games have been employed to obtain high-fidelity data for coaching and benchmarking visual observation methods [27, 28].

Studies on the conflicts between AVs and cyclists rely on properly selected simulation platforms. Most traditional simulation platforms cannot deal with the meso- and microscopic traffic, the multiple types of road participants, and the AV sensors input and vehicle maneuvers at the same time. For instance, TORCS, The Open Racing Car Simulator [26], does not exhibit the complexity of urban driving, as they lack pedestrians, intersections, cross-traffic, traffic rules, and other complexities that separate track racing from urban driving. Some emerging popular video games that resemble urban surroundings at significant accuracies, such as Grand Theft Auto V [27, 28], do not maintain comprehensive driving strategies nor the proper presentation of traffic dynamics. Commercial microscopic simulation platforms usually have limited ability for customization over AV characteristics, and they critically lack accurate sensor suite specifications. Vehicle-level microscopic specifications, such as explicit feedback upon breach of traffic laws and smooth shift of views between bird-view and point-of-view as the first person to the ego vehicles (connected and/or automated vehicle, the behavior of which is of primary interest in testing, trialing or operational scenarios), as well as mesoscopic multiple performance indicators for the system, all contribute to the selection of simulation platforms in this study. This research selects a co-simulation platform that jointly utilizes the SUMO [29] and CARLA [30].

CARLA provides complete control of all static and dynamic actors, map generation, and much more. Though this platform was established for establishing the development, preparation, and validation of the driving system of autonomous vehicles,
CARLA is not ideal for constructing a complicated traffic network [31] because it lacks the manageability of large traffic volumes and fails to represent realistic traffic flow. SUMO has the capability to manage large-scale traffic flows. SUMO has dynamics modeling for each vehicle, allowing users to assemble customized traffic strategies quickly through the TraCI (Traffic Control Interface) API. From SUMO, the data of other vehicles, including the bicycle’s performance, can be retrieved. In addition, SUMO provides characteristics of each vehicle running in the simulation scenario, including the speed, lost time, collision time, etc. From CARLA, detailed information on the AVs can be retrieved, including the speed, acceleration, and collision history of the AVs.

Combining these two simulation platforms, all the collected data were analyzed on how the unsignalized bicycle crossing may experience conflicts with the AVs presented on the main roadway. We then evaluate the simulation results for various settings of the AV’s gap acceptance and waiting times.

This research is organized as follows. First, Section ‘Method” presents the methodology for this study. The methodology includes the autonomous vehicle design in CARLA and the scenario design in SUMO. Then, the section ‘Result and discussions’ involves designing two extreme AV strategies and different modified AV strategies for bicycle crossing scenarios. Data analysis and the explanation of the outputs are discussed further in this section. This study then concludes with the research findings and discussions on future research.

2.3 Materials and Methods

This research employed simulator experiments to evaluate the interaction between the autonomous vehicle (AV) and bicycle users. SUMO [29] and CARLA [30] have been
CARLA is an open-source simulator introduced for independent or self-supported driving investigation. CARLA was designed thoroughly to sustain the growth, activity, evolution, and assurance of autonomous urban driving strategies. In addition, CARLA is equipped with open digital support such as vehicles, buildings, urban layouts, and open-source code and protocols designed to fulfill this goal and can be operated freely. CARLA incorporates urban designs, models of vehicles, structures, walkers, signs on the road, etc. The simulation program encourages the adjustable settings of sensor suites. It renders signals which can practice driving policies, such as coordinates of GPS, velocity, acceleration, and comprehensive data on accidents and other breaches. A broad spectrum of environmental circumstances can be defined, including time of day and weather. CARLA promotes the improvement, coaching, and comprehensive performance interpretation of autonomous driving operations. The combination of realism, flexibility, and a robust set of features makes CARLA a powerful tool for the comprehensive testing and development of autonomous driving technologies, allowing for rigorous and diverse testing before real-world deployment.

Even though CARLA provides a traffic manager module for generating background traffic, they are based on simplistic behavior rules, which cannot represent actual driver behavior. Further, SUMO can render traffic using diverse well-accepted driver models (e.g., Intelligent Driver Model [32]). CARLA has a co-simulation feature with SUMO and other simulation platforms. When the co-simulation is triggered, SUMO will regulate the traffic and transform the background human-driven automobiles into the
CARLA server. The AVs governed by CARLA will respond to the traffic to complete their driving responsibilities.

Figure 2.1 signifies the chosen traffic network, which represents an urban area consisting of freeways, ramps, and signalized and unsignalized urban intersections with local roads, with no adverse weather and ambient daytime lighting. The freeway has a posted speed limit of 96.6 km/h (60 mph), divided by a median and the signalized roadways have a speed limit of 48.3 km/h (30 mph). There was also surrounding traffic density for keeping the simulation scenario similar to the real-life situation.

We choose unsignalized bicycle crossings on a highway street in an urban traffic network as the study locations. The reason we chose these spots is that such unsignalized intersections are simple enough to identify conflict zones between cyclists and AVs, while the conflicts are very likely to occur due to no strict right-of-way enforced by any traffic control devices or signs. Figure 2 represents the intersection where the interaction between bicycle and AV was studied.

Figure 2.1 Layout of the scenario in SUMO (Left) and CARLA (Right).
Figure 2. 2 Layout of the intersection in CARLA (Left) and SUMO (Right).

Figure 2. 3 Interaction between Autonomous Vehicle and Bicycle (Crossing).
CARLA, we are able to decide the AV’s strategies to determine when a potential conflict emerges and whether the AVs should enact accordingly. Figure 4 represents the procedure of AV’s decisions when a potential conflict arises against cyclists in a conflicting movement. There are two major parameters of an AV in such collision avoidance strategies. The first one is the gap acceptance. For example, the AV may observe a bunch of cyclists in a row in a conflicting direction. The AV will seek a sufficient gap to cross the conflicting zone. In other words, the AV is designed to see if there is enough gap between the AV and all other objects (including any conflicting bicycles). This study selects such an accepted gap from a range of 0 to 12 feet in this study. If the actual gap is shorter than the accepted gap, the vehicle stops and waits. The second parameter is the waiting time after the AV decides the existing gap is shorter than the accepted gap. The waiting time is selected from a range of 0 to 5 seconds. To simplify the simulations, after the waiting period is over, the AV starts to move and attempt to cross the conflict zone. Such a move leads to either a collision with a bicycle or no collision (i.e., a safe move).
Figure 2.4 Flowchart of AV conflict-avoidance behavior at a bicycle crossing.

Designing an Autonomous Vehicle

One of the most important and critical tasks was to design and add an autonomous vehicle, which would also assist in investigating the behavior of bicycles on the roadways. Several issues were considered during designing the autonomous vehicles, such as: following traffic signals, stopping or adjusting the speed while there is a slow-moving vehicle (\emph{i.e.}, bicycles or other vehicles) in front of it, maintaining lane while
driving, and turning properly while driving along with a curve. The design and the
necessary coding for the autonomous vehicle were done in CARLA. The car running in
the scenario is a TESLA model 3 with the modified gap acceptance and waiting times.
This research concentrates on Level 4 automation [33], under which the car user entirely
hands over the control.

2.4 Results

After designing all the vehicles and related scenarios, the simulation was run 10
times for each pair of gap acceptance and waiting time settings. To find out the conflict
during the bicycle crossing, we collected the average speed, average impulse, lost time,
etc., for each setting. In addition, data were collected for three types of AV movements:
through, left-turn, and right-turn movements. This section outlines the results and
discusses the findings.

We first design two extreme AV strategies and will expand on them later. The
first one is aggressive, where the ego AV only checks its immediate surroundings and
will wait for a brief period before starting to move. The problem with such an aggressive
maneuver is that the ego vehicle, at most, only dodges the first bicycle and cannot
consider other following bikes. Such a myopic strategy is very likely to cause a collision
between the bicycles and the ego AV. On the other hand, if there happens to be no
bicycle around, such an aggressive strategy saves the travel time of the AV.

The other extreme strategy is a conservative one. The ego AV will seek a large
gap where no other objects are detected in a larger surrounding area before counting the
waiting time. The waiting time is long enough to consider any new upcoming bicycles
from outside of the initial surrounding area. This approach maximizes the possibility of
avoiding collision between the ego vehicles and the bicycles and guarantees the best possible safety. On the other hand, it slows down the ego AV significantly and further slows down the following vehicles behind the ego AV; thus, it can hardly be an efficient strategy.

**Figure 2.5** Average speed of ego vehicle for (a) “Aggressive” (b) “Conservative” approach.

Figure 2.5 shows the difference of average speed for two extreme methods. In the figure, “before” means the average speed of the ego vehicle when it was driving without facing any obstacle on the roadway, and “after” means the average speed of the ego vehicle after it faced any obstacle while driving. The obstacle here was in the forms of slow-moving vehicle in front of the ego vehicle. From figure 5a, the average speed of the AV does not change much after perceiving a potential conflict, and the reason is that even the AV reduced its speed after perceiving a conflict, it tried to get back to the normal speed in a quick manner. Two-tailed paired t-tests were performed to compare the before ($\mu_1$) and after ($\mu_2$) segments to assess any significant differences in the average speed. The assumed null hypothesis was that there were no significant differences in speed before and after observing the bicycle crossing. The independence and normality of the
sample data were sufficiently satisfied. The null hypothesis is rejected by the t-tests, which means the average speeds before and after the potential conflict are significantly different, and they are not from the same distribution.

Table 2.1 provides descriptive statistics of average speed measurements for the “Conservative” approach. Significant differences in average speed, with a medium effect size (Cohen’s $d > 0.5$), were observed before and after perceiving bicycle crossing ($p < 0.001$). A 68% percent reduction in average speed was observed in the conservative approach, as shown in Figure 2.5.

<table>
<thead>
<tr>
<th>Avg Speed</th>
<th>Mean</th>
<th>N</th>
<th>Std. Deviation</th>
<th>Std. Error Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>Before</td>
<td>31.9733</td>
<td>300</td>
<td>1.96518</td>
<td>0.11346</td>
</tr>
<tr>
<td>After</td>
<td>10.1700</td>
<td>300</td>
<td>6.05226</td>
<td>0.34943</td>
</tr>
</tbody>
</table>

When the ego vehicle collided with bicycles, the collision incidents were recorded by the co-simulation. Along with it, the impulse is retrieved from the collision event as 3D vector components. Impulse is the quantity that describes how the collision between the ego vehicle and the bicycle changes the linear momentum of the bicycle. Initially, simulations were run 10 times to observe any difference in collision records between the conservative and aggressive approaches. Figure 6 shows the number of collisions and average impulse associated with each simulation.
From figure 2.6, it can be observed that the conservative approach is much safer than the aggressive one. But there is one limitation in the cautious mode: the ego vehicle takes too much time to move, resulting in delay. It also causes time loss for the other vehicles following the ego vehicle. So, comparing the time lost is necessary. When the ego vehicle encounters a bicycle ahead, it waits for one second and starts to move in an aggressive approach. In the conservative approach, the ego vehicle stops if the gap is less than 12 ft, then waits for 5 seconds. The waiting time and allowed gap of the ego vehicle were changed to develop different designs, and later autonomous vehicles with these designs were run in the simulation. There were 65 different combinations with accepted gap and waiting time which is shown in Table 2.2, and each design scenario was observed by running in the same simulation environment to keep the consistency of the output data.
### Table 2.2 Different Scenarios for Ego Vehicle.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Gap (ft)</th>
<th>Waiting Time (s)</th>
<th>Scenario</th>
<th>Gap (ft)</th>
<th>Waiting Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aggressive</td>
<td>0</td>
<td>1</td>
<td>A32</td>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td>Conservative</td>
<td>12</td>
<td>5</td>
<td>A33</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>A1</td>
<td>0</td>
<td>2</td>
<td>A34</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>A2</td>
<td>0</td>
<td>3</td>
<td>A35</td>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>A3</td>
<td>0</td>
<td>4</td>
<td>A36</td>
<td>7</td>
<td>2</td>
</tr>
<tr>
<td>A4</td>
<td>0</td>
<td>5</td>
<td>A37</td>
<td>7</td>
<td>3</td>
</tr>
<tr>
<td>A5</td>
<td>1</td>
<td>1</td>
<td>A38</td>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td>A6</td>
<td>1</td>
<td>2</td>
<td>A39</td>
<td>7</td>
<td>5</td>
</tr>
<tr>
<td>A7</td>
<td>1</td>
<td>3</td>
<td>A40</td>
<td>8</td>
<td>1</td>
</tr>
<tr>
<td>A8</td>
<td>1</td>
<td>4</td>
<td>A41</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td>A9</td>
<td>1</td>
<td>5</td>
<td>A42</td>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>A10</td>
<td>2</td>
<td>1</td>
<td>A43</td>
<td>8</td>
<td>4</td>
</tr>
<tr>
<td>A11</td>
<td>2</td>
<td>2</td>
<td>A44</td>
<td>8</td>
<td>5</td>
</tr>
<tr>
<td>A12</td>
<td>2</td>
<td>3</td>
<td>A45</td>
<td>9</td>
<td>1</td>
</tr>
<tr>
<td>A13</td>
<td>2</td>
<td>4</td>
<td>A46</td>
<td>9</td>
<td>2</td>
</tr>
<tr>
<td>A14</td>
<td>2</td>
<td>5</td>
<td>A47</td>
<td>9</td>
<td>3</td>
</tr>
<tr>
<td>A15</td>
<td>3</td>
<td>1</td>
<td>A48</td>
<td>9</td>
<td>4</td>
</tr>
<tr>
<td>A16</td>
<td>3</td>
<td>2</td>
<td>A49</td>
<td>9</td>
<td>5</td>
</tr>
<tr>
<td>A17</td>
<td>3</td>
<td>3</td>
<td>A50</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>A18</td>
<td>3</td>
<td>4</td>
<td>A51</td>
<td>10</td>
<td>2</td>
</tr>
<tr>
<td>A19</td>
<td>3</td>
<td>5</td>
<td>A52</td>
<td>10</td>
<td>3</td>
</tr>
<tr>
<td>A20</td>
<td>4</td>
<td>1</td>
<td>A53</td>
<td>10</td>
<td>4</td>
</tr>
<tr>
<td>A21</td>
<td>4</td>
<td>2</td>
<td>A54</td>
<td>10</td>
<td>5</td>
</tr>
<tr>
<td>A22</td>
<td>4</td>
<td>3</td>
<td>A55</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>A23</td>
<td>4</td>
<td>4</td>
<td>A56</td>
<td>11</td>
<td>2</td>
</tr>
<tr>
<td>A24</td>
<td>4</td>
<td>5</td>
<td>A57</td>
<td>11</td>
<td>3</td>
</tr>
<tr>
<td>A25</td>
<td>5</td>
<td>1</td>
<td>A58</td>
<td>11</td>
<td>4</td>
</tr>
<tr>
<td>A26</td>
<td>5</td>
<td>2</td>
<td>A59</td>
<td>11</td>
<td>5</td>
</tr>
<tr>
<td>A27</td>
<td>5</td>
<td>3</td>
<td>A60</td>
<td>12</td>
<td>1</td>
</tr>
<tr>
<td>A28</td>
<td>5</td>
<td>4</td>
<td>A61</td>
<td>12</td>
<td>2</td>
</tr>
</tbody>
</table>
Figure 2.7, 2.8 and 2.9 represent time diagram of the ego vehicle for three different scenarios. It consists of three phases: 1) cruising without obstacles (from -1 to 0 second), 2) perceiving a potential conflict (bicycle crossing), symbolized by the star sign and 3) The decision made to decelerate to avoid the collision (from 0 to 1.3 second). The moment the ego vehicle predicts a potential conflict, a stop command is delivered to the vehicle.

Figure 2.7 illustrates how the ego vehicle behaves in an aggressive setup. In this particular scenario, the vehicle doesn’t look for a gap and wait just for 1 second after it encounters any obstacles. After waiting for 1 second, the vehicle starts moving, and it’s not safe as the AV doesn’t consider any gap and there might be another bicycle crossing at that time, which can result into a collision. The particular example shown in Figure 7 suggests a constant deceleration rate around 16 m/s$^2$ or 52 ft/s$^2$. 

<table>
<thead>
<tr>
<th>A29</th>
<th>5</th>
<th>5</th>
<th>A62</th>
<th>12</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>A30</td>
<td>6</td>
<td>1</td>
<td>A63</td>
<td>12</td>
<td>4</td>
</tr>
<tr>
<td>A31</td>
<td>6</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Figure 2. 7 Time history of ego vehicle (Aggressive).

Figure 2.8 illustrates how the ego vehicle behaves in a conservative setup. The particular example shown in Figure 8 suggests a constant deceleration rate around 12 m/s^2 or 39 ft/s^2, which is beyond the comfortable deceleration for regular vehicles as recommended (11.2 ft/s^2). This indicates that in the conservative scenario, the ego AV sacrifices the comfortableness of its own passengers to make sure there is no conflict or collision between the vehicle and the cyclists.
Figure 2.8 Time history of ego vehicle (Conservative).

Figure 2.9 illustrates how the ego vehicle behaves in A44 scenario, where the AV stops if the gap between it and the bicycle is less than 8 ft, and then waits for 5 more seconds. This design is also helpful in avoiding collision with crossing bicycle, which we can see from Figure 2.10. The example shown in Figure 2.9 suggests a constant deceleration rate around 16 m/s^2 or 52 ft/s^2, which is also beyond the comfortable deceleration for regular vehicles as recommended (11.2 ft/s^2).

Figure 2.9 Time history of ego vehicle (A44).
Figure 2.10 represents the collision record of through movement. Besides the cautious one, design A44, A53, A58, A59, A61, and A63 are the safest, without bicycle collisions at the crossing.

![Figure 2.10 Collision Records for (a) Through movement.](image)

From the simulation, the lost time of the following vehicles, along with the ego vehicle, was also documented. Figure 2.11 shows the lost time for right-turn movement.

![Figure 2.11 Lost time for (b) Right-turn movement.](image)

The objective was not only to find the design with no collision but also the design with comparatively smaller lost time. Moreover, 6 more vehicles were designed to follow the ego vehicle in the simulation scenario to determine if the strategy change also
changed those vehicles' lost time. The following figures illustrate how different gaps and waiting times altered the output of separate incidents.

Figure 2. 12 Contour Plots for Collisions between a through AV and cyclists (a) Collision Incident, (b) Collision Impulse with Bicycle, (c) Lost Time of Following Vehicles and (d) Lost Time of Ego Vehicle.
Figure 2.13 Contour Plots for Collisions between a left turning AV and cyclists (a) Collision Incident, (b) Collision Impulse with Bicycle, (c) Lost Time of Following Vehicles and (d) Lost Time of Ego Vehicle.
Figure 2.14 Contour Plots for Collisions between a right turning AV and cyclists (a) Collision Incident, (b) Collision Impulse with Bicycle, (c) Lost Time of Following Vehicles and (d) Lost Time of Ego Vehicle.

Figures 2.12, 2.13, and 2.14 are the plots for Collisions between a through-moving AV and cyclists, a left-turning AV and cyclists, and a right-turning AV and cyclists, respectively. These contour plots present four performance indicators for different AV gap acceptance and waiting time thresholds. One pattern that can be observed here is that the collision of AV with bicycles tends to decrease with the increase in waiting time and gap. This is because as the AV keeps more gaps and waits a bit longer, its chance of colliding with a bicycle decreases. More specifically, for the same
observed gap distance, if the allowed waiting time for the AV increases, the collision between AV and bikes goes down; and this pattern is mostly consistent for through, left, and right turning movement of the AV, which can be seen in Figures 2.12, 2.13 and 2.14.

On the other hand, the lost time for both the ego vehicle and other following vehicles goes up along with the increment of gap and waiting time. For the same gap distance of the AV, the more waiting time the AV has, the more time loss occurs. So, it is necessary to get a proper combination of accepted gap distance and waiting time of AV, which will be safe for bicycles in the roadway, and at the same time, will also be efficient enough to reduce the time loss.

Though safety is the primary concern regarding autonomous vehicles, keeping the time loss as low as possible should also be the focus. Therefore, one of the goals of this study was to find a proper design scenario for AV that would help balance the conflict vs. efficiency of AV and cyclist scenarios. From the simulation, design A44, with an 8 ft gap and 5 seconds allowed waiting time, was found to have a perfect balance; as in A44, the AV did not collide with the cyclists a single time for all three movements, and A44 also had the shortest lost time compared to other safe options where zero collision with bicycles was observed.

2.5 Limitations and Safer Approach

One primary limitation of this research is the conflict avoidance protocol that the AV follows here. In the simulation, the bicycles are running at the same speed while crossing, but in the practical field, that might not be the case. In addition, the timing of bicycle crossing is highly uncertain. Therefore, it will be more impactful if we approach in such a way where there will be no collision between AV and bicycles. Keeping this in
mind, we planned to modify the AV’s collision avoidance strategies. Here, after observing bicycles in the conflicting zone, the AV will seek a sufficient gap for crossing. If the actual gap is shorter than the accepted gap, the vehicle stops and waits for a certain period (0.5 to 2 seconds), then again it looks for the gap. In this way, even if there are more bicycles coming after the original waiting time, the AV should still manage to avoid collision.

Figure 2.15 Updated Flowchart of AV conflict-avoidance behavior at a bicycle crossing.
Our next plan was to design the AV in such a way where there will be no collision between AV and bicyclists. In this process, we planned to find out the gaps that will ensure full safety. We also plan to study various maneuvering speed of the AV to find out if it is a safe option or not. Moreover, we plan to design AV for various number of bicycles for multiple speeds, which will allow us to analyze the relationship between lost time and allowed gap, bicycle speed and AV’s maneuvering speed. As people is concerned about the safety issue of AV, this research will help us take a step further to achieve a safer environment where AV and vulnerable road users can co-exist.

Table 2.3 presents a comprehensive analysis of potential interactions between autonomous vehicles (AVs) and groups of bicyclists under varying speed conditions. The scenarios considered encapsulate three distinct speed profiles for both AVs and bicycles: low, medium, and high. The analysis extends to consider groups of up to 10 bicycles to capture the complexities involved when multiple bicycles are present.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Bike Speed (mph)</th>
<th>Allowed Gap (ft)</th>
<th>Lost Time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low Speed AV with 1 Low Speed Bike</td>
<td>10</td>
<td>30</td>
<td>3</td>
</tr>
<tr>
<td>Low Speed AV with 1 Medium Speed Bike</td>
<td>15</td>
<td>28</td>
<td>2.8</td>
</tr>
<tr>
<td>Low Speed AV with 1 High Speed Bike</td>
<td>20</td>
<td>25</td>
<td>2.5</td>
</tr>
<tr>
<td>Low Speed AV with 10 High Speed Bikes</td>
<td>20</td>
<td>50</td>
<td>7</td>
</tr>
<tr>
<td>Medium Speed AV with 1 Low Speed Bike</td>
<td>10</td>
<td>38</td>
<td>3.8</td>
</tr>
<tr>
<td>Medium Speed AV with 1 Medium Speed Bike</td>
<td>15</td>
<td>35</td>
<td>3.5</td>
</tr>
<tr>
<td>Medium Speed AV with 10 High Speed Bikes</td>
<td>20</td>
<td>60</td>
<td>8.5</td>
</tr>
<tr>
<td>High Speed AV with 1 Low Speed Bike</td>
<td>10</td>
<td>45</td>
<td>4.5</td>
</tr>
<tr>
<td>High Speed AV with 1 Medium Speed Bike</td>
<td>15</td>
<td>42</td>
<td>4.2</td>
</tr>
<tr>
<td>High Speed AV with 10 High Speed Bikes</td>
<td>20</td>
<td>70</td>
<td>10</td>
</tr>
</tbody>
</table>

Table 2.3 Autonomous Vehicle Interactions with Bicyclists at Varied Speeds.
From an overview, it's evident that as the number of bicyclists or their speed increases, the required allowed gap and lost time tend to augment, highlighting the intricacies involved in ensuring seamless and safe interactions. This table helps understanding the challenges and requirements of integrating autonomous vehicles into urban landscapes, where they might encounter multiple bicyclists with varying speeds.

2.6 Conclusions

This research investigated the interactions and collisions between autonomous vehicles and cyclists with conflicting movements at urban unsignalized intersections in a co-simulation environment. The simulation results on collisions and lost times with different AV gap acceptance and waiting time setup suggests that the balance between safety and efficiency may be important for AV behavior designs. Here, one pattern that may be noticed is that as waiting time and gap rise, the likelihood of an AV colliding with bicycles tends to decrease. On the other hand, as the gap and waiting time increase, so do the lost time for the ego car and other following vehicles.

The simulation scenario in this study was based on a CARLA built-in urban area map. Future research can develop more realistic 3D maps to reflect real-world scenarios so that simulation studies on specific real urban areas with the mixed traffic of AVs and other non-motorists are enabled. Moreover, the framework introduced in this research can expand to fuse real-time data to augment the simulations further. Various realistic environmental impacts, such as nighttime, foggy or rainy days, etc., can be incorporated in the co-simulation to study the impacted AV performance and the interactions among AVs and other road participants in future research. Further, the shifts between Level 4
full automation and human-intervened driving at the emergency can also be incorporated into future studies.
Chapter 3. A CARLA and SUMO Co-Simulation Study Between Autonomous Vehicles and Other Road Users

3.1 Introduction

As the world progresses toward the widespread adoption of autonomous vehicles (AVs), it becomes crucial to ensure that they incorporate seamlessly with the current transportation ecosystem. AVs must interact not only with other automated vehicles but also with diverse road users such as pedestrians, cyclists, and human-driven vehicles on complex roadways. Effectively navigating these interactions is crucial for the safety, effectiveness, and dependability of autonomous vehicles.

Understanding and predicting the complex interactions between these vehicles becomes crucial for their safe and effective deployment as their proliferation accelerates. The application of Chicken Game Theory, a strategic framework that models conflict escalation and resolution between self-interested parties, is a compelling method for addressing this challenge.

It is essential that AVs demonstrate not only technical proficiency, but also the ability to navigate the complexities of human behavior and decision-making. However, the application of Chicken Game Theory [34] to the domain of autonomous vehicles and their interactions with other road users presents a unique set of obstacles, making this topic exceedingly complex. While Chicken Game Theory has provided valuable insights for comprehending and predicting the complex interactions between autonomous vehicles and other road users, it is incontrovertible that the application of this framework presents numerous challenges. The extremely dynamic nature of microscopic traffic scenarios necessitates an approach that is more adaptable and data-driven. Consequently, surrogate
safety measures have emerged as a practical set of microscopic measurements for evaluating road safety in complex roadway environments, where the simplified payoff tables applied in conventional Chicken Game Theory may be too naïve to tackle the conflicting scenarios.

This study aims to explore the complex dynamics of multiple autonomous vehicles interacting on the road and demonstrates how the Chicken Game Theory can provide valuable insights into decision-making processes, risk assessment, and cooperative strategies for the simple two-player scenarios. Then we extend the discussion on scenarios with multiple players in urban intersection conflicting scenarios, where AV, traditional vehicles and cyclists may interact with each other so conventional Chicken Game and the corresponding analytical methods cannot apply. We then investigate the applicability of certain Surrogate Safety Measures in the context of interactions between autonomous vehicles and other road users in complex roadway environments. We will discuss the applicability of various SSMs, including time-to-collision, post-encroachment time, and minimum lateral distance in real time, for predicting and mitigating potential conflicts. In addition, we will investigate ways to emphasize the trade-off between safety and effectiveness. Through comprehensive analysis, we aim to contribute to the ongoing discussion surrounding the incorporation of autonomous vehicles into our existing transportation networks and shed light on the design of safer, more efficient, and more harmonious roadways for all road users.
3.2 Literature Review

Transportation researchers, urban planners, and policymakers have long been fascinated with the study of road user interactions. It is anticipated that the widespread use of autonomous vehicles would transform transportation, bringing about important advantages like better traffic flow, less congestion, and enhanced safety [35]. However, how AVs interact with other road users, such as human-driven cars and cyclists, is a crucial component of this transformation [36]. To create algorithms and laws that allow the efficient coexistence of AVs and other road users, it is imperative to comprehend these interactions [37]. The prediction of human driver behavior is a crucial component of comprehending AV interactions with human-driven vehicles [38]. For example, rule-based models, deep learning methods, and inverse reinforcement learning have all been created [39]. These techniques seek to infer in real-time the intents and deeds of human drivers, which can assist AVs in making wise choices during encounters [38]. According to Shalev-Shwartz et al. (2016), the relationship between AVs and human-driven vehicles might be either cooperative or competitive [40]. When two cars cooperate to complete a task, like merging or navigating an intersection, cooperative interactions take place [38]. Contrarily, in competitive encounters, both vehicles are attempting to gain an advantage, for as when passing or changing lanes [37]. Both sorts of interactions have been addressed by researchers through the development of game-theoretic strategies, decentralized control algorithms, and reinforcement learning [41, 42].

Predicting bicyclist behaviors is crucial for Autonomous Vehicles (AVs) to navigate safely and maintain a smooth traffic flow. Similar to how human drivers can be predicted, bicyclists must be predicted for AVs to navigate around them safely and
effectively [43]. According to Rasouli and Tsotsos (2018) [37], these models can aid AVs in anticipating the movements of bikers, enabling safer encounters. AVs must be able to safely coexist on the road with cyclists while preserving a smooth flow of traffic. Rule-based algorithms and reinforcement learning have been studied in this field with the goal of developing decision-making frameworks that satisfy these requirements [37].

In recent years, the implementation of game theory, specifically Chicken Game Theory, has provided invaluable insights into comprehending and predicting the complex decision-making processes in traffic scenarios. Rapoport and Chammah (1966) were among the first to apply Chicken Game Theory to the study of subject interactions [44]. Their findings paved the way for future studies on road user interactions. Fudenberg and Levine (1998) [45] and Elvik (2001) [46] investigated the significance of the Chicken Game Theory in comprehending pedestrian behavior at crosswalks. They investigated pedestrians' willingness to assert their right-of-way, weighing their personal safety against their desire to cross the road swiftly. Their research revealed critical insights into how pedestrians and drivers approach potential conflict situations strategically. Multiple researches [47, 48] extended the application of Chicken Game Theory to investigate bicycle-vehicle interactions, especially in cases where limited road space led to conflicts over right-of-way. They analyzed the decision-making process of both cyclists and motorists, focusing on the roles of risk perception, communication, and cooperation in the resolution of conflicts. Researchers also investigated the applicability of Chicken Game Theory to comprehending interactions between human-driven and autonomous vehicles as autonomous vehicles began to emerge [49, 50]. Their research emphasized the need for AVs to adapt their behavior to human decision-making patterns and investigated
cooperative strategies to prevent potential conflicts. The application of Chicken Game Theory to the analysis of complex roadways with multiple vehicles is limited in several ways. The Chicken Game Theory is based on a two-player paradigm that may oversimplify the actual complexities of roadway interactions involving multiple vehicles, pedestrians, and cyclists. The Chicken Game may not adequately convey the nuances of these multi-agent interactions, given that traffic situations typically involve more than two decision-making agents. The Chicken Game assumes that interactions between road users are transitive, so if a vehicle yields to another vehicle, and that vehicle yields to a third vehicle, then the first vehicle should also cede to the third vehicle. This presumption, however, does not always hold true in real-world traffic situations, which can involve numerous priorities, behaviors, and circumstances. The operation of autonomous vehicles is based on algorithms and computer programming, which may not exhibit the same tendencies as human drivers. The unique characteristics and decision-making processes of autonomous vehicles may not be completely captured by applying the Chicken Game to interactions involving these vehicles. Typically, Chicken Game Theory investigates single, non-iterative decision-making situations. However, real-world traffic scenarios frequently involve repeated interactions between road users, whose decisions may change based on past experiences or altered conditions. This limitation diminishes the applicability of the Chicken Game for analyzing long-term or recurrent interactions between road users.

An alternative method for analyzing interactions between autonomous vehicles and other road users in complex roadway environments is to use surrogate safety measures. These measures can be utilized as a compromise of the Chicken Game Theory
to circumvent the limitations of conventional game-theoretic approaches. Using the occurrence of prospective conflicts, Gettman and Head (2003) propose the Surrogate Safety Assessment Model (SSAM) to assess the safety of roadways [51]. The SSAM employs traffic simulation and conflict analysis to assess the efficacy of traffic control strategies and roadway layout. Laureshyn et al. (2010) [52] investigate the application of surrogate safety measures in traffic safety analysis, with an emphasis on their use in comprehending near-crash situations. The authors demonstrate how surrogate measures can be used to evaluate the safety of traffic scenarios without relying on actual accident data by utilizing traffic conflict techniques.

The use of simulation in surrogate safety measures (SSMs) research has grown in popularity because it enables researchers to assess traffic safety in a controlled setting without having to wait for real-world crash data. Through simulations, it is possible to estimate the likelihood of crashes in a variety of transportation systems, including how safely autonomous vehicles (AVs) operate in mixed traffic situations. By conducting a comparative analysis of traffic safety evaluation using microsimulation, Archer (2005) [53] developed this concept. The study examined various SSMs and demonstrated that microsimulation might be a potent tool for microscopic analysis of traffic safety. Researchers have recently used simulation to examine SSMs in particular traffic scenarios. Microsimulation was employed by Lee et al. (2012) [54] to study surrogate safety assessment in work zones, while video-based simulation data by Zangenehpour et al. (2016) [55] was used to assess the safety of signalized intersections with cycle tracks. Zheng et al. (2010) assess the safety of cooperative adaptive cruise control (CACC) systems using surrogate safety measures in their study [56]. The authors propose a
method for conflict prediction based on time-to-collision (TTC) and create a risk assessment framework for CACC systems. This study sheds light on the application of substitute safety measures for analyzing the safety performance of autonomous vehicle control systems in complex roadway environments. As autonomous vehicles have gained popularity, simulation has become increasingly important in evaluating their safety performance utilizing SSMs. Using microscopic simulation, one research [57] investigated the effect of AVs on traffic safety, and another one [58] examined substitute safety measures for lane change maneuvers carried out by AVs.

Although there has been significant progress in the study of how autonomous vehicles (AVs) interact with other road users, there are still several research gaps that must be filled in order to allow for the safe and effective integration of AVs into current transportation systems. Even while AV interactions with other road users have been the subject of several research, a thorough understanding of the safety consequences of these interactions is still difficult. To evaluate AV performance and create reliable control algorithms, more in-depth evaluations of probable conflicts and hazardous situations are required. The majority of current AV research is based on carefully controlled real-world testing, which might not accurately reflect the variety and uncertainties found in real-world traffic dynamics. Researchers can produce many and complex traffic scenarios that closely approximate actual traffic conditions by combining the microscopic traffic modeling capabilities of SUMO with the high-fidelity 3D urban environment of CARLA.

The goal of this research is:
a. Incorporating Chicken Game Theory into a comprehensive simulation framework to simulate the decision-making processes and strategies of two autonomous vehicles during their interactions on the road.

b. Establish a comprehensive simulation framework that effectively incorporates Surrogate Safety Measures to assess the safety performance of autonomous vehicles in their interactions with other road users.

c. Evaluate the efficiency of Time-to-Collision (TTC), Post-Encroachment Time (PET) in predicting the safety of autonomous vehicles and their interactions with other road users in various traffic scenarios.

3.3 Methods

SUMO [31] and CARLA [30] have been employed for depicting the environment, simulating automobile dynamics, and spawning the vehicular and cyclists traffic demand. CARLA offers an open-source, high-fidelity simulator made for the creation and testing of autonomous vehicle algorithms in various, realistic settings. It is appropriate for analyzing the strategic interactions between autonomous vehicles since it provides precise representations of vehicle dynamics, physics, and sensor models. On the other hand, SUMO is a potent traffic simulator that simulates a variety of traffic situations and aspects.

3.3.1 Chicken Game Theory

Two vehicles start traveling at a fast rate of speed toward one another in the game of chicken. Whoever avoids "chickening out" and swerving out of the way is the winner. The worst case scenario is that neither of the motorist swerves, and they both perish in the collision. As a result, one driver will get out because losing the game is better than
dying. A realistic setting for investigating the behavior of two autonomous vehicles in the framework of Chicken Game Theory is made possible by the co-simulation between CARLA and SUMO. Researchers can manipulate and watch the behavior of two autonomous vehicles in a game-theoretic setting by using a co-simulation between CARLA and SUMO. Understanding the implications of Chicken Game Theory in real-world traffic conditions requires thorough analysis of the strategies used by the autonomous vehicles and the effects of various factors on their interactions, which is only possible in this controlled environment.

Though our primary approach was to introduce multiple AVs in the simulation environment, it was found later that more than two AVs can not perform well in CARLA environment. Therefore, we decided to work on two AVs.
3.3.1.1 Scenario 1

The game of chicken could be viewed in the perspective of autonomous vehicles as two cars on a collision path with one another. Each vehicle has the option to either stay on course (continue straight) or swerve to avoid the collision. The goal in this case is usually to avoid collisions while maintaining efficiency.

Here's how the payoff matrix might look for two autonomous vehicles in a chicken-like situation:

<table>
<thead>
<tr>
<th></th>
<th>AV B stays</th>
<th>AV B swerves</th>
</tr>
</thead>
<tbody>
<tr>
<td>AV A stays</td>
<td>(Crash, Crash)</td>
<td>(Efficient, Inefficient)</td>
</tr>
<tr>
<td>AV A swerves</td>
<td>(Inefficient, Efficient)</td>
<td>(Safe, Safe)</td>
</tr>
</tbody>
</table>
3.3.1.2 Scenario 2

Two autonomous cars are approaching an intersection in this scenario; one wants to proceed straight while the other wants to make a left turn. Once more, each car has the option of either continuing as intended (going straight or turning left) or yielding to the opposing vehicle (stopping or slowing down). Here's how the payoff matrix might look for these two autonomous vehicles:

<table>
<thead>
<tr>
<th></th>
<th>AV B continues straight</th>
<th>AV B yields</th>
</tr>
</thead>
<tbody>
<tr>
<td>AV A turns left</td>
<td>(Crash, Crash)</td>
<td>(Efficient, Inefficient)</td>
</tr>
<tr>
<td>AV A yields</td>
<td>(Inefficient, Efficient)</td>
<td>(Safe, Safe)</td>
</tr>
</tbody>
</table>

3.3.1.3 Scenario 3

When one autonomous vehicle (AV) wants to go right, and the other wants to turn left, they're approaching each other at an intersection. Each vehicle has a choice: proceed with their planned direction (right or left), or yield (stop or slow down) to let the other vehicle pass.

Here's how the payoff matrix might look for these two autonomous vehicles:

<table>
<thead>
<tr>
<th></th>
<th>AV B turns right</th>
<th>AV B yields</th>
</tr>
</thead>
<tbody>
<tr>
<td>AV A turns left</td>
<td>(Crash, Crash)</td>
<td>(Efficient, Inefficient)</td>
</tr>
<tr>
<td>AV A yields</td>
<td>(Inefficient, Efficient)</td>
<td>(Safe, Safe)</td>
</tr>
</tbody>
</table>

Here are possible interpretations of the payoffs:
• (Crash, Crash): Both autonomous vehicles (AVs) continue with their planned turns without yielding, leading to a crash. This is a highly negative outcome, as it could potentially lead to damage to both vehicles and harm to passengers or pedestrians.

• (Efficient, Inefficient) and (Inefficient, Efficient): One AV yields (stops or slows down), and the other doesn't. The AV that continues with its planned direction maintains its efficiency, as it doesn't have to stop or slow down. The one that yields becomes inefficient, as it has to deviate from its original plan, potentially leading to increased travel time, higher energy consumption, etc.

• (Safe, Safe): Both AVs decide to yield, avoiding a crash. This is a safer outcome, as both vehicles avoid a collision. This might lead to some inefficiency in both vehicles due to the stop or slowdown, but the priority on safety is maintained.

3.3.2 Surrogate Safety Measures (SSM)

The surrogate safety measures that are analyzed in this study are: TTC, PET.

Time To Collision (TTC) is a safety indicator used in traffic analysis and engineering that represents the time it would take for two vehicles to collide if they continue on their current paths at their current speeds, assuming no evasive action is taken [59]

\[ TTC = \frac{d}{(V_f-V_l)}. \]

Here, \( d \) represents the current distance between the two vehicles (or a vehicle and any other obstacle) involved in the traffic scenario. \( V_f \) is the speed of the following
vehicle and \( V_1 \) is the speed of the leading vehicle or obstacle that the following vehicle may collide with.

Post Encroachment Time (PET) is a safety indicator used in traffic analysis to measure the time between the end of one vehicle's occupancy in a conflict zone and the start of another vehicle's occupancy in the same zone [60]. PET is used to evaluate the safety and efficiency of various traffic scenarios, such as intersections, roundabouts, or merging lanes. Lower PET values imply higher collision risk, while higher PET values suggest safer interactions.

The co-simulation between CARLA and SUMO combines the strengths of both simulators to create a comprehensive environment for studying autonomous vehicles and other road users. SUMO's traffic simulation capabilities enable the creation of large-scale traffic scenarios with numerous road users and complex traffic patterns. Combining this with CARLA's high-fidelity simulation of autonomous vehicles allows for the study of interactions between autonomous vehicles and other road users on a larger scale, enhancing the understanding of their behavior in complex roadway environments. The co-simulation of CARLA and SUMO enables the analysis of various Surrogate Safety Measures, such as Time-to-Collision (TTC), Post-Encroachment Time (PET). These measures can be evaluated in different traffic scenarios, providing valuable insights into the safety performance of autonomous vehicles and their interactions with other road users.

**Scenario**

The chosen traffic network will represent unsignalized urban intersections with no adverse weather and ambient daytime lighting. The roadways will have different posted
speed limit from 30 mph to up to 60 mph. There will also be surrounding traffic density for keeping the simulation scenario similar to the real-life situation.

The scenarios we plan to investigate are given below.

a. Intersection negotiation while a cyclist or manually driven vehicle is present: Determine whether the autonomous car can do so safely. The simulation can feature a bike or vehicle coming from one way and an autonomous car coming from another nearing an intersection. Whereas the CARLA simulation can control the movement of the autonomous vehicle, the SUMO traffic simulation can control the movement of the cyclist and vehicle. To prevent crashes, the autonomous vehicle must be able to recognize and react to the presence of the vehicle.

b. Test the ability of the autonomous vehicle to perform safely with a slower manually driven vehicle or bike in the roadway. The simulation can involve an autonomous vehicle driving on a road with a slow moving manually driven vehicle or bicycle in front of it. The SUMO simulation can simulate the manually driven vehicle or bicycle. The autonomous vehicle must detect the potential collision and maneuver safely to avoid any collisions.

3.4 Results

3.4.1 Chicken Game Theory

The table below represents a payoff matrix in the context of a game theoretical approach to autonomous vehicles interaction. Here, each cell in the tables represents the expected payoff or outcome when both autonomous vehicles (AV A and AV B) choose a particular strategy simultaneously.
In this case, the strategies available are either to Accelerate, Maintain Constant speed, Decelerate, or Stop, while AV A can choose to go Straight or Right and AV B can choose to go Left or Straight. The payoff seems to represent collision impulse, with lower values indicating better outcomes. Impulse is the quantity that describes how the collision between two ego vehicles changes their linear momentum.

**Table 3.1** Collision Impulse between Two AVs.

<table>
<thead>
<tr>
<th></th>
<th>AV B- LEFT</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Accelerate</td>
<td>Constant</td>
<td>Decelerate</td>
<td>Stop</td>
<td></td>
</tr>
<tr>
<td>AV B- LEFT</td>
<td>926.32</td>
<td>505.48</td>
<td>195.28</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>657.21</td>
<td>452.78</td>
<td>103.23</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>201.36</td>
<td>137.67</td>
<td>45.47</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>AV B- STRAIGHT</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Accelerate</td>
<td>Constant</td>
<td>Decelerate</td>
<td>Stop</td>
<td></td>
</tr>
<tr>
<td>AV B- STRAIGHT</td>
<td>1032.14</td>
<td>482.00</td>
<td>211.46</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>731.91</td>
<td>467.23</td>
<td>125.45</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>222.45</td>
<td>200.05</td>
<td>57.61</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>AV B- LEFT</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Accelerate</td>
<td>Constant</td>
<td>Decelerate</td>
<td>Stop</td>
<td></td>
</tr>
<tr>
<td>AV B- RIGHT</td>
<td>855.46</td>
<td>385.74</td>
<td>199.21</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>731.91</td>
<td>502.4</td>
<td>121.22</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>356.24</td>
<td>232.25</td>
<td>117.87</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

Below is the analysis for the given payoff matrix:

*AVA goes Straight and AV B goes Left*
The best strategy for both AVs in this case appears to be to stop, which results in a payoff of 0, which is the least costly scenario.

If both AVs decide to maintain a constant speed or decelerate, then the payoff is moderate, and collision risk may be balanced.

If both AVs or either one of them chooses to accelerate, it results in the highest payoff value, indicating high risk or potentially a severe collision.

**AV A goes Straight and AV B goes Straight**

- Again, stopping yields the least costly outcome with a payoff of 0.
- If either one or both of the AVs choose to accelerate, the outcome leads to the highest risk.
- If both maintain a constant speed or decelerate, we observe a moderate risk level.

**AV A goes Right and AV B goes Left**

- When both AVs stop, the payoff is again 0, which is the safest option.
- If both AVs maintain a constant speed, the payoff indicates a high risk, but slightly less compared to the other scenarios.
- Acceleration from either or both AVs leads to the most significant risk.

Overall, it's clear that acceleration generally leads to high-risk situations while stopping results in the safest outcomes. In all cases, the mutual decision to maintain a constant speed or to decelerate leads to a moderate level of risk. This analysis is assuming that the payoff is inversely proportional to safety.
3.4.2 Surrogate Safety Measure

"LV Speed" refers to the speed of the bicycle as leading vehicle (LV) in frames per second (fps), "AV Speed" refers to the speed of the autonomous vehicle (AV) also in fps, "Distance" is the initial distance between these two vehicles in feet (ft), "TTC" is the Time-to-Collision (which is the Distance divided by the relative speed difference between the two vehicles), and "Collision" refers to whether a collision will occur or not, depending on if the TTC is below a certain threshold. The following table shows the TTC and collision result for the interaction between a bicycle of 20 fps (13.6 mph) and AV of different speed.

Table 3.2 Time to Collision (AV and Bicycle).

<table>
<thead>
<tr>
<th>LV Speed (fps)</th>
<th>AV Speed (fps)</th>
<th>Distance (ft)</th>
<th>TTC</th>
<th>Collision</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>36</td>
<td>150</td>
<td>9.375</td>
<td>No</td>
</tr>
<tr>
<td>20</td>
<td>36</td>
<td>120</td>
<td>7.5</td>
<td>No</td>
</tr>
<tr>
<td>20</td>
<td>36</td>
<td>100</td>
<td>6.25</td>
<td>No</td>
</tr>
<tr>
<td>20</td>
<td>36</td>
<td>70</td>
<td>4.375</td>
<td>No</td>
</tr>
<tr>
<td>20</td>
<td>36</td>
<td>60</td>
<td>3.75</td>
<td>No</td>
</tr>
<tr>
<td>20</td>
<td>36</td>
<td>50</td>
<td>3.125</td>
<td>No</td>
</tr>
<tr>
<td>20</td>
<td>36</td>
<td>40</td>
<td>2.5</td>
<td>No</td>
</tr>
<tr>
<td>20</td>
<td>36</td>
<td>30</td>
<td>1.875</td>
<td>Yes</td>
</tr>
<tr>
<td>20</td>
<td>36</td>
<td>20</td>
<td>1.25</td>
<td>Yes</td>
</tr>
<tr>
<td>20</td>
<td>36</td>
<td>10</td>
<td>0.625</td>
<td>Yes</td>
</tr>
<tr>
<td>20</td>
<td>37</td>
<td>150</td>
<td>8.823529</td>
<td>No</td>
</tr>
<tr>
<td>20</td>
<td>37</td>
<td>120</td>
<td>7.058824</td>
<td>No</td>
</tr>
<tr>
<td>20</td>
<td>37</td>
<td>100</td>
<td>5.882353</td>
<td>No</td>
</tr>
<tr>
<td>20</td>
<td>37</td>
<td>70</td>
<td>4.117647</td>
<td>No</td>
</tr>
<tr>
<td>20</td>
<td>37</td>
<td>60</td>
<td>3.529412</td>
<td>No</td>
</tr>
<tr>
<td>Speed (AV)</td>
<td>Speed (LV)</td>
<td>Distance (m)</td>
<td>TTC (s)</td>
<td>Result</td>
</tr>
<tr>
<td>-----------</td>
<td>-----------</td>
<td>--------------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>20</td>
<td>37</td>
<td>50</td>
<td>2.941176</td>
<td>No</td>
</tr>
<tr>
<td>20</td>
<td>37</td>
<td>40</td>
<td>2.352941</td>
<td>Yes</td>
</tr>
<tr>
<td>20</td>
<td>37</td>
<td>30</td>
<td>1.764706</td>
<td>Yes</td>
</tr>
<tr>
<td>20</td>
<td>37</td>
<td>20</td>
<td>1.176471</td>
<td>Yes</td>
</tr>
<tr>
<td>20</td>
<td>37</td>
<td>10</td>
<td>0.588235</td>
<td>Yes</td>
</tr>
<tr>
<td>20</td>
<td>40</td>
<td>150</td>
<td>7.5</td>
<td>No</td>
</tr>
<tr>
<td>20</td>
<td>40</td>
<td>120</td>
<td>6</td>
<td>No</td>
</tr>
<tr>
<td>20</td>
<td>40</td>
<td>100</td>
<td>5</td>
<td>No</td>
</tr>
<tr>
<td>20</td>
<td>40</td>
<td>70</td>
<td>3.5</td>
<td>No</td>
</tr>
<tr>
<td>20</td>
<td>40</td>
<td>60</td>
<td>3</td>
<td>No</td>
</tr>
<tr>
<td>20</td>
<td>40</td>
<td>50</td>
<td>2.5</td>
<td>Yes</td>
</tr>
<tr>
<td>20</td>
<td>40</td>
<td>40</td>
<td>2</td>
<td>Yes</td>
</tr>
<tr>
<td>20</td>
<td>40</td>
<td>30</td>
<td>1.5</td>
<td>Yes</td>
</tr>
<tr>
<td>20</td>
<td>40</td>
<td>20</td>
<td>1</td>
<td>Yes</td>
</tr>
<tr>
<td>20</td>
<td>40</td>
<td>10</td>
<td>0.5</td>
<td>Yes</td>
</tr>
<tr>
<td>20</td>
<td>45</td>
<td>150</td>
<td>6</td>
<td>No</td>
</tr>
<tr>
<td>20</td>
<td>45</td>
<td>120</td>
<td>4.8</td>
<td>No</td>
</tr>
<tr>
<td>20</td>
<td>45</td>
<td>100</td>
<td>4</td>
<td>No</td>
</tr>
<tr>
<td>20</td>
<td>45</td>
<td>70</td>
<td>2.8</td>
<td>Yes</td>
</tr>
<tr>
<td>20</td>
<td>45</td>
<td>60</td>
<td>2.4</td>
<td>Yes</td>
</tr>
<tr>
<td>20</td>
<td>45</td>
<td>50</td>
<td>2</td>
<td>Yes</td>
</tr>
<tr>
<td>20</td>
<td>45</td>
<td>40</td>
<td>1.6</td>
<td>Yes</td>
</tr>
<tr>
<td>20</td>
<td>45</td>
<td>30</td>
<td>1.2</td>
<td>Yes</td>
</tr>
<tr>
<td>20</td>
<td>45</td>
<td>20</td>
<td>0.8</td>
<td>Yes</td>
</tr>
<tr>
<td>20</td>
<td>45</td>
<td>10</td>
<td>0.4</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Based on the data achieved from the simulation, the Time to Collision (TTC) can be analyzed for various scenarios. The table indicates that the TTC decreases as the autonomous vehicle (AV) speed increases, holding the lead vehicle (LV) speed constant,
and the initial distance to the LV decreases. The TTC decreases as the distance between AV and LV decreases, which is to be expected. For instance, at the same speed of 36 fps for AV and 11 fps for LV, TTC decreases from 6 seconds at 150 feet distance to 0.4 seconds at 10 feet distance.

For all scenarios, a collision seems to be associated with a TTC of less than 3 seconds. This critical threshold can be used to implement preventive actions in an autonomous driving system.

**Table 3.3** Time to Collision (AV and Manually Driven Vehicle).

<table>
<thead>
<tr>
<th>LV Speed (fps)</th>
<th>AV Speed (fps)</th>
<th>Distance (ft)</th>
<th>TTC</th>
<th>Collision</th>
</tr>
</thead>
<tbody>
<tr>
<td>36</td>
<td>45</td>
<td>200</td>
<td>22.2222</td>
<td>No</td>
</tr>
<tr>
<td>36</td>
<td>45</td>
<td>180</td>
<td>20</td>
<td>No</td>
</tr>
<tr>
<td>36</td>
<td>45</td>
<td>150</td>
<td>16.6667</td>
<td>No</td>
</tr>
<tr>
<td>36</td>
<td>45</td>
<td>120</td>
<td>13.3333</td>
<td>No</td>
</tr>
<tr>
<td>36</td>
<td>45</td>
<td>100</td>
<td>11.1111</td>
<td>No</td>
</tr>
<tr>
<td>36</td>
<td>45</td>
<td>80</td>
<td>8.888889</td>
<td>No</td>
</tr>
<tr>
<td>36</td>
<td>45</td>
<td>50</td>
<td>5.555556</td>
<td>No</td>
</tr>
<tr>
<td>36</td>
<td>45</td>
<td>20</td>
<td>2.222222</td>
<td>Yes</td>
</tr>
<tr>
<td>36</td>
<td>50</td>
<td>200</td>
<td>14.28571</td>
<td>No</td>
</tr>
<tr>
<td>36</td>
<td>50</td>
<td>180</td>
<td>12.85714</td>
<td>No</td>
</tr>
<tr>
<td>36</td>
<td>50</td>
<td>150</td>
<td>10.71429</td>
<td>No</td>
</tr>
<tr>
<td>36</td>
<td>50</td>
<td>120</td>
<td>8.571429</td>
<td>No</td>
</tr>
<tr>
<td>36</td>
<td>50</td>
<td>100</td>
<td>7.142857</td>
<td>No</td>
</tr>
<tr>
<td>36</td>
<td>50</td>
<td>80</td>
<td>5.714286</td>
<td>No</td>
</tr>
<tr>
<td>36</td>
<td>50</td>
<td>50</td>
<td>3.571429</td>
<td>No</td>
</tr>
<tr>
<td>36</td>
<td>50</td>
<td>20</td>
<td>1.428571</td>
<td>Yes</td>
</tr>
<tr>
<td>36</td>
<td>55</td>
<td>200</td>
<td>10.52632</td>
<td>No</td>
</tr>
<tr>
<td>36</td>
<td>55</td>
<td>180</td>
<td>9.473684</td>
<td>No</td>
</tr>
<tr>
<td>36</td>
<td>55</td>
<td>150</td>
<td>7.894737</td>
<td>No</td>
</tr>
<tr>
<td>36</td>
<td>55</td>
<td>120</td>
<td>6.315789</td>
<td>No</td>
</tr>
<tr>
<td>LV Speed</td>
<td>AV Speed</td>
<td>Distance</td>
<td>TTC</td>
<td>Collision</td>
</tr>
<tr>
<td>----------</td>
<td>----------</td>
<td>----------</td>
<td>-----------</td>
<td>-----------</td>
</tr>
<tr>
<td>36</td>
<td>55</td>
<td>100</td>
<td>5.263158</td>
<td>No</td>
</tr>
<tr>
<td>36</td>
<td>55</td>
<td>80</td>
<td>4.210526</td>
<td>No</td>
</tr>
<tr>
<td>36</td>
<td>55</td>
<td>50</td>
<td>2.631579</td>
<td>Yes</td>
</tr>
<tr>
<td>36</td>
<td>55</td>
<td>20</td>
<td>1.052632</td>
<td>Yes</td>
</tr>
<tr>
<td>36</td>
<td>60</td>
<td>200</td>
<td>8.333333</td>
<td>No</td>
</tr>
<tr>
<td>36</td>
<td>60</td>
<td>180</td>
<td>7.5</td>
<td>No</td>
</tr>
<tr>
<td>36</td>
<td>60</td>
<td>150</td>
<td>6.25</td>
<td>No</td>
</tr>
<tr>
<td>36</td>
<td>60</td>
<td>120</td>
<td>5</td>
<td>No</td>
</tr>
<tr>
<td>36</td>
<td>60</td>
<td>100</td>
<td>4.166667</td>
<td>No</td>
</tr>
<tr>
<td>36</td>
<td>60</td>
<td>80</td>
<td>3.333333</td>
<td>Yes</td>
</tr>
<tr>
<td>36</td>
<td>60</td>
<td>50</td>
<td>2.083333</td>
<td>Yes</td>
</tr>
<tr>
<td>36</td>
<td>60</td>
<td>20</td>
<td>0.833333</td>
<td>Yes</td>
</tr>
</tbody>
</table>

From the dataset, it can be inferred that as the speed difference between the Leading Vehicle (LV) and Autonomous Vehicle (AV) increases, the likelihood of a collision increases. This is because, when AV is faster, it has less time to react and stop or avoid the LV. The likelihood of a collision decreases as the initial distance between LV and AV increases, for a given set of speeds. More distance allows more time for the AV to react and adjust its speed to avoid a collision. TTC (Time-to-Collision) appears to be a critical factor in determining the possibility of a collision. In your dataset, collisions are reported when the TTC is less than or equal to 4. Therefore, the AV can be programmed to consider it would crash if it can't stop or avoid the obstacle within this timeframe.

The analysis indicates that the interaction between AV speed, LV speed, and distance plays a vital role in the TTC and the likelihood of a collision. Future research could investigate ways to optimize these parameters to maximize safety in autonomous driving scenarios.
Based on this result, we designed the AV in such a way that it would brake once it reaches the TTC value 4.3, and after running the simulation for both AV-bicycle and AV-other vehicles interaction, it was found that no collision occurred under this condition.

**Table 3.4** PET (Post Encroachment Time).

<table>
<thead>
<tr>
<th>AV Speed (fps)</th>
<th>PET</th>
<th>Collision</th>
</tr>
</thead>
<tbody>
<tr>
<td>44</td>
<td>0.1</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>0.2</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>0.3</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>0.4</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>0.5</td>
<td>No</td>
</tr>
<tr>
<td>50</td>
<td>0.1</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>0.2</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>0.3</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>0.4</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>0.5</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>0.6</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>0.7</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>0.8</td>
<td>No</td>
</tr>
<tr>
<td>60</td>
<td>0.1</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>0.2</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>0.3</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>0.4</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>0.5</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>0.6</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>0.7</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>0.8</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>0.9</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>1.1</td>
<td>No</td>
</tr>
<tr>
<td>70</td>
<td>0.1</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>0.2</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>0.3</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>0.4</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>0.5</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>0.6</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>0.7</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>0.8</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>0.9</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>1.1</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>1.2</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>1.3</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>1.4</td>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>75</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
</tr>
<tr>
<td>0.2</td>
</tr>
<tr>
<td>0.3</td>
</tr>
<tr>
<td>0.4</td>
</tr>
<tr>
<td>0.5</td>
</tr>
<tr>
<td>0.6</td>
</tr>
<tr>
<td>0.7</td>
</tr>
<tr>
<td>0.8</td>
</tr>
<tr>
<td>0.9</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>1.1</td>
</tr>
<tr>
<td>1.2</td>
</tr>
<tr>
<td>1.3</td>
</tr>
<tr>
<td>1.4</td>
</tr>
<tr>
<td>1.5</td>
</tr>
<tr>
<td>1.6</td>
</tr>
<tr>
<td>1.7</td>
</tr>
</tbody>
</table>
Here are some insights derived from the data:

1. **Effect of Speed on PET Threshold:** As the speed of the AV increases, the PET value that can prevent a collision also increases. This makes sense as at higher speeds, more time may be needed to perceive and react to an encroaching vehicle.

2. **AV Speed and Collision:** Higher AV speeds are associated with a higher likelihood of collision, which aligns with general expectations. As the speed increases, it may be more difficult to avoid a collision due to longer stopping distances and less reaction time.

3. **PET and Collision:** Lower PET values are associated with a higher likelihood of collision. For example, at all given speeds, a PET of 0.1 results in a collision. This could suggest that there is insufficient time for the AV to react and adjust its path or speed to prevent the collision.

4. **Specific PET thresholds:** For an AV speed of 44 fps, a collision is avoided when PET is at least 0.5. At 50 fps, PET needs to be at least 0.8 to avoid a collision. At 60 fps, PET must be at least 1.1. At 70 fps, PET must be 1.4 or more, and at 75 fps, PET needs to be at least 1.7. These PET thresholds highlight the increasing time required for safe interaction as the speed of the vehicle increases.

Based on this result, we designed the AV in such a way that it would brake once it reaches the PET value 1.7, and after running the simulation for both AV-bicycle and AV-other vehicles interaction, it was found that no collision occurred under this condition.
These are just inferences based on the results from the simulation. Real-life situations can be much more complex and require the autonomous vehicle to account for many more variables like road conditions, weather, visibility, vehicle condition, and behaviors of other road users.

3.5 Conclusions

This research presented an innovative study that leverages the CARLA and SUMO co-simulation environment to analyze the interactions between autonomous vehicles and other road users. The study provided new insights into how autonomous vehicles might behave and interact with other AVs in a variety of situations by combining ideas from game theory, notably the Chicken Game Theory. Surrogate safety measures have proven to be useful in assessing how well autonomous cars perform in terms of safety. These measures made it possible to understand the circumstances that can possibly result in collisions or near-miss incidents in more detail. Given their capability for quick, real-time decision-making and emotionless response to stimuli, our findings point to a great potential for autonomous vehicles to improve overall road safety.

The successful integration of autonomous vehicles into current traffic networks, however, necessitates not only cutting-edge algorithms but also a thorough understanding of human driving patterns. Future research efforts should concentrate on the dynamics of human-driver interactions and responses since they have the potential to greatly affect how autonomous cars behave.

The importance of simulation environments like CARLA and SUMO in the context of autonomous vehicle research is also highlighted by this study. They make it possible for us to evaluate a wide range of events and settings in a secure, controlled
setting, which is not possible during real-world tests. Such simulation tools will become more and more important as the technology underlying autonomous vehicles develops in order to comprehend the dynamics of interaction between autonomous vehicles and other road users. Future research should concentrate on enlarging these models to accommodate more complex traffic behavior and cover a wider range of traffic circumstances. The conclusions will also be more reliable and broadly applicable if they take into account the disturbances and uncertainties that exist in the actual world, such as weather, road conditions, and unexpected human driving behaviors.

In conclusion, this study emphasizes the significance of a multifaceted approach to studying autonomous vehicles, one that combines cutting-edge simulation tools, game theory ideas, and substitute safety measures, in order to effectively predict, comprehend, and improve their interactions with other road users.
Chapter 4. Analyzing the Impact of Penetration Rates of Autonomous Vehicle in an Urban Environment—A SUMO Simulation Study

4.1 Introduction

Researchers and practitioners can better understand and improve real-world performance and safety by analyzing the interactions, behavior, and effects of a significant number of autonomous cars interacting with conventional vehicles, pedestrians, and bicycles. Research on autonomous vehicles started with a single autonomous vehicle in simple scenarios. When comparing the study of several autonomous vehicles in a mixed traffic metropolitan environment to that of a single autonomous vehicle, several advantages emerge.

Researchers can more accurately simulate real-world traffic situations by observing numerous autonomous vehicles interacting with one another, as well as with cyclists, human-driven cars, and other vehicles on the road. This helps them better comprehend the intricacies and difficulties of urban traffic. Real-world traffic is not composed of isolated vehicles; instead, it's a dynamic environment where various entities interact, sometimes unpredictably. To truly understand how autonomous vehicles will perform in real-world conditions, it's essential to study them in scenarios that closely mimic these interactions. By observing numerous autonomous vehicles in mixed traffic situations, researchers can gather more comprehensive data about potential challenges and hazards, leading to more robust and safer autonomous systems. Observing multiple autonomous vehicles allows for a broader range of scenarios to be studied. These vehicles can mimic the unpredictability of human-driven cars, making the simulation more representative of actual conditions. Moreover, when these autonomous vehicles interact
with cyclists, human drivers, and other road entities, researchers can better understand potential challenges and design solutions for safer integration into mixed traffic environments. It is possible to develop more robust and dependable autonomous systems by researching several autonomous vehicles in situations with mixed traffic. Policymakers and regulators can learn a lot about how multiple autonomous vehicles interact with human-driven vehicles and other road users by doing this. In order to enable the effective and safe integration of autonomous vehicles into the urban transportation ecology, this may help define future legislation and regulations.

Even if there are now a small number of autonomous vehicles (AVs) on the road, conducting simulation experiments with a higher penetration rate of AVs on the road can be helpful in comprehending the potential effects and implications of widespread AV adoption. Researchers can assess how AVs might affect traffic flow and efficiency, relieve congestion, and maximize the use of road space with the use of simulations with increased AV penetration rates. Policymakers, urban planners, and transportation engineers can decide on essential infrastructure changes or new road designs to facilitate the widespread use of AVs by studying the implications of high AV penetration rates. Researchers can discover possible safety issues and fix them by enhancing AV algorithms and technology to reduce risks and enhance overall road safety by simulating various traffic scenarios with a high percentage of AVs. In order to help guide environmental legislation and sustainability initiatives, high AV penetration simulations can offer insights into potential changes in fuel consumption and greenhouse gas emissions that may arise from the widespread use of AVs. Policymakers can find the regulations and rules necessary to enable the safe, effective integration of AVs into the
transportation system by running simulations with higher AV penetration rates. Researchers can better understand how human drivers may interact with AVs and devise techniques to improve overall road safety during the transition phase when both human-driven and autonomous cars use the road by simulating a variety of scenarios with high AV penetration rates. In conclusion, conducting simulation tests with higher AV penetration rates can offer useful information and insights to a variety of stakeholders, enabling them to make knowledgeable decisions regarding the future of transportation and the introduction of autonomous vehicles onto the road.

4.2 Literature Review

Numerous studies on AVs focus on the technologies related to their hardware. In urban settings, autonomous vehicles (AVs) act differently from conventional vehicles, mostly due to their internal control algorithms, sensor systems, and decision-making procedures. AVs use cutting-edge sensors and artificial intelligence (AI) to navigate around the urban environment, whereas human drivers rely on their instincts, experience, and understanding of traffic laws. To understand their environment, AVs use a variety of sensors, including LiDAR, cameras, radar, and ultrasonic sensors. They enable smoother and safer operation by precisely identifying, tracking, and forecasting the motions of pedestrians, bicycles, and other vehicles [61, 62]. In order to plan their routes, AVs use sophisticated algorithms that consider traffic patterns, road regulations, and other dynamic factors. These algorithms can optimize for multiple goals, such as lowering energy use or trip time, leading to behavior that differs from that of human-driven vehicles [63, 64].
Through the use of machine learning techniques, AVs may continuously improve their behavior by learning from the massive amounts of data gathered from numerous sensors. In contrast to human drivers, who may have a limited capacity for learning and adaptation, this can result in more effective and safe driving in complex urban situations [65, 66]. To enhance traffic flow and safety, connected autonomous vehicles (CAVs) can communicate with one another and the local infrastructure. CAVs can coordinate their operations and make better decisions than human drivers by exchanging real-time information about their status, position, and intentions [67].

The general outlook of many autonomous vehicles in urban settings is complex and is constantly changing as technology develops and is more often used. People's opinions are influenced by a variety of things, including their knowledge of the technology, confidence in automation, individual experiences, media coverage, and the advantages and disadvantages of autonomous vehicles. The safety of driverless vehicles is a major societal concern, particularly in dense urban contexts. In particular, when interacting with pedestrians, cyclists, and human-driven cars, people are concerned about these vehicles' ability to make wise decisions in real-time [68]. One of the most important variables affecting public opinion is trust in the technology. Many individuals still have doubts about how well-built and effective autonomous vehicles are, as well as how well-equipped they are to deal with unforeseen circumstances [69]. Nevertheless, research has demonstrated that exposure to and familiarity with the technology can boost trust and acceptability [70]. The privacy and security of the data that autonomous vehicles acquire and share, including personal data and location data, is also a worry for the general public. The possible effects on traffic, congestion, and pollutants have an impact on how
people view autonomous vehicles. According to several research, the public has a positive outlook on the potential advantages of autonomous vehicles in terms of increasing traffic efficiency and lowering environmental impact [71]. The potential influence of autonomous vehicles on transportation fairness and accessibility, particularly for underserved groups and individuals with disabilities, is another factor influencing public perception [72]. Public perception may also be impacted by worries about employment loss brought on by the introduction of autonomous vehicles, particularly in sectors like taxi and haulage [73].

Although there has been a growing body of research on how people feel about autonomous vehicles in recent years, the study on public perception is still in its infancy. The public's views and perceptions are expected to change as a result of the technology's continued rapid advancement. The research done thus far has shed vital light on the aspects that influence public opinion, such as demographics, perceived benefits, safety, trust, familiarity, and trustworthiness. To completely comprehend and measure the public's perspective of autonomous vehicles, further research must be done.

By gathering information on the security, effectiveness, and efficiency of these cars in a controlled virtual environment, simulation studies might indirectly aid in quantifying public perception of multiple autonomous vehicles in metropolitan contexts. Even though simulation studies don't test public perception directly, they can offer insights into a range of factors that might affect public opinion. Multiple autonomous cars interacting with other road users, such as pedestrians, bicycles, and human-driven vehicles, can be evaluated in terms of their safety performance using simulations. Multiple autonomous cars can have an impact on traffic flow, congestion, trip duration,
and pollutants, which can be measured through simulation studies. These measurements can assist illustrate the potential advantages of autonomous vehicles in terms of the impact they have on the environment and traffic efficiency, which may help change people's perceptions of them.

Complex urban traffic situations can be simulated in studies, which enables researchers to examine how various autonomous vehicles interact with diverse road users under various circumstances. These realistic simulations can help address public concerns about the ability of autonomous vehicles to navigate complex urban environments and may help build trust in the technology. Although simulation studies can't directly record user experiences, they can offer information on the capabilities and performance of autonomous cars, which can help user experience research. A more complete knowledge of public perception can be provided by fusing simulation data with actual user experience assessments.

It is important to remember that, while simulation studies might offer insightful information on a variety of characteristics of many autonomous cars operating in urban settings, they cannot assess public opinion. A combination of polls, interviews, focus groups, and research of actual user experiences should be used to properly comprehend and measure public perception. Nevertheless, simulation studies can offer crucial information about the performance of autonomous vehicles that can assist allay public worries and gradually change public perception.

The rapid development of vehicle automation and the anticipated arrival of shared mobility via completely autonomous vehicles herald the start of a new era of mobility that has the potential to drastically alter the way people travel in cities in the future.
Communities need to get ready to adapt to these advances in light of the potential changes they may bring about. Thus, to investigate the effects of various market penetration rates of shared autonomous vehicles (SAVs) on a city-sized traffic system, the traffic flow theory, simulation-based dynamic traffic assignment, and a computer experiment utilizing PTV Visum software were all used in a study [74]. Budapest was selected as the case study city, and a simulation model was made by incorporating SAV components and their relationships into the city's existing traffic model [74]. Three potential future penetration rates were then compared to five key performance indicators (KPIs) [76]. The outcomes of the simulation showed that the SAV system's deployment improves traffic efficiency [74]. Based on the correlations between the modeled SAV demand shares and the network's KPIs in the planned scenarios, both the SAV demand share and overall network performance improved [74].

Full autonomy is a long way off, despite the excitement in the media, research, and industry. Autonomous cars have the potential to drastically cut parking demand in cities [10] and reduce traffic casualties by over 90% [75, 76]. It is yet unknown, though, how widely adopted they will be, how they will interact with other urban infrastructures, and what effect this will have on urban form [77, 78]. It is evident that there will be a phase of mixed traffic, in which human-driven and autonomous vehicles will share the road and negotiate traffic, between degrees zero and five of autonomy, which correspond to drivers having complete control over their vehicles and being fully autonomous and capable of operating in any driving scenario without human intervention at any point. Indeed, vehicles with varying degrees of autonomy are already commonplace; for example, when certain functions respond based on information about the driving
environment (degree two), or when the driver needs to be prepared to take over the wheel (degree one).

Hörl [79] offers an addition to the MATSim simulator that enables the modeling of driverless taxis. The author [79] employed a scenario with 84,000 individuals in Sioux Falls, USA, to test the extension by modeling travels by vehicle, taxi, bus, and foot. A utility function based on waiting time and price was suggested by the author [79]. The findings indicated that a fleet of 1000 AVs would be required to meet the significant demand for this new service, with over 45% of all journeys and a large number of people switching from private vehicles and public transportation to the AV service [79]. A model to replicate an AV on-demand service in Melbourne, Australia, is presented by Dia and Javanshour [80]. The concept was for customers to use shared automobiles with a five-minute waiting time limit in place of individual vehicles [80]. According to the simulation, 247 cars could replace all 2047 of the original private cars, which would mean 88% fewer cars on the road and 83% less parking space needed at the given waiting time [80]. One drawback was that the vehicles' empty excursions to get to the next passenger resulted in an almost 10% increase in the overall kilometers traveled [80]. Bischoff and Maciejewski [81] created a simulation of Berlin in which autonomous taxis took the place of all privately owned vehicles. A normal workday serves as the basis for the imagined scenario [81]. The algorithm they developed to dispatch the cabs based on supply and demand balancing is their most significant contribution [82]. The number of cars in the simulation was increased from 50,000 to 25,000 by the authors [82]. The findings demonstrated that 110,000 autonomous taxis, with an average wait time of less than three minutes, could replace the city's 1.1 million fleet of taxis [82]. In Halifax,
Canada, Alam and Habib [83] created a simulation using shared autonomous vehicles (SAVs). 57,694 trips in the morning peak were taken into account in the study [83] using the VISSIM simulator. Four different fleet sizes—450, 900, 1800, and 3600 SAVs—were taken into consideration [83]. According to the data, 15% of the journeys in the first scenario were completed by SAVs, and that number rose to 65% in the final scenario [83]. The simulation also revealed a drop in the percentage of journeys made in traditional cars, from 25% to 47%. The study [83] did find, however, that the empty journeys taken by AVs resulted in longer travel times and a higher total distance traveled.

A few studies assess the combined use of human-driven and autonomous vehicles in traffic. For instance, Chen et al. [84] provide a cellular automaton model in which the HVs halt at random while the AVs are aware of the placements and speeds of the cars in front of them. This study discovered that the HVs' slowing has a significant effect on the flow of general traffic and that the AVs should be able to detect the status of at least five cars ahead in order to optimize the flow. A cellular automaton model was also created by Liu et al. [85] to assess the effects of lane change algorithms on mixed traffic. To determine the ideal time to execute the move, the AVs in the model can interact with other AVs that are in the target lane. A three-lane freeway was used in the scenario, and various AV rates (0%, 40%, 80%, and 100%) were simulated. The findings indicated that there may be a more than 70% improvement in the maximum vehicle flow on the road, from 2000 to 3500 vehicles per hour. An example of a mixed-traffic microsimulation on an Irish motorway is presented by Rezaei and Caulfield [86]. Starting with no AVs, the scenarios increased the share of AVs by 10% until all vehicles were AVs. The simulation demonstrated a significant increase in the share of autonomous vehicles (AVs) in traffic,
particularly in terms of shorter wait times and fewer vehicle stops [86]. It also
demonstrated that the ideal AV share is 60% [86]. Potential conflicts in intersections for
varying rates of autonomous vehicle penetration were assessed by Virdi et al. [87].
According to the findings, low AV penetration rates (20%) might lead to higher conflicts
at signalized crossings but lower ones at priority intersections (such round-abounds) [87].
High penetration rates, on the other hand, can significantly reduce conflicts at all
crossroads [87]. Additionally, some publications model platoons in mixed traffic. In order
to reduce pollution produced by vehicle acceleration, Zhao et al. [88] described the usage
of vehicle platoons for mixed automated and human-driven cars at one signalized
intersection. The authors created an optimization model where the AVs lead their platoon
and try to use the least amount of fuel while having the least amount of an effect on the
journey time of the other vehicles [90]. The goal of the model [88] is to regulate the
vehicles’ speed so that they don't stop at red lights. The study [88] outlines five
simulation scenarios including 20%, 40%, 60%, 80%, and 100% of autonomous vehicles.
The findings demonstrated a significant decrease in fuel usage from 20% to 60% and
only a negligible effect in the final two scenarios [88]. From the first to the second case,
the journey time reduced before stabilizing [88]. An AV and HV cooperative platoon
system is proposed by Gong and Du [89]. According to their model [89], AVs behave
normally smoothly and use online learning to forecast and adjust to the behavior of HVs.
According to the authors, HVs will soon be found inside networked AV platoons [89].
Their approach was able to effectively reduce the spread of traffic oscillations and
stabilize the platoon flow as demonstrated by numerical simulation [89]. Due to the
deployment of AVs in urban or highway situations, all of the works show favorable
effects on traffic flow [89]. They typically, however, employ a fictitious setting with a single road [89]. In conclusion, it is evident that a variety of urban and highway scenarios have been simulated in the literature, generally demonstrating the substantial advantages that arise from the deployment of AVs and platoons in mixed-traffic situations.

The research gaps of these studies are:

Studies often consider single autonomous vehicle in isolation, without taking into account how they interact with existing or future traffic. Also, many simulation studies lack validation against real-world empirical data, making it challenging to ensure that the simulated environments accurately represent actual conditions. Many simulations focus on a small, controlled area and do not address how the findings would scale to a larger urban environment with a more complex road network.

In this study, we aim to explore the impact of different penetration rates of autonomous vehicles in a real world area by utilizing the state-of-the-art traffic simulation platform, Simulation of Urban Mobility (SUMO). By simulating a range of AV adoption scenarios in a real-world urban environment, this research aims to provide valuable insights into the possible benefits and challenges that widespread AV implementation may bring to the transportation landscape of Huntsville and beyond.

In this research, we will address the following research questions:

From simulations, what impact does the introduction of autonomous vehicles have on the overall traffic speed in an urban environment?

Are there observable improvements in traffic safety with higher penetration rates of AVs?
To address these questions, we will simulate a range of traffic scenarios with varying AV penetration rates, from minimal presence to full adoption. The simulation will generate data on average speed, queue length and near-miss incident. This dataset will allow us to analyze the potential benefits and challenges associated with the widespread implementation of AVs.

4.3 Methodology

We attempted to study how increasing numbers of self-driving cars (or Autonomous Vehicles, AVs) might affect traffic. We looked at scenarios from no AVs at all, up to all cars being self-driving. Our main focus was on average driving speed, the length of car lines at stoplights, and near-miss accidents. The resulting data was then statistically analyzed using a two-tailed t-test to discern the significance of AV-induced changes in these indicators.

In order to replicate genuine traffic demand and mobility patterns, we chose to base our scenario on the road network of a real world city. Because of its topology's similarity to that of many other US cities and the accessibility of traffic statistics that may be used to calibrate traffic demand, the Downtown area of City of Huntsville, Alabama (Figure 1) was our choice. Another benefit is that for a microscopic simulator, its size is reasonable in terms of complexity.

The initial dataset is extracted from OSM and it provides many of the data required to build the Huntsville Downtown scenario. OSM is a collection of crowd-sourced information and the accuracy is not always consistent (Mordechai, 2010). A free and editable map of the entire planet is the goal of the open-source, collaborative initiative known as OpenStreetMap (OSM) (www.openstreetmap.org). Comprehensive
information about roads, structures, land use, tourist attractions, and other geographical aspects is contained in OSM data. It is a dependable and current source of geographic data since it is continuously updated and improved by a global community of volunteers.

For a variety of purposes, including transportation and urban planning, this comprehensive dataset offers useful information. Simulation of Urban Mobility (SUMO) is a popular open-source, microscopic, and multi-modal traffic simulation software designed to simulate traffic flow, public transportation, and urban mobility scenarios. In order to use real-world map data for precise and realistic traffic simulations, OSM data must be converted to SUMO network files.
Scenarios

To comprehensively evaluate the impact of different levels of autonomous vehicle (AV) penetration in an urban environment, we designed a series of simulation scenarios. These scenarios range from a baseline case, with no AVs, to full AV penetration, allowing for an incremental assessment of the system's performance with respect to key metrics.

The baseline scenario serves as a control for our study, comprising exclusively of human-driven vehicles. This provides a foundational reference point against which the impact of introducing various proportions of AVs into the traffic system can be evaluated. To establish a benchmark for the performance of Autonomous Vehicles (AVs) in urban environments, it is critical to first understand the behavior of traditional Human-
Driven Vehicles (HDVs) under similar conditions. This section delineates the methodology used to simulate HDVs at signalized intersections using a predefined set of dummy data. The aim is to capture key performance indicators (KPIs) such as average speed, queue length, fuel consumption, and near-miss incidents for HDVs. These KPIs serve as a standard to which we can compare the subsequent performance of various penetration rates of AVs. The following KPIs were identified for measurement during the simulation of HDVs:

1. **Average Speed**: Measured in mph, this KPI will assess the speed efficiency of HDVs navigating through signalized intersections.

2. **Queue Length**: This KPI will be quantified by the average number of vehicles waiting at a red light at the selected intersections. Queue length serves as a direct indicator of traffic congestion.

3. **Near-Miss Incidents**: Defined as instances where collision was narrowly avoided, near-miss incidents will be counted at each signalized intersection. This KPI provides a measure of safety risk associated with HDVs.

The dummy data set that was created incorporates various vehicle types, including sedans, SUVs, and trucks, all manually driven. In order to capture a more realistic representation of urban traffic dynamics, our simulation framework introduces variability in compliance with driving rules among Human-Driven Vehicles (HDVs). While most vehicles in our model adhere to standard traffic laws, including obeying speed limits and safe following distances, we also incorporate a subset of vehicles that exhibit non-compliant driving behaviors. The Simulation of Urban MObility (SUMO) tool provides options to configure different driving behavior profiles. For compliant vehicles, default
settings in SUMO that adhere to legal speed limits and safe following distances are employed. For non-compliant vehicles, specific parameters are adjusted to allow for higher speeds and shorter car-following headways, thereby introducing a level of risk and unpredictability commonly observed in real-world traffic conditions. Introducing both compliant and non-compliant vehicles into our simulation serves multiple purposes. The mixed behavior model mirrors the diversity of driving behaviors typically observed in real-life traffic, enhancing the simulation’s external validity. This setup allows for a nuanced evaluation of how varying penetration rates of AVs might interact with both rule-abiding and rule-breaking HDVs. By including non-compliant vehicles, we can assess how the introduction of AVs impacts overall traffic safety in scenarios where not all HDVs are obeying the law. The signalized intersections selected for the simulation are representative of typical urban intersections with varying traffic densities.

To explore the impact of differing AV penetration rates, we developed the following scenarios:

**Low Penetration - 10%**: This scenario explores a system where 10% of the vehicles are autonomous, serving as an introductory level of AV integration.

**Moderate Penetration - 25%**: Here, one-quarter of the total vehicles are autonomous, reflecting a moderate level of AV adoption.

**High Penetration - 50%**: In this case, half of all vehicles are autonomous, representing a balanced mix of human-driven and autonomous vehicles.

**Very High Penetration - 75%**: This scenario examines the impacts of a system where 75% of the vehicles are autonomous, simulating a near-future situation with widespread AV adoption.
Full Penetration - 100%: The final scenario investigates a fully autonomous traffic system, providing insights into the ultimate impact of a complete shift from human-driven to autonomous vehicles.

Each of these scenarios was subject to the same traffic conditions and signal timings to ensure comparability.

4.4 Data Collection & Analysis

In addition to utilizing a dummy dataset for simulating Human-Driven Vehicles (HDVs) in SUMO, we have further enriched our research by incorporating real-world traffic data. This empirical dataset was collected from multiple signalized intersections in downtown Huntsville to serve as a validation set for our simulations. This real-world dataset serves not only as a critical point of comparison but also as a validation mechanism for the dummy data-based simulation results.

We focused our data collection on two primary parameters: **Number of Vehicles** (A count of vehicles traversing each selected intersection during the study period) and **Traffic Signal Timing** (The cycle durations of green, yellow, and red lights at each intersection). Data was collected over a short but representative timeframe of 15 minutes for each intersection.

Given the limited duration of data collection, the raw data for the number of vehicles and signal timing were extrapolated to represent traffic flow rates and signal cycles. These converted parameters were then incorporated into the SUMO simulation environment to model realistic traffic conditions in downtown Huntsville. Once converted, the empirical data was integrated into our existing SUMO simulation setup. Through the analytical leverage of the simulation, we derived output metrics such as
average speed, queue length, and near-miss incidents. This generated a rich spectrum of data, providing a comprehensive basis for in-depth analysis and interpretation. This allowed us to compare key performance indicators generated using real-world data against those obtained through simulations based on the dummy dataset. The primary objective of incorporating real-world data was to assess the fidelity and reliability of our initial simulations, which utilized dummy data. By juxtaposing results derived from both datasets, we aimed to determine the extent to which our dummy data-based simulations can reliably represent actual traffic conditions. The use of real-world data for simulation validation adds a layer of empirical rigor to our study. The comparative analysis enables us to calibrate our dummy dataset and refine the assumptions that underlie our broader investigation into the impact of varying Autonomous Vehicle penetration rates on urban traffic dynamics.

### 3.4.1 Calculation of Number of Simulation

To gauge the preliminary behavior of the proposed traffic model, we executed an initial set of 100 simulation runs. The calculated sample standard deviations for the key performance indicators were as follows:

**Sample Average Speed:** $\sigma_{\text{speed}} = 6.60$ mph

**Sample Queue Length:** $\sigma_{\text{queue length}} = 4.09$ vehicles
3.4.1.1 Confidence Level and Desired Margin of Error

We have chosen a 95% confidence level for the study, corresponding to a Z-score of 1.96. Additionally, we aim for the margin of error to be restricted to 1 for both average speed and queue length.

3.4.1.2 Sample Size Calculation

The sample size $n$ required for a desired margin of error $ME$ at a given confidence level can be calculated using the following formula:

$$n = \left( \frac{Z \cdot \sigma}{ME} \right)^2.$$

For Average Speed

Using $\sigma_{\text{speed}} = 6.60042$ and $ME=1,$

we find $n_{\text{speed}} = [(1.96 \times 6.600421)/1]^2 \approx 167.36.$

Rounding up, we require approximately 168 simulation runs for average speed.

For Queue Length

Using $\sigma_{\text{queue length}} = 4.0867$ and $ME=1,$

we find $n_{\text{queue length}} = [(1.96 \times 4.08671)/1]^2 \approx 64.16.$

Rounding up, we require approximately 65 simulation runs for queue length.

Given that the higher sample size will provide a more robust model, we opt for 168 simulation runs to satisfy the conditions for both average speed and queue length at the 95% confidence level with a margin of error of 1.

As we have decided to run 168 simulations for the scenario involving only HDVs and we want to compare this with scenarios involving different rates of autonomous vehicle (AV) penetration, it would be statistically sound to keep the number of runs consistent across all scenarios. This ensures that the conditions for comparison are as
uniform as possible, enhancing the validity of any observed differences or trends. For comparability, we planned to run 168 simulations for each of the AV penetration scenarios (10%, 25%, 50%, 75%, and 100%). This ensures that the manual and AV scenarios are directly comparable in terms of statistical power and confidence levels.

With 168 simulations for each of 5 scenarios, we conducted a total of 168×5 = 840 simulations for AVs.

Prior to executing the two-tailed t-test, a pivotal preliminary analysis was undertaken to ascertain the normality of the data distribution. Ensuring that the data adheres to a normal distribution is quintessential for the validity and reliability of the subsequent t-test. This foundational step underscores the rigor and meticulousness of our analytical process, laying a robust groundwork for the ensuing statistical evaluations and interpretations within this dissertation. Thus, the analytical findings presented herein are substantiated through a systematic validation of the underlying data’s distributional properties.

3.4.2. Analysis of Different Penetration Rate of AV

Figure 4.2 and 4.3 represents the variations in average speed and the frequency of near-miss incidents across diverse penetration rates of Autonomous Vehicles (AVs). While these illustrations provide preliminary insights into the trends and patterns associated with varying AV penetration rates, they do not inherently convey the statistical significance of the observed changes. To ascertain the substantiability of these alterations and to provide a more rigorous analytical evaluation, a two-tailed t-test was employed. The comprehensive findings and discussions derived from this statistical examination will be discussed in subsequent segments of this dissertation, ensuring a robust
interpretation of the data and a thorough understanding of the implications associated with AV penetration in traffic systems.

Figure 4.2 Change in Average Speed for Different Penetration Rate of AV.
3.4.2.1 Low Penetration of AV - 10%

Table 4.1 provides descriptive statistics of average speed measurements for a low penetration rate (10%) of AV. A comparison of the means among the 1250 vehicles in the paired t-test resulted in a p-value of 0.846 (Table 1). The p-value is a measure of the evidence against a null hypothesis. In the context of a t-test, the null hypothesis typically posits that there is no difference between the two groups being compared.

Table 4.1 T-Test Results for Average Speed for 10% AV Penetration.

<table>
<thead>
<tr>
<th></th>
<th>Speed (HDV)</th>
<th>Speed (HDV+10%AV)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>40.59</td>
<td>40.49</td>
</tr>
<tr>
<td>Variance</td>
<td>35.10</td>
<td>34.53</td>
</tr>
<tr>
<td>Observations</td>
<td>1250</td>
<td>1250</td>
</tr>
<tr>
<td>Hypothesized Mean Difference</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>df</td>
<td>2498</td>
<td></td>
</tr>
<tr>
<td>t Stat</td>
<td>0.194</td>
<td></td>
</tr>
<tr>
<td>P(T&lt;=t) one-tail</td>
<td>0.42</td>
<td></td>
</tr>
<tr>
<td>t Critical one-tail</td>
<td>1.65</td>
<td></td>
</tr>
<tr>
<td>P(T&lt;=t) two-tail</td>
<td><strong>0.85</strong></td>
<td></td>
</tr>
<tr>
<td>t Critical two-tail</td>
<td>1.96</td>
<td></td>
</tr>
</tbody>
</table>
From table 3.1, the t-test results for comparing average speeds between solely human-driven vehicles (HDV) and a scenario with 10% autonomous vehicles (HDV+10%AV) produced a t-statistic of 0.194. Given its proximity to 0 and when contrasted against critical t-values for both one-tailed (1.65) and two-tailed (1.96) tests, it suggests that there's no statistically significant difference in speeds between the two scenarios. This conclusion is reinforced by the high p-values of 0.42 (one-tailed) and 0.85 (two-tailed), both well above the conventional 0.05 threshold, confirming that the inclusion of 10% autonomous vehicles doesn't notably impact average speeds.

In the comparative analysis between HDV and HDV+10% AV, it was observed that the introduction of Autonomous Vehicles did not yield a statistically significant difference in queue length.

Statistical analysis of near-miss incidents using a two-tailed t-test yielded a p-value of 0.936. This result suggests that the observed differences in near-miss incidents between the two groups are not statistically significant.

<table>
<thead>
<tr>
<th>Near Miss Incident</th>
</tr>
</thead>
<tbody>
<tr>
<td>(HDV)</td>
</tr>
<tr>
<td>Mean</td>
</tr>
<tr>
<td>Variance</td>
</tr>
<tr>
<td>Observations</td>
</tr>
<tr>
<td>Hypothesized Mean Difference</td>
</tr>
<tr>
<td>df</td>
</tr>
<tr>
<td>t Stat</td>
</tr>
<tr>
<td>P(T&lt;=t) one-tail</td>
</tr>
<tr>
<td>t Critical one-tail</td>
</tr>
<tr>
<td>P(T&lt;=t) two-tail</td>
</tr>
<tr>
<td>t Critical two-tail</td>
</tr>
</tbody>
</table>

Table 4.2 T-Test Results for Near-Miss Incident for 10% AV Penetration.
3.4.2.2 Moderate Penetration - 25%

We conducted a rigorous statistical analysis to assess the impact of 25% autonomous vehicle (AV) penetration on three key performance indicators: average speed, queue length, and near-miss incidents. To this end, a two-tailed t-test was performed for each metric, comparing the scenarios of 100% human-driven vehicles (HDV) and a mixed environment of HDVs and 25% AVs (HDV+25%AV). Our analyses revealed that the p-values for average speed, queue length, and near-miss incidents were all above the conventional alpha level of 0.05. Consequently, the introduction of 25% AV penetration did not produce statistically significant changes in any of the examined metrics when compared to a pure HDV environment. The results are shown in Table 3.3 and 3.4.

<table>
<thead>
<tr>
<th>Table 4.3 T-Test Results for Average Speed for 25% AV Penetration.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td>Speed (HDV)</td>
</tr>
<tr>
<td>Mean</td>
</tr>
<tr>
<td>Variance</td>
</tr>
<tr>
<td>Observations</td>
</tr>
<tr>
<td>Hypothesized Mean Difference</td>
</tr>
<tr>
<td>df</td>
</tr>
<tr>
<td>t Stat</td>
</tr>
<tr>
<td>P(T&lt;=t) one-tail</td>
</tr>
<tr>
<td>t Critical one-tail</td>
</tr>
<tr>
<td>P(T&lt;=t) two-tail</td>
</tr>
<tr>
<td>t Critical two-tail</td>
</tr>
</tbody>
</table>
Table 4.4 T-Test Results for Near-Miss Incident for 25% AV Penetration.

<table>
<thead>
<tr>
<th></th>
<th>Near Miss Incident (HDV)</th>
<th>Near Miss Incident (HDV+25%AV)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>4.32</td>
<td>4.067</td>
</tr>
<tr>
<td>Variance</td>
<td>2.88</td>
<td>2.753</td>
</tr>
<tr>
<td>Observations</td>
<td>1250</td>
<td>1250</td>
</tr>
<tr>
<td>Hypothesized Mean Difference</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>df</td>
<td>2498</td>
<td></td>
</tr>
<tr>
<td>t Stat</td>
<td>0.397</td>
<td></td>
</tr>
<tr>
<td>P(T&lt;=t) one-tail</td>
<td>0.346</td>
<td></td>
</tr>
<tr>
<td>t Critical one-tail</td>
<td>1.65</td>
<td></td>
</tr>
<tr>
<td>P(T&lt;=t) two-tail</td>
<td>0.692</td>
<td></td>
</tr>
<tr>
<td>t Critical two-tail</td>
<td>1.96</td>
<td></td>
</tr>
</tbody>
</table>

3.4.2.3 High Penetration - 50%

In an effort to quantify the effects of a 50% penetration of autonomous vehicles (AVs) on traffic dynamics, a two-tailed t-test was executed for each of these parameters to compare two scenarios: one featuring 100% human-driven vehicles (HDVs) and another with a mixed fleet of HDVs and 50% AVs (HDV+50%AV). Our results indicated that the inclusion of 50% AVs in the traffic mix led to statistically significant changes in both average speed and near-miss incidents, as evidenced by p-values below the conventional alpha level of 0.05. However, the change in queue length was found to be statistically insignificant, with a p-value above 0.05. The following table (3.5 and 3.6) provides a detailed summary of the statistical tests conducted for average speed and near-miss incidents.
Table 4.5 T-Test Results for Average Speed for 50% AV Penetration.

<table>
<thead>
<tr>
<th></th>
<th>Speed (HDV)</th>
<th>Speed (HDV+50%AV)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>40.59</td>
<td>37.60</td>
</tr>
<tr>
<td>Variance</td>
<td>35.10</td>
<td>21.29</td>
</tr>
<tr>
<td>Observations</td>
<td>1250</td>
<td>1250</td>
</tr>
<tr>
<td>Hypothesized Mean Difference</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>df</td>
<td>2498</td>
<td></td>
</tr>
<tr>
<td>t Stat</td>
<td>7.83</td>
<td></td>
</tr>
<tr>
<td>P(T&lt;=t) one-tail</td>
<td>5.66502E-14</td>
<td></td>
</tr>
<tr>
<td>t Critical one-tail</td>
<td>1.65</td>
<td></td>
</tr>
<tr>
<td>P(T&lt;=t) two-tail</td>
<td>1.133E-13</td>
<td></td>
</tr>
<tr>
<td>t Critical two-tail</td>
<td>1.97</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.6 T-Test Results for Near-Miss Incident for 50% AV Penetration.

<table>
<thead>
<tr>
<th></th>
<th>Near Miss Incident (HDV)</th>
<th>Near Miss Incident (HDV+50%AV)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>4.32</td>
<td>1.588</td>
</tr>
<tr>
<td>Variance</td>
<td>2.88</td>
<td>1.26</td>
</tr>
<tr>
<td>Observations</td>
<td>1250</td>
<td>1250</td>
</tr>
<tr>
<td>Hypothesized Mean Difference</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>df</td>
<td>2498</td>
<td></td>
</tr>
<tr>
<td>t Stat</td>
<td>14.47</td>
<td></td>
</tr>
<tr>
<td>P(T&lt;=t) one-tail</td>
<td>2.28E-37</td>
<td></td>
</tr>
<tr>
<td>t Critical one-tail</td>
<td>1.65</td>
<td></td>
</tr>
<tr>
<td>P(T&lt;=t) two-tail</td>
<td>4.57E-37</td>
<td></td>
</tr>
<tr>
<td>t Critical two-tail</td>
<td>1.97</td>
<td></td>
</tr>
</tbody>
</table>

3.4.2.4 Very High and Full Penetration (75% & 100% of AV)

In the continuation of our statistical analysis, we extended the study to examine the impact of higher levels of autonomous vehicle (AV) penetration, specifically 75% and 100%, on three key traffic performance indicators: average speed, queue length, and near-miss incidents. A two-tailed t-test was employed for each of these metrics to scrutinize the differences between scenarios featuring only human-driven vehicles (HDVs) and mixed fleets of HDVs combined with 75% and 100% AVs. Our findings demonstrated statistically significant changes in both average speed and near-miss incidents.
incidents, as indicated by p-values falling below the commonly accepted alpha level of 0.05. Notably, these results were consistent with those observed at the 50% AV penetration rate. However, the variation in queue length remained statistically insignificant across all levels of AV penetration, reaffirming the results obtained at the 50% penetration rate.

Table 3.5 and 3.6 provides a detailed summary of the statistical tests conducted for average speed.

**Table 4.7** T-Test Results for Average Speed for 75% AV Penetration.

<table>
<thead>
<tr>
<th></th>
<th>Speed (HDV)</th>
<th>Speed (HDV+75%AV)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>40.59</td>
<td>36.79</td>
</tr>
<tr>
<td>Variance</td>
<td>35.10</td>
<td>24.29</td>
</tr>
<tr>
<td>Observations</td>
<td>1250</td>
<td>1250</td>
</tr>
<tr>
<td>Hypothesized Mean Difference</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>df</td>
<td>2498</td>
<td></td>
</tr>
<tr>
<td>t Stat</td>
<td>8.81</td>
<td></td>
</tr>
<tr>
<td>P(T&lt;=t) one-tail</td>
<td>4.664E-17</td>
<td></td>
</tr>
<tr>
<td>t Critical one-tail</td>
<td>1.65</td>
<td></td>
</tr>
<tr>
<td>P(T&lt;=t) two-tail</td>
<td>9.33E-17</td>
<td>1.97</td>
</tr>
<tr>
<td>t Critical two-tail</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Table 4.8** T-Test Results for Average Speed for 100% AV Penetration.

<table>
<thead>
<tr>
<th></th>
<th>Speed (HDV)</th>
<th>Speed (HDV+100%AV)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>40.59</td>
<td>35.21</td>
</tr>
<tr>
<td>Variance</td>
<td>35.10</td>
<td>16.85</td>
</tr>
<tr>
<td>Observations</td>
<td>1250</td>
<td>1250</td>
</tr>
<tr>
<td>Hypothesized Mean Difference</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>df</td>
<td>2498</td>
<td></td>
</tr>
<tr>
<td>t Stat</td>
<td>9.61</td>
<td></td>
</tr>
<tr>
<td>P(T&lt;=t) one-tail</td>
<td>7.909E-20</td>
<td></td>
</tr>
<tr>
<td>t Critical one-tail</td>
<td>1.65</td>
<td></td>
</tr>
<tr>
<td>P(T&lt;=t) two-tail</td>
<td>1.58E-19</td>
<td>1.97</td>
</tr>
<tr>
<td>t Critical two-tail</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 3.5 summarizes the significance of the change of speed, near-miss incident and queue length of different penetration rate of AV compared to no AV on the roadway that were evaluated in the simulation study.

Table 4.9 T-Test Results for Various AV Penetration (Dummy Data).

<table>
<thead>
<tr>
<th>KPI</th>
<th>AV Penetration Rate</th>
<th>Significant Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10%</td>
<td></td>
<td>No</td>
</tr>
<tr>
<td>25%</td>
<td></td>
<td>No</td>
</tr>
<tr>
<td>50%</td>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td>75%</td>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td>100%</td>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td>Near-Miss Incident</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10%</td>
<td></td>
<td>No</td>
</tr>
<tr>
<td>25%</td>
<td></td>
<td>No</td>
</tr>
<tr>
<td>50%</td>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td>75%</td>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td>100%</td>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td>Queue Length</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10%</td>
<td></td>
<td>No</td>
</tr>
<tr>
<td>25%</td>
<td></td>
<td>No</td>
</tr>
<tr>
<td>50%</td>
<td></td>
<td>No</td>
</tr>
<tr>
<td>75%</td>
<td></td>
<td>No</td>
</tr>
<tr>
<td>100%</td>
<td></td>
<td>No</td>
</tr>
</tbody>
</table>

3.4.3 Utilizing Collected Data from Real-World

In an endeavor to validate the robustness of our preliminary findings, we extended our analysis to include data collected from various intersections. As mentioned earlier, due to the short data collection period, the initial data on vehicle count and signal timing were extrapolated to estimate traffic flow rates and signal cycles. The transformed parameters were subsequently included into the SUMO simulation environment to simulate authentic traffic conditions in downtown Huntsville. After conversion, the empirical data was incorporated into our pre-existing SUMO simulation framework. For this expanded analysis, we maintained our focus on the three principal traffic
performance indicators: average speed, queue length, and near-miss incidents. This additional dataset was helpful in comparing the consistency between the simulated outcomes from field data and the replicated results from 'dummy' data.

A two-tailed t-test was employed to evaluate the statistical significance of the observed changes in these indicators for different autonomous vehicle (AV) penetration into the mixed traffic flow comprising human-driven vehicles (HDVs) and AVs. Our results indicate that the introduction of AVs at these higher penetration rates (50%, 75% and 100%) had a statistically significant impact on both average speed and the occurrence of near-miss incidents. However, the same could not be said for queue length, for which the change was not statistically significant. Intriguingly, these results corroborate findings obtained using preliminary dummy data, thereby strengthening the validity of our initial observations.

Our findings has also shown that for 25% AV penetration, while the variations in average speed and queue length were not statistically significant (p-values exceeding the alpha level of 0.05), the changes in near-miss incidents reached statistical significance, as evidenced by p-values falling below the predetermined alpha level. These results provide a nuanced understanding of the impact of a moderate level of AV penetration on the selected performance indicators and validate the utility of our simulation model for future exploratory studies.

In summary, while higher penetration rates of AVs were observed to significantly reduce the average speed and reduce the frequency of near-miss incidents, they did not result in a statistically significant reduction in queue length. This consistency in findings, even with varying levels of AV penetration and across different data sets, offers
compelling evidence for the transformative potential of autonomous vehicles in enhancing road safety and traffic efficiency. Nonetheless, it also highlights that the integration of AVs might not be a silver bullet for all traffic-related issues, such as queue length, warranting further investigation.

The introduction of Autonomous Vehicles (AVs) into a traffic system may not necessarily improve queue lengths for several reasons. If the road is already near or at its capacity, introducing AVs might not make much of a difference in queue lengths because the bottleneck is not necessarily the behavior of the drivers but the physical limitations of the road. In a scenario where AVs and Human-Driven Vehicles (HDVs) share the road, human behavior, such as hesitancy or aggressive driving, might neutralize the efficiency gains that AVs could bring in an ideal, AV-only environment. At lower penetration rates, the number of AVs may not be sufficient to bring about noticeable changes in system-wide performance metrics like queue length. If traffic lights or other traffic control measures are not optimized for the mixed flow of AVs and HDVs, this can maintain or even worsen queue lengths. If AVs are programmed to prioritize safety and thus exhibit more conservative behavior in terms of speed changes, this could reduce the system's overall throughput, affecting queue lengths.

4.5 Conclusion

In this comprehensive study, we sought to understand the implications of introducing Autonomous Vehicles (AVs) into existing traffic systems composed predominantly of Human-Driven Vehicles (HDVs). Our research aimed to quantify changes in key performance indicators, namely average speed, queue length, and the
frequency of near-miss incidents under varying penetration rates of AVs (10%, 25%, 50%, 75%, and 100%).

Our results present a nuanced picture of the impact of AV integration. Utilizing two-tailed t-tests, we observed that the introduction of AVs led to statistically significant improvements in average speed and a reduction in near-miss incidents at higher penetration rates (50%, 75%, and 100%). These findings were consistent across both simulated and real-world intersection data, strengthening their validity. These improvements could be attributed to the precise and predictive nature of AV algorithms, which minimize erratic driving behaviors that often lead to inefficiencies and near-miss events in HDV-dominated systems.

However, the data revealed no significant reduction in queue lengths at any penetration level. If the road is already near or at its capacity, introducing AVs might not make much of a difference in queue lengths because the bottleneck is not necessarily the behavior of the drivers but the physical limitations of the road. One possibility is that the overall traffic flow and road capacity weren't substantially improved with the introduction of AVs. Additionally, the mixed traffic conditions, consisting of human-driven vehicles and AVs, might not have optimized the traffic flow effectively, preventing a notable decrease in queue lengths. This highlights the limitations of AVs in addressing certain forms of road congestion, especially in scenarios already at or near saturation levels or when the traffic control systems are not optimized for a heterogeneous mix of AVs and HDVs. The limitations could also be due to mixed traffic behaviors, route choices, or policy and regulatory constraints that prevent AVs from fully optimizing queue throughput.
It was particularly interesting to note a significant reduction in near-miss incidents even at the lower AV penetration rate of 25%, solely in the simulated real-world intersection scenarios. This suggests that even a modest introduction of AVs could markedly improve road safety under specific conditions. However, further research is needed to isolate the underlying factors contributing to this significant change.

Our study underscores the complexity of integrating AVs into existing transport ecosystems and the need for a multi-faceted approach to fully realize their potential benefits. While our results are promising, especially concerning improvements in average speed and safety, they also point to areas where AV integration alone may be insufficient.

As autonomous driving technologies continue to advance and gain regulatory approval, studies like ours are pivotal in informing transport policy, urban planning, and public perception. Future research should focus on optimizing traffic control systems for mixed vehicular traffic and investigating the scalability of our findings to broader, more diverse settings.

Thus, while the introduction of Autonomous Vehicles has shown potential in enhancing certain dimensions of road traffic performance, it is not a panacea for all traffic-related issues. The journey toward full AV integration will be an iterative process requiring collaboration among technologists, policymakers, and urban planners.

Ultimately, our findings will contribute to more informed decision-making and assist in the development of comprehensive strategies to harness the potential of autonomous vehicles in creating sustainable, efficient, and equitable urban mobility solutions.
Chapter 5. Conclusions

This research explored the complex terrain of integrating autonomous vehicles (AVs) in urban settings through a series of related investigations. The overall goal was to identify how AVs interact with current traffic actors, and anticipate future changes to urban mobility.

Initially, we investigated the complex relationships between AVs and bicycles in urban unsignalized intersections. Our co-simulation results highlighted a crucial trade-off between efficiency and safety in AV behavioral designs, using the integrated urban area map provided by CARLA. The compromise is the corresponding increase in wasted time for automobiles, with a trend clearly showing a decrease in AV-cyclist collisions as waiting times and distance criteria climb. Future research might make greater use of complex 3D maps, incorporate real-time data, and incorporate a variety of environmental factors to more closely mimic the difficulties and uncertainties encountered in the actual world. This would increase the realism and application of these findings.

After moving on from the micro-dynamics of AV-cyclist interactions, we expanded our story to include AV interactions with the larger ecosystem of road users. With the use of the cooperative environments of CARLA and SUMO co-simulation, the study brought game theory concepts—particularly the Chicken Game Theory—to the domain of AVs. The use of surrogate safety measures highlighted the potential of AVs in enhancing road safety by offering insights into possible crash scenarios. Nevertheless, understanding the intricacies of patterns driven by humans is crucial. The smooth operation of autonomous vehicles depends on the capacity to accept and mimic the diversity of driving behaviors exhibited by human drivers. This study phase highlights
the unwavering importance of simulation tools and emphasizes the necessity to enhance and improve these models in order to capture the complex reality of traffic behavior.

Lastly, the research trajectory propelled into the domain of AV integration ramifications under varied penetration rates. Comprehensive data analysis manifested palpable improvements in average speeds and marked reductions in near-miss incidents, particularly at elevated AV penetration rates. However, the persistence of queue lengths across penetration levels underscores the multi-dimensional challenges of urban congestion. Intriguingly, even a modest AV integration was indicative of heightened road safety under particular conditions, necessitating further investigation into the intrinsic contributing factors.

This research provides a comprehensive exploration into the intricate dynamics of autonomous vehicles (AVs) interacting within urban traffic environments, with a special focus on conflicts at unsignalized intersections and the nuanced behaviors in mixed traffic scenarios. Utilizing innovative co-simulation environments like CARLA and SUMO, the study meticulously evaluates various surrogate safety measures and delves into the impactful realms of game theory to augment our understanding of AV interactions. It unveils insights into the balance between safety and efficiency in AV operations, offering an analysis of collisions, waiting times, and traffic flow implications under varied AV penetration rates. Through this multi-faceted approach, the research offers valuable contributions towards enhancing AV safety protocols, algorithmic functionalities, and overall traffic management strategies, thereby facilitating the informed integration of autonomous vehicles into our evolving urban mobility landscapes.
In summation, this integrative research endeavor underscores the profound potential and inherent complexities associated with AV integration in urban settings. While AVs undeniably offer transformative possibilities in redefining mobility, their holistic integration demands a confluence of cutting-edge technology, adaptive policy frameworks, and proactive urban planning. As we stand at the cusp of an automotive revolution, our hope is that this research will be helpful in illuminating the path towards sustainable, efficient, and inclusive urban mobility solutions.
**References**


49. Luetge C. 2017. The German ethics code for automated and connected driving. *Philosophy & Technology* 30:547-58
to address safety validation of automated driving. *Proc. 2018 21st International
Conference on Intelligent Transportation Systems (ITSC), 2018:491-6: IEEE*

*Transportation Research Record* 1840:104-15

52. Laureshyn A, Svensson Å, Hydén C. 2010. Evaluation of traffic safety, based on
micro-level behavioural data: Theoretical framework and first implementation. *Accident
Analysis & Prevention* 42:1637-46

53. Archer J. 2005. *Indicators for traffic safety assessment and prediction and their
application in micro-simulation modelling: A study of urban and suburban intersections.*
KTH

traffic safety analysis. *Journal of transport geography* 38:13-21

intersections with cycle tracks safer? A case–control study based on automated surrogate
safety analysis using video data. *Accident Analysis & Prevention* 86:161-72

occurrences. *Accident Analysis & Prevention* 42:626-36

57. Talebpour A, Mahmassani HS. 2016. Influence of connected and autonomous
vehicles on traffic flow stability and throughput. *Transportation research part C:
emerging technologies* 71:143-63

58. Löcken A, Golling C, Riener A. How should automated vehicles interact with
59. Hayward JC. 1972. Near miss determination through use of a scale of danger


71. Schoettle B, Sivak M. 2014. A survey of public opinion about autonomous and self-driving vehicles in the US, the UK, and Australia, University of Michigan, Ann Arbor, Transportation Research Institute


from __future__ import print_function

import argparse
import collections
import datetime
import glob
import logging
import math
import os
import random
import re
import sys
import weakref
import network
import time
import csv

try:
    import pygame
    from pygame.locals import KMOD_CTRL
    from pygame.locals import K_ESCAPE
    from pygame.locals import K_q
except ImportError:
    raise RuntimeError('cannot import pygame, make sure pygame package
is installed')

try:
    import numpy as np
except ImportError:
    raise RuntimeError('cannot import numpy, make sure numpy package is installed')

#======================================================================
#-- Find CARLA module --------------------------------------------------
#======================================================================

try:
    sys.path.append(glob.glob('.. /*%s.egg' % (sys.version_info.major,
    sys.version_info.minor,
    'win-amd64' if os.name == 'nt' else 'linux-x86_64'))[0])
except IndexError:
    pass
try:
    sys.path.append(os.path.dirname(os.path.dirname(os.path.abspath(__file__))) + '/carla')
except IndexError:
    pass

import carla
from carla import ColorConverter as cc

from agents.navigation.behavior_agent import BehaviorAgent  # pylint: disable=import-error
from agents.navigation.roaming_agent import RoamingAgent  # pylint: disable=import-error
from agents.navigation.basic_agent import BasicAgent  # pylint: disable=import-error

# ----------------------------------------
# Global functions
# ----------------------------------------

file = None

def find_weather_presets():
    rgx = re.compile('.+?(?:(?<=[a-z])(?=[A-Z])|(?<=[A-Z])(?=[A-Z][a-zA-Z])|$)')
    def name(x):
        return ''.join(m.group(0) for m in rgx.finditer(x))
    presets = [x for x in dir(carla.WeatherParameters) if re.match('[A-Z]+', x)]
    return [(getattr(carla.WeatherParameters, x), name(x)) for x in presets]

def get_actor_display_name(actor, truncate=250):
    name = actor.type_id.replace('_', '.').title().split('.')[:-1]
```python
return (name[:truncate - 1] + u'…') if len(name) > truncate
else name

# """ New Global functions for Data Collection """

# Prepare Filename
filename = 'Locations_' + time.strftime("%Y%m%d-%H%M%S") + '.csv'
file = open(filename, 'a+)

# CONVERT TIME

def convert_time(seconds):
    seconds = seconds % (24*3600)
    hour = seconds // 3600
    seconds %= 3600
    minutes = seconds // 60
    seconds %= 60
    mili = (seconds * 1000)%1000
    return '%d:%02d:%02d:%04d' % (hour, minutes, seconds, mili)

# LOG DATA

def get_locations(snapshot):
    # filter exact type of ego vehicle?
    # otherwise alter this function to specifically get ego vehicle
    for vehicles in actor_list.filter('vehicle. '):
        extract_locations(snapshot, vehicles)

# """ World """

class World(object):
    """ Class representing the surrounding environment """
    def __init__(self, carla_world, hud, args):
        """ Constructor method """
        self.world = carla_world
```
try:
    self.map = self.world.get_map()
except RuntimeError as error:
    print('RuntimeError: {}\n'.format(error))
    print('The server could not send the OpenDRIVE (.xodr) file:')
    print('Make sure it exists, has the same name of your town, and is correct.')
sys.exit(1)
self.hud = hud
self.player = None
self.collision_sensor = None
self.lane_invasion_sensor = None
self.gnss_sensor = None
self.camera_manager = None
self.weather_presets = find_weather_presets()
self.weather_index = 0
self._actor_filter = args.filter
self._gamma = args.gamma
self.restart(args)
self.world.on_tick(hud.on_world_tick)
self.recording_enabled = False
self.recording_start = 0

# CONVERT TIME
def convert_time(self, seconds):
    seconds = seconds % (24 * 3600)
    hour = seconds // 3600
    seconds %= 3600
    minutes = seconds // 60
    seconds %= 60
    mili = (seconds * 1000) % 1000
    return '%d:%02d:%02d:%04d' % (hour, minutes, seconds, mili)

# EXTRACT DATA
def extract_locations(self, snap, vehicle):

    # Gather all data
    vehicle_snap = snap.find(vehicle.id)
    transform = vehicle_snap.get_transform()
    frame = str(snap.frame)
    time = self.convert_time(snap.timestamp.elapsed_seconds)
    actor_id = str(vehicle.id)  # blueprint library id
    type_id = str(vehicle.type_id)  # vehicle type
    x = str('%.10f' % (transform.location.x))
    y = str('%.10f' % (transform.location.y))
    z = str('%.10f' % (transform.location.z))
    v = vehicle_snap.get_velocity()
    a = vehicle_snap.get_acceleration()
    kmh = str('%.15f' % (3.6 * math.sqrt(v.x**2 + v.y**2 + v.z**2))))
ms2 = str(("% 15.2f" % (3.6*math.sqrt(a.x**2 + a.y**2 + a.z**2))))
gear = str(vehicle.get_control().gear)

# Write data
output = (frame + ',' + time + ',' + actor_id + ',' + type_id + ',' + x + ',' + y + ',' + z + ',' + kmh + ',' + ms2 + ',' + gear)
print(output)
#file = open(filename, 'a+')
file.write(output + "\n")

def get_locations(self, snapshot):
    # filter exact type of ego vehicle?
    # otherwise alter this function to specifically get ego vehicle
    self.extract_locations(snapshot, self.player)

def restart(self, args):
    """Restart the world""
    # Keep same camera config if the camera manager exists.
    cam_index = self.camera_manager.index if self.camera_manager is not None else 0
    cam_pos_id = self.camera_manager.transform_index if self.camera_manager is not None else 0
    # Set the seed if requested by user
    if args.seed is not None:
        random.seed(args.seed)
    # Get a random blueprint.
    blueprint = self.world.get_blueprint_library().filter('model3')[0]
    blueprint.set_attribute('role_name', 'hero')
    if blueprint.has_attribute('color'):
        color = random.choice(blueprint.get_attribute('color').recommended_values)
        blueprint.set_attribute('color', color)
    # Spawn the player.
    print("Spawning the player")
    if self.player is not None:
        spawn_point = self.player.get_transform()
        spawn_point.location.z += 2.0
        spawn_point.rotation.roll = 0.0
        spawn_point.rotation.pitch = 0.0
        self.destroy()
        self.player = self.world.try_spawn_actor(blueprint, spawn_point)

    while self.player is None:
        if not self.map.get_spawn_points():
            print('There are no spawn points available in your map/town."
        print('Please add some Vehicle Spawn Point to your UE4 scene."
        sys.exit(1)
spawn_points = self.map.get_spawn_points()
spawn_point = carla.Transform(carla.Location(x=17.2, y=-170.6, z=10), carla.Rotation())
#spawn_point = spawn_points[4] if spawn_points else carla.Transform()
self.player = self.world.try_spawn_actor(blueprint, spawn_point)

# Set up the sensors.
    self.collision_sensor = CollisionSensor(self.player, self.hud)
    self.lane_invasion_sensor = LaneInvasionSensor(self.player, self.hud)
    self.gnss_sensor = GnssSensor(self.player)
    self.camera_manager = CameraManager(self.player, self.hud, self._gamma)
    self.camera_manager.transform_index = cam_pos_id
    self.camera_manager.set_sensor(cam_index, notify=False)
    actor_type = get_actor_display_name(self.player)
    self.hud.notification(actor_type)

def next_weather(self, reverse=False):
    """Get next weather setting""
    self._weather_index += -1 if reverse else 1
    self._weather_index %= len(self._weather_presets)
    preset = self._weather_presets[self._weather_index]
    self.hud.notification('Weather: %s' % preset[1])
    self.player.get_world().set_weather(preset[0])

def tick(self, clock):
    """Method for every tick""
    self.hud.tick(self, clock)

def render(self, display):
    """Render world""
    self.camera_manager.render(display)
    self.hud.render(display)

def destroy_sensors(self):
    """Destroy sensors""
    self.camera_manager.sensor.destroy()
    self.camera_manager.sensor = None
    self.camera_manager.index = None

def destroy(self):
    """Destroys all actors""
    actors = [
        self.camera_manager.sensor,
        self.collision_sensor.sensor,
        self.lane_invasion_sensor.sensor,
        self.gnss_sensor.sensor,
        self.player]
    for actor in actors:
        if actor is not None:
            actor.destroy()
class KeyboardControl(object):
    def __init__(self, world):
        world.hud.notification("Press 'H' or '?' for help.",
                            seconds=4.0)

    def parse_events(self):
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                return True
            if event.type == pygame.KEYUP:
                if self._is_quit_shortcut(event.key):
                    return True

    @staticmethod
    def _is_quit_shortcut(key):
        """Shortcut for quitting""
        return (key == K_ESCAPE)
        or (key == K_q and pygame.key.get_mods() & KMOD_CTRL)

class HUD(object):
    """Class for HUD text""
    def __init__(self, width, height):
        """Constructor method""
        self.dim = (width, height)
        font = pygame.font.Font(pygame.font.get_default_font(), 20)
        font_name = 'courier' if os.name == 'nt' else 'mono'
        fonts = [x for x in pygame.font.get_fonts() if font_name in x]
        default_font = 'ubuntumono'
        mono = default_font if default_font in fonts else fonts[0]
        mono = pygame.font.match_font(mono)
self._font_mono = pygame.font.Font(mono, 12 if os.name == 'nt' else 14)
self._notifications = FadingText(font, (width, 40), (0, height - 40))
self.help = HelpText(pygame.font.Font(mono, 24), width, height)
self.server_fps = 0
self.frame = 0
self.simulation_time = 0
self.show_info = True
self.info_text = []
self.server_clock = pygame.time.Clock()

def on_world_tick(self, timestamp):
    """Gets informations from the world at every tick""
    self.server_clock.tick()
    self.server_fps = self.server_clock.get_fps()
    self.frame = timestamp.frame_count
    self.simulation_time = timestamp.elapsed_seconds

def tick(self, world, clock):
    """HUD method for every tick""
    self.notifications.tick(world, clock)
    if not self.show_info:
        return
    transform = world.player.get_transform()
    vel = world.player.get_velocity()
    control = world.player.get_control()
    heading = 'N' if abs(transform.rotation.yaw) < 89.5 else ''
    heading += 'S' if abs(transform.rotation.yaw) > 90.5 else ''
    heading += 'E' if 179.5 > transform.rotation.yaw > 0.5 else ''
    heading += 'W' if -0.5 > transform.rotation.yaw > -179.5 else ''

    colhist = world.collision_sensor.get_collision_history()
    collision = [colhist[x + self.frame - 200] for x in range(0, 200)]
    max_col = max(1.0, max(collision))
    collision = [x / max_col for x in collision]
    vehicles = world.world.get_actors().filter('vehicle.*')

    self.info_text = [
        'Server: %16.0f FPS' % self.server_fps,
        'Client: %16.0f FPS' % clock.get_fps(),
        'Server clock: %16.0f FPS' % self.server_clock.get_fps(),
        'Vehicle: %20s' % get_actor_display_name(world.player, truncate=20),
        'Map: %20s' % world.map.name,
        'Simulation time: %12s' %
        'Speed: %15.0f km/h' % (3.6 * math.sqrt(vel.x**2 + vel.y**2 + vel.z**2)),
    ]
if isinstance(control, carla.VehicleControl):
    self._info_text += [
        ('Throttle:', control.throttle, 0.0, 1.0),
        ('Steer:', control.steer, -1.0, 1.0),
        ('Brake:', control.brake, 0.0, 1.0),
        ('Reverse:', control.reverse),
        ('Hand_brake:', control.hand_brake),
        ('Manual:', control.manual_gear_shift),
        ('Gear: %s' % {-1: 'R', 0: 'N'}).get(control.gear, control.gear)]
elif isinstance(control, carla.WalkerControl):
    self._info_text += [
        ('Speed:', control.speed, 0.0, 5.556),
        ('Jump:', control.jump)]
    self._info_text += [
        '',
        'Collision:',
        collision,
        '',
        'Number of vehicles: % 8d' % len(vehicles)]

if len(vehicles) > 1:
    self._info_text += ['Nearby vehicles:
']

def dist(l):
    return math.sqrt((l.x - transform.location.x)**2 + (l.y - transform.location.y)**2 + (l.z - transform.location.z)**2)
vehicles = [(dist(x.get_location()), x) for x in vehicles if x.id != world.player.id]
for dist, vehicle in sorted(vehicles):
    if dist > 200.0:
        break
    vehicle_type = get_actor_display_name(vehicle, truncate=22)
    self._info_text.append('% 4d %s' % (dist, vehicle_type))

def toggle_info(self):
    """Toggle info on or off""
    self._show_info = not self._show_info

def notification(self, text, seconds=2.0):
    """Notification text""
    self.notifications.set_text(text, seconds=seconds)
def error(self, text):
    """Error text""
    self._notifications.set_text('Error: %s' % text, (255, 0, 0))

def render(self, display):
    """Render for HUD class""
    if self._show_info:
        info_surface = pygame.Surface((220, self.dim[1]))
        info_surface.set_alpha(100)
        display.blit(info_surface, (0, 0))
        v_offset = 4
        bar_h_offset = 100
        bar_width = 106
        for item in self._info_text:
            if v_offset + 18 > self.dim[1]:
                break
            if isinstance(item, list):
                if len(item) > 1:
                    points = [(x + 8, v_offset + 8 + (1 - y) * 30)
                              for x, y in enumerate(item)]
                    pygame.draw.lines(display, (255, 136, 0), False, points, 2)
                    item = None
                    v_offset += 18
            elif isinstance(item, tuple):
                if isinstance(item[1], bool):
                    rect = pygame.Rect((bar_h_offset, v_offset + 8), (6, 6))
                    pygame.draw.rect(display, (255, 255, 255), rect, 0 if item[1] else 1)
                else:
                    rect_border = pygame.Rect((bar_h_offset, v_offset + 8), (bar_width, 6))
                    pygame.draw.rect(display, (255, 255, 255), rect_border, 1)
                    fig = (item[1] - item[2]) / (item[3] - item[2])
                    if item[2] < 0.0:
                        rect = pygame.Rect((bar_h_offset + fig * (bar_width - 6),
                                             v_offset + 8), (6, 6))
                    else:
                        rect = pygame.Rect((bar_h_offset, v_offset + 8), (fig * bar_width, 6))
                    pygame.draw.rect(display, (255, 255, 255), rect)
                    item = item[0]
            else:
                surface = self._font_mono.render(item, True, (255, 255, 255))
                display.blit(surface, (8, v_offset))
        v_offset += 18
self.notifications.render(display)
self.help.render(display)

#======================================================================
#------
# -- FadingText ------------------
#------
#======================================================================

class FadingText(object):
    """ Class for fading text """
    def __init__(self, font, dim, pos):
        """Constructor method""
        self.font = font
        self.dim = dim
        self.pos = pos
        self.seconds_left = 0
        self.surface = pygame.Surface(self.dim)

    def set_text(self, text, color=(255, 255, 255), seconds=2.0):
        """Set fading text""
        text_texture = self.font.render(text, True, color)
        self.surface = pygame.Surface(self.dim)
        self.seconds_left = seconds
        self.surface.fill((0, 0, 0))
        self.surface.blit(text_texture, (10, 11))

    def tick(self, _, clock):
        """Fading text method for every tick""
        delta_seconds = 1e-3 * clock.get_time()
        self.seconds_left = max(0.0, self.seconds_left - delta_seconds)
        self.surface.set_alpha(500.0 * self.seconds_left)

    def render(self, display):
        """Render fading text method""
        display.blit(self.surface, self.pos)

#======================================================================
#------
# -- HelpText ------------------
#------
#======================================================================
class HelpText(object):
    """ Helper class for text render"""
    def __init__(self, font, width, height):
        lines = __doc__.split('\n')
        self.font = font
        self.dim = (680, len(lines) * 22 + 12)
        self.pos = (0.5 * width - 0.5 * self.dim[0], 0.5 * height - 0.5 * self.dim[1])
        self.seconds_left = 0
        self.surface = pygame.Surface(self.dim)
        self.surface.fill((0, 0, 0, 0))
        for i, line in enumerate(lines):
            text_texture = self.font.render(line, True, (255, 255, 255))
            self.surface.blit(text_texture, (22, i * 22))
        self._render = False
        self.surface.set_alpha(220)
    def toggle(self):
        """Toggle on or off the render help"""
        self._render = not self._render
    def render(self, display):
        """Render help text method"""
        if self._render:
            display.blit(self.surface, self.pos)

# ===============
# -- CollisionSensor
# ===============

class CollisionSensor(object):
    """ Class for collision sensors"""
    def __init__(self, parent_actor, hud):
        self.sensor = None
        self.history = []
        self._parent = parent_actor
        self.hud = hud
        world = self._parent.get_world()
        blueprint = world.get_blueprint_library().find('sensor.other.collision')
        self.sensor = world.spawn_actor(blueprint, carla.Transform(), attach_to=self._parent)
# We need to pass the lambda a weak reference to self to avoid circular reference.
weak_self = weakref.ref(self)
self.sensor.listen(lambda event: 
    CollisionSensor._on_collision(weak_self, event))

def get_collision_history(self):
    """Gets the history of collisions""
    history = collections.defaultdict(int)
    for frame, intensity in self.history:
        history[frame] += intensity
    return history

@staticmethod
def _on_collision(weak_self, event):
    """On collision method""
    self = weak_self()
    if not self:
        return
    actor_type = get_actor_display_name(event.other_actor)
    self.hud.notification('Collision with %r %r' % actor_type)
    impulse = event.normal_impulse
    intensity = math.sqrt(impulse.x ** 2 + impulse.y ** 2 +
        impulse.z ** 2)
    self.history.append((event.frame, intensity))
    if len(self.history) > 4000:
        self.history.pop(0)

# ==-----------------------------==================================================
#
# -- LaneInvasionSensor ----------------------------------------------------------
#
# ==---------------------------------------------------------------------------
#
class LaneInvasionSensor(object):
    """Class for lane invasion sensors""

    def __init__(self, parent_actor, hud):
        """Constructor method""
        self.sensor = None
        self.parent = parent_actor
        self.hud = hud
        world = self.parent.get_world()
        bp = world.get_blueprint_library().find('sensor.other.lane_invasion')
        self.sensor = world.spawn_actor(bp, carla.Transform(),
            attach_to=self.parent)
        # We need to pass the lambda a weak reference to self to avoid circular
# reference.
weak_self = weakref.ref(self)
self.sensor.listen(lambda event:
LaneInvasionSensor._on_invasion(weak_self, event))

@staticmethod
def _on_invasion(weak_self, event):
    
    """On invasion method""
    self = weak_self()
    if not self:
        return
    lane_types = set(x.type for x in event.crossed_lane_markings)
text = ['%r %str(x).split()[-1] for x in lane_types]
self.hud.notification('Crossed line %s % ' and '.join(text))

# ==============================================================
# == GnssSensor
# ==============================================================

class GnssSensor(object):
    """Class for GNSS sensors""
    def __init__(self, parent_actor):
        """Constructor method""
        self.sensor = None
        self._parent = parent_actor
        self.lat = 0.0
        self.lon = 0.0
    world = self._parent.get_world()
    blueprint = world.get_blueprint_library().find('sensor.other.gnss')
    self.sensor = world.spawn_actor(blueprint, carla.Transform(carla.Location(x=1.0, z=2.8)),
    attach_to=self._parent)
    # We need to pass the lambda a weak reference to
    # self to avoid circular reference.
    weak_self = weakref.ref(self)
    self.sensor.listen(lambda event:
    GnssSensor._on_gnss_event(weak_self, event))

@staticmethod
def _on_gnss_event(weak_self, event):
    """GNSS method""
    self = weak_self()
    if not self:
        return
    self.lat = event.latitude
self.lon = event.longitude
#
# """"Class for camera management"""
#
class CameraManager(object):
    """Constructor method"""
    def __init__(self, parent_actor, hud, gamma_correction):
        self.sensor = None
        self.surface = None
        self._parent = parent_actor
        self.hud = hud
        self.recording = False
        bound_y = 0.5 + self._parent.bounding_box.extent.y
        attachment = carla.AttachmentType
        self._camera_transforms = [
            (carla.Transform(carla.Location(x=-5.5, z=2.5),
                             carla.Rotation(pitch=8.0)), attachment.SpringArm),
            (carla.Transform(carla.Location(x=1.6, z=1.7)), attachment.Rigid),
            (carla.Transform(carla.Location(x=5.5, y=1.5, z=1.5)),
                             attachment.SpringArm),
            (carla.Transform(carla.Location(x=-8.0, z=6.0),
                             carla.Rotation(pitch=6.0)), attachment.SpringArm),
            (carla.Transform(carla.Location(x=-1, y=-bound_y, z=0.5)),
                             attachment.Rigid)
        ]
        self.transform_index = 1
        self.sensors = [
            ['sensor.camera.rgb', cc.Raw, 'Camera RGB'],
            ['sensor.camera.semantics', cc.CityScapesPalette, 'Camera Semantic Segmentation (CityScapes Palette)'],
            ['sensor.lidar.ray_cast', None, 'Lidar (Ray-Cast)']]
world = self._parent.get_world()
bp_library = world.get_blueprint_library()

for item in self.sensors:
    blp = bp_library.find(item[0])
    if item[0].startswith('sensor.camera'):
        blp.set_attribute('image_size_x', str(hud.dim[0]))
        blp.set_attribute('image_size_y', str(hud.dim[1]))
        if blp.has_attribute('gamma'):
            blp.set_attribute('gamma', str(gamma_correction))
    elif item[0].startswith('sensor.lidar'):
        blp.set_attribute('range', '50')
    item.append(blp)

self.index = None

def toggle_camera(self):
    """Activate a camera""

    self.transform_index = (self.transform_index + 1) %
    len(self._camera_transforms)
    self.set_sensor(self.index, notify=False, force_respawn=True)

def set_sensor(self, index, notify=True, force_respawn=False):
    """Set a sensor"

    index = index % len(self.sensors)
    needs_respawn = True
    if self.index is None else (force_respawn or
                                (self.sensors[index][0] !=
                                 self.sensors[self.index][0]))
    if needs_respawn:
        if self.sensor is not None:
            self.sensor.destroy()
            self.surface = None
        self.sensor = self._parent.get_world().spawn_actor(
            self.sensors[index][-1],
            self._camera_transforms[self.transform_index][0],
            attach_to=self._parent,
            attachment_type=self._camera_transforms[self.transform_index][1])

        # We need to pass the lambda a weak reference to
        # self to avoid circular reference.
        weak_self = weakref.ref(self)
        self.sensor.listen(lambda image:
            CameraManager._parse_image(weak_self, image))
    if notify:
        self.hud.notification(self.sensors[index][2])
    self.index = index

def next_sensor(self):
    """Get the next sensor"

    self.set_sensor(self.index + 1)

def toggle_recording(self):
    """Toggle recording on or off"

    self.recording = not self.recording
self.hud.notification('Recording %s' % ('On' if self.recording else 'Off'))

def render(self, display):
    """Render method""
    if self.surface is not None:
        display.blit(self.surface, (0, 0))

@staticmethod
def _parse_image(weak_self, image):
    self = weak_self()
    if not self:
        return
    if self.sensors[self.index][0].startswith('sensor.lidar'):
        points = np.frombuffer(image.raw_data,
                                dtype=np.dtype('f4'))
        points = np.reshape(points, (int(points.shape[0] / 4), 4))
        lidar_data = np.array(points[:, :2])
        lidar_data *= np.min(self.hud.dim) / 100.0
        lidar_data += (0.5 * self.hud.dim[0], 0.5 *
                        self.hud.dim[1])
        lidar_data = np.fabs(lidar_data)  # pylint:
        # disable=assignment-from-no-return
        lidar_data = lidar_data.astype(np.int32)
        lidar_data = np.reshape(lidar_data, (-1, 2))
        lidar_img_size = (self.hud.dim[0], self.hud.dim[1], 3)
        lidar_img = np.zeros(lidar_img_size)
        lidar_img[tuple(lidar_data.T)] = (255, 255, 255)
        self.surface = pygame.surfarray.make_surface(lidar_img)
    else:
        image = np.frombuffer(image.raw_data,
                               dtype=np.dtype("uint8"))
        array = np.reshape(image, (image.height, image.width, 4))
        array = array[:, :, :3]
        array = array[:, :, ::-1]
        self.surface = pygame.surfarray.make_surface(array.swapaxes(0, 1))
        if self.recording:
            image.save_to_disk('_out/%08d' % image.frame)

#PBS-- Game Loop -------------------------------------------------------
#PBS-- GAME LOOP

def game_loop(args):
    """Main loop for agent"""
pygame.init()
pygame.font.init()
world = None
tot_target_reached = 0
num_min_waypoints = 21

routeWaypoints = [carla.Location(x=-166.6, y=426.4, z=10),
carla.Location(x=-102, y=-404.7, z=10), carla.Location(x=-69, y=-385, z=10)]

try:

client = carla.Client(args.host, args.port)
client.set_timeout(4.0)

display = pygame.display.set_mode((args.width, args.height), pygame.HWSURFACE | pygame.DOUBLEBUF)

hud = HUD(args.width, args.height)
world = World(client.get_world(), hud, args)
controller = KeyboardControl(world)

# START DATA COLLECTION
# actor_list
actor_list = [world.player]

# Prepare Filename
# filename = 'Locations_' + time.strftime('%Y%m%d-%H%M%S') + '.csv'

# Create and open file
with open(filename, 'w', newline='') as file:
    writer = csv.writer(file)
    writer.writerow(('frame', 'time_sim', 'id', 'type', 'x', 'y', 'z', 'kmh', 'm/s2', 'gear'))
    #world.world.on_tick(lambda snapshot:
    world.get_locations(snapshot))
if args.agent == "Roaming":
    agent = RoamingAgent(world.player)
elif args.agent == "Basic":
    agent = BasicAgent(world.player)
    spawn_point = world.map.get_spawn_points()[0]
    agent.set_destination((spawn_point.location.x,
                           spawn_point.location.y,
                           spawn_point.location.z))
else:
    agent = BehaviorAgent(world.player,
                           behavior=args.behavior)

    spawn_points = world.map.get_spawn_points()
    # random.shuffle(spawn_points)

    if spawn_points[0].location !=
        agent.vehicle.get_location():
        destination = spawn_points[0].location
    else:
        destination = spawn_points[1].location

    destination=carla.Location(x=311.7, y=-261.7, z=10)
    agent.set_destination(agent.vehicle.get_location(),
                          destination, clean=True)

clock = pygame.time.Clock()

while True:
    clock.tick_busy_loop(60)
    if controller.parse_events():
        return

    # As soon as the server is ready continue!
    if not world.world.wait_for_tick(10.0):
        continue

    if args.agent == "Roaming" or args.agent == "Basic":
        if controller.parse_events():
            return

        # as soon as the server is ready continue!
world.world.wait_for_tick(10.0)

world.tick(clock)
world.render(display)
pygame.display.flip()
control = agent.run_step()
control.manual_gear_shift = False
world.player.apply_control(control)

else:
    agent.update_information()

world.tick(clock)
world.render(display)
pygame.display.flip()

    # Set new destination when target has been reached
    if len(agent.get_local_planner().waypoints_queue) <
        num_min_waypoints and args.loop:
        agent.reroute(spawn_points)
        tot_target_reached += 1
        world.hud.notification("The target has been reached " +
                str(tot_target_reached) + " times.", seconds=4.0)

    elif len(agent.get_local_planner().waypoints_queue) ==
0 and not args.loop:
        print("Target reached, mission accomplished...")
        break

    speed_limit = world.player.get_speed_limit()
    agent.get_local_planner().set_speed(speed_limit)
    control = agent.run_step()
    world.player.apply_control(control)

finally:
    # close file
    file.close()

    if world is not None:
        world.destroy()

    pygame.quit()
def main():
    """Main method""
    argparser = argparse.ArgumentParser(
        description='CARLA Automatic Control Client')
    argparser.add_argument(
        '-v', '--verbose',
        action='store_true',
        dest='debug',
        help='Print debug information')
    argparser.add_argument(
        '--host',
        metavar='H',
        default='127.0.0.1',
        help='IP of the host server (default: 127.0.0.1)')
    argparser.add_argument(
        '-p', '--port',
        metavar='P',
        default=2000,
        type=int,
        help='TCP port to listen to (default: 2000)')
    argparser.add_argument(
        '--res',
        metavar='WIDTHxHEIGHT',
        default='1280x720',
        help='Window resolution (default: 1280x720)')
    argparser.add_argument(
        '--filter',
        metavar='PATTERN',
        default='vehicle.*',
        help='Actor filter (default: "vehicle.*")')
    argparser.add_argument(
        '--gamma',
        default=2.2,
        type=float,
        help='Gamma correction of the camera (default: 2.2)')
    argparser.add_argument(
        '-l', '--loop',
        action='store_true',
        dest='loop',
        help='Sets a new random destination upon reaching the previous one (default: False)')
    argparser.add_argument(
        '-b', '--behavior', type=str,
        choices=['cautious', 'normal', 'aggressive'],
        default='normal',
        help='Drive agent behavior (default: normal)')
    return argparser
argparser.add_argument("-a", "--agent", type=str,
               choices=["Behavior", "Roaming", "Basic"],
               help="select which agent to run",
               default="Behavior")

argparser.add_argument(
    '-s', '--seed',
    help='Set seed for repeating executions (default: None)',
    default=None,
    type=int)

args = argparser.parse_args()

args.width, args.height = [int(x) for x in args.res.split('x')]

log_level = logging.DEBUG if args.debug else logging.INFO
logging.basicConfig(format='%(levelname)s: %(message)s',
                    level=log_level)

logging.info('listening to server %s:%s', args.host, args.port)

print(__doc__)

try:
    game_loop(args)

except KeyboardInterrupt:
    print('Cancelled by user. Bye!')

if __name__ == '__main__':
    main()
import carla
import math

def calculate_distance(point1, point2):
    return math.sqrt((point2.x - point1.x)**2 + (point2.y - point1.y)**2)

def calculate_pet(av_location, av_speed, cyclist_location, cyclist_speed, intersection_point):
    av_distance_to_intersection = calculate_distance(av_location, intersection_point)
    cyclist_distance_to_intersection = calculate_distance(cyclist_location, intersection_point)

    av_time_to_intersection = av_distance_to_intersection / av_speed
    if av_speed != 0 else float('inf')
    cyclist_time_to_intersection = cyclist_distance_to_intersection / cyclist_speed
    if cyclist_speed != 0 else float('inf')

    pet = abs(av_time_to_intersection - cyclist_time_to_intersection)
    return pet

# Assume that we already have these values from CARLA environment in each simulation step
av_location = ...  # carla.Location of the AV
av_speed = ...  # Speed of the AV

cyclist_location = ...  # carla.Location of the cyclist
cyclist_speed = ...  # Speed of the cyclist

intersection_point = ...  # carla.Location of the intersection point

pet = calculate_pet(av_location, av_speed, cyclist_location, cyclist_speed, intersection_point)
print(f"PET: {pet} seconds")
Appendix C. Python Code for Calculating TTC (CARLA)

```python
import carla
import math

def calculate_distance(point1, point2):
    return math.sqrt((point2.x - point1.x)**2 + (point2.y - point1.y)**2)

def calculate_ttc(av_location, av_velocity, object_location, object_velocity):
    relative_velocity = math.sqrt((av_velocity.x - object_velocity.x)**2 + (av_velocity.y - object_velocity.y)**2)

    if relative_velocity == 0:
        return float('inf')  # Infinite TTC when relative velocity is zero

    distance = calculate_distance(av_location, object_location)
    ttc = distance / relative_velocity

    return ttc

# Assume that we have these values from the CARLA environment
av_location = ...  # carla.Location of the AV
av_velocity = ...  # carla.Vector3D, velocity of the AV
object_location = ...  # carla.Location of the other object
object_velocity = ...  # carla.Vector3D, velocity of the other object

ttc = calculate_ttc(av_location, av_velocity, object_location, object_velocity)
print(f"TTC: {ttc} seconds")
```