UNIVERSITY OF SOUTHAMPTON

Faculty of Engineering and Physical Sciences
School of Electronics and Computer Science

Towards a Robust and Efficient Traffic Junction Management

by

Phuriwat Worrawichaipat

ORCiD: 0000-0002-9559-3843

A thesis for the degree of
Doctor of Philosophy

February 2024

University of Southampton

Abstract

Faculty of Engineering and Physical Sciences
School of Electronics and Computer Science

Doctor of Philosophy

Towards a Robust and Efficient Traffic Junction Management

by Phuriwat Worrawichaipat

As autonomous vehicles (AVs) are becoming more advanced, the future where all vehicles are connected and autonomous vehicles (CAVs) is highly likely to be a reality. To realise their potential, existing literature has proposed approaches for non-signalised or autonomous junction management to improve the traffic flow efficiency at junctions by utilising the communication capability of CAVs. However, existing methods entail important limitations arising from several simplifying assumptions that are often made. For example, most methods are centralised, which makes the control more straightforward but introduces a computational bottleneck issue, raising concerns about scalability and latency. Also, dynamic features, such as obstructions that could lead to a deadlock, are typically not considered. Moreover, several approaches adopt platooning to improve traffic efficiency, whereby vehicles maintain a close gap between each other and cross the junction as a group. This is usually operated in a restricted and static manner (e.g. using pre-generated and fixed-sized platoons), which can be inefficient. Additionally, only a few approaches have considered key practical constraints, such as the presence of pedestrians at the junction in urban areas, which introduces considerable complexity due to their shared road usage with vehicles. Furthermore, while all existing approaches have been validated at the individual junction level, only a few have tested their performance and impacts at the corridor or network level. It is imperative that these challenges are addressed if CAV-enable signal-less traffic management at junctions in urban areas is to be implemented in practice.

The present thesis addresses the challenges identified in a number of ways. Firstly, a novel computationally decentralised signal-less traffic management approach has been introduced and formulated as a multi-agent system consisting of a manager agent and driver agents. Specifically, the main reliance of the manager agent is alleviated by transferring most of the computation to the driver agents, thereby addressing the bottleneck issue and improving scalability & latency. By having minimal information provided by the manager agent, the driver agents can perform local calculations which are the prediction of crossing paths and resolving conflicts between each other. With similar settings from the state-of-the-art method, i.e. a 3-lane-4-way junction, our method can

address the challenge of the bottleneck at the manager agent and reduce the manager's computation burden, the number of exchanged messages between the manager and driver agents required to perform automation control, and as well as enabling parallel system operation.

Secondly, with possible road obstruction, e.g., a construction site, delaying traffic flow at the junction, we propose a multi-vehicle collision avoidance approach that guarantees vehicles safe crossing and alleviates delays. In particular, vehicles intelligently calculate a path that safely avoids obstruction while using the least possible space. A microscopic traffic simulation, Simulation of Urban MObility (SUMO), is used to model an environment after a practical junction in Manhattan, producing two obstruction scenarios: obstacle at the entry and junction area. By addressing this, our method is more robust to possible obstructions located in the junction vicinity.

To further improve traffic efficiency, we introduce the use of platooning at the signalless junction, utilising the close gap between vehicles and group crossing that majorly increases traffic capacity and reduces delays due to fewer stop-and-go movements. Specifically, we propose an agent-based dynamic platoon formation mechanism, where the manager agent calculates the benefits of forming each platoon in terms of waiting time to determine the optimal platoons' size dynamically. This is to speed up platoon members' crossing movements while minimising the delays of several vehicles waiting for such lengthy vehicles, i.e. platoons, to cross the junction. More importantly, the group's leaders are responsible for members' path prediction and reservation requests, thereby reducing computation load and the number of exchanged messages with the manager agent.

Moreover, to realise the performance of platooning in a more realistic environment, we expand our study to the network level, covering multiple junctions. A real-world case study network from Athens, Greece, is considered, comprising real vehicle movement data from an extensive drone dataset. We calibrate the simulation after such dataset, reproducing the ground-truth traffic demand with the practical corridor geometry and heterogeneous vehicle types, e.g. buses, taxis, and motorcycles, to ensure the realism of the environment as much as possible. Additionally, the randomness of vehicle generation is also introduced in order to reduce the wave-like bias from the dataset due to the use of conventional traffic lights. In this way, our platooning method can be evaluated extensively and realistically.

Lastly, we address the crucial aspect of pedestrian considerations in autonomous junctions. This is to anticipate the future usage of autonomous junctions in urban areas where pedestrians are a crucial system element, especially from a safety and complexity viewpoint. To this end, we introduce a waiting-time-driven approach that dynamically switches between different operational phases at the junction, including pedestrian and freely automated phases. By taking advantage of the uneven traffic flow on different

inbound roads, we can maximise traffic throughput while balancing pedestrian waiting time. These operational phases can be switched dynamically to accommodate varying traffic conditions taking the vehicles and pedestrian waiting time into account, ensuring that the system can adapt to changing circumstances and optimise performance in real time.

In conclusion, our decentralised junction management model can still maintain a similar performance to the state-of-the-art approach. The results show that with single junction scenarios, our approach can reduce the number of exchanged messages by up to ≈40%. For obstruction avoidance, the simulation results show that whenever obstructions exist in the junction area and at the entry our model can maintain the throughput up to 94% and 99% respectively, compared to the no-obstruction baseline. Moreover, with our dynamic platoon formation, the evaluation with a single junction shows that the throughput can be increased by up to ≈12%, and the average travel time can be shortened by up to ≈31% compared to a non-platoon-based state of the art. Furthermore, with a highly realistic corridor environment, the simulations with light and heavy traffic scenarios show that our platooning can reduce the trip duration by up to ≈22% and ≈45% compared to conventional traffic lights and state-of-the-art approach. Additionally, with pedestrians in the system circulation, at the network level with light traffic volumes, our approach can reduce the vehicle trip duration by up to ≈12% and ≈21% with and without platooning compared to traffic light controls. However, the out-performance disappears in heavy traffic scenarios.

Contents

List of Figures

List of Tables

Declaration of Authorship

I declare that this thesis and the work presented in it is my own and has been generated by me as the result of my own original research.

I confirm that:

1. This work was done wholly or mainly while in candidature for a research degree at this University;
2. Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;
3. Where I have consulted the published work of others, this is always clearly attributed;
4. Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
5. I have acknowledged all main sources of help;
6. Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;
7. Parts of this work have been published as:
   Phuriwat Worrawichaipat, Enrico Gerding, Ioannis Kaparias, and Sarvapali Ramchurn. Resilient intersection management with multi-vehicle collision avoidance. *Frontiers in Sustainable Cities*, 3:670454, 2021
   Phuriwat Worrawichaipat, Enrico Gerding, Ioannis Kaparias, and Sarvapali Ramchurn. Multi-agent signal-less intersection management with dynamic platoon formation. In *Proceeding of the 22nd International Conference on Autonomous Agents and Multiagent, AAMAS '23*, pages 1542–1550, 2023

Signed:..... Date:.....

Acknowledgements

First of all, I would like to thank my supervisors, Professor Sarvapili (Gopal) Ramchurn, Professor Enrico Gerding, and Dr. Ioannis Kaparias, who guided me through this PhD study. Professor Gopal provided me with a PhD student opportunity to join the Agents, Interaction and Complexity group, offering consistent inspiration with his innovative ideas, shaping me to be an open-minded person. Professor Enrico consistently provided invaluable advice and guidance, refining my research capabilities. Dr. Ioannis Kaparias always provided many beneficial comments and opinions from the practical or engineering point of view, strongly enhancing the research quality and widening my knowledge. I am sincerely grateful for their support.

Furthermore, I thank my examiners, Dr. Ben Waterson and Dr. Tim Baarslag, for their invaluable feedback and constructive criticism, which explicitly strengthened this thesis. I extend my thanks to the Ministry of Higher Education, Science, Research and Innovation (MHESI), Thailand, for their scholarship. Additionally, I acknowledge the use of the IRIDIS High Performance Computing Facility, and associated support services at the University of Southampton, in the completion of this work.

I appreciate my friends in Southampton who have contributed, in one way or another, to enriching life in the UK. With their shoulders, the stress of pursuing a PhD was greatly reduced. In this regard, I thank the company of all of my colleagues and Thai friends, both senior and junior, who have made this challenging journey profoundly memorable and cherishable.

Finally, my deepest gratitude goes to my family, Mr. Amornsak and Mrs. Thitikarn Worrawichaipat, for their unwavering belief in me and their continual support from Thailand. Without them, I could not have attained this remarkable achievement. I am also grateful to Ms. Saralporn Wichayachart (Grace) for her unwavering support and encouragement from the beginning to the end.

Chapter 1

Introduction

According to a report from the European Union (Alonso Raposo et al., 2019), road transportation is currently following four major development trends: automation, connectivity, decarbonisation, and sharing. These trends are believed to enhance overall transportation efficiency in terms of traffic capacity, congestion, delays and emission. Automation is a primary focus for many car manufacturing companies interested in vehicle automation, striving to build and commercialise their driverless or Autonomous Vehicles (AVs). Moreover, the advancement in information technologies allows vehicles to be more and more connected to either the traffic infrastructure or between themselves, introducing another type of vehicle referred to as Connected Autonomous Vehicles (CAVs). Another report from the Victoria Transport Policy Institute in Canada (Litman, 2020) predicts that shared AVs (self-driving taxis) may become widely available by the 2030s and, by 2060, half of the vehicles on the road will be AVs. The report further predicts that, by the 2080s, all vehicles will be AVs and CAVs. Therefore, it is not difficult to imagine a future where roads are full of AVs and CAVs.

Despite these developments, traffic congestion remains a significant challenge for road transport with negative impacts on many cities worldwide in terms of the quality of living and economy. For example, in London alone, as its streets are continuously congested with traffic, road users spend more than one hundred hours in traffic every year (Alonso Raposo et al., 2019). With a growing population and scarce space, this issue is worsening every day, contributing to rising air pollution and increasing the risk of respiratory diseases. Beyond its impact on public health, traffic congestion also has a significant economic impact. In particular, the high levels of traffic congestion in certain areas can affect people choosing where to live and work while also impacting business decisions about where to invest, due to the increased fuel costs and excessive travel times. This can reduce overall economic activity.

One of the major contributors to traffic congestion is the inefficiency of the current traffic signal control. Cao et al. (2016) suggest that conventional fixed-time traffic light control is inefficient in handling constantly changing traffic conditions and increasing demand, leading to massive delays. To this end, in the last few decades, researchers have investigated ways to optimise the performance of traditional traffic light controllers. The most common optimisation approach is to amend the traffic light controllers with a dynamic adaptation capability. To be specific, the controllers are able to utilise the traffic-related information provided by the real-time road sensors, e.g., the number of waiting vehicles, vehicles' speed, waiting time, and even the environmental condition and adapt their settings to suit the specific traffic condition. Several approaches have been proposed and used practically, such as Sydney Coordinated Adaptive Traffic System (SCATs) (Sims and Dobinson, 1980) and Split Cycle Offset Optimization Technique (SCOOT) (Hunt et al., 1982), which have been implemented in cities in Australia including Brisbane and Melbourne. However, with the driverless car becoming more of a reality, we have an opportunity to make a paradigm change from traffic light controls and leverage the full potential of CAVs, especially from their connectivity capability.

To this end, in the realm of traffic management, there is a growing interest in harnessing emerging technologies that leverage the connectivity of CAVs. Several years ago, an innovative approach to traffic management has gained attention, named Autonomous Intersection Management (AIM). This approach is specially designed to anticipate the potential future where all vehicles are CAVs, and road junctions become non-signalised, essentially, rendering traditional signalized junctions obsolete. In the AIM approach, CAVs engage in a resource reservation mechanism by communicating with road infrastructures and reserving access time slots to cross the junction in a First-Come-First-Serve manner (FCFS). This approach ensures smooth and simultaneous crossing of vehicles, reducing congestion, improving traffic flow, and minimising travel delays. However, many studies, such as (Dresner and Stone, 2008; Carlino et al., 2013), have been designed in a centralised manner. In this setup, a central unit shoulders the entire computational load, handling tasks such as trajectory calculations, conflict checking/verification, and reservation processing. Many wasteful communication efforts between the central unit and vehicles can be seen, especially in (Dresner and Stone, 2008), in the process of avoiding reservation conflicts. Even though the centralised can significantly improve the traffic system and offer simple configuration, monitoring, and maintenance, from a computational perspective, this centralisation design introduces a potential bottleneck issue, resulting in concerns of scalability and potential network congestion, leading to further operation delays. This raises questions about the scalability of such a method.

Moreover, besides optimising traffic flow performance, reducing travel delays, and alleviating congestion, it is crucial to ensure that traffic management models are sufficiently robust to handle unexpected incidents that frequently occur in real-world situations. These incidents include accidents, emergencies, and obstructions, which can significantly disrupt the traffic flow and cause vehicle delays. However, many existing works on traffic management, such as those proposed by (Dresner and Stone, 2008; Vu et al., 2018; Lin et al., 2017; Zohdy and Rakha, 2016), often overlook the issue of obstructions, which can lead to a cascade of impacts and result in traffic gridlock. This omission is particularly concerning since even minor interruptions can cause the system to fail, rendering such methods unreliable. Therefore, there is significant scope for research to develop traffic management models that can effectively handle obstructions and ensure smooth and uninterrupted traffic flow, even in the face of unexpected incidents.

Another opportunity to improve the efficiency of road transportation is to utilise a technology called Adaptive Cruise Control (ACC). ACC is a radar-based system that automatically maintains the safety gap to the vehicle in front. When the CAVs exchange information, this technology can be enhanced further to so-called Cooperative Adaptive Cruise Control (CACC). This enhancement allows vehicles to exchange their intentions and maintain a constant gap between each other while driving as a group or "platoon". This approach offers a significant advantage from a traffic optimization perspective, notably increasing traffic capacity by enabling denser vehicle queues and hastening vehicle junction crossing. Additionally, from the algorithm perspective, this concept further decentralises the system, as the platoon leader is responsible for most of the computation and communication. The benefits of the platoon have been explored in a variety of contexts, e.g. on the freeway (Shladover et al., 2012), heavy-duty vehicles (Liang, 2014; Liang et al., 2015) and urban areas (Lioris et al., 2016). However, platooning studies often assume several restrictive assumptions, i.e. fixed-sized and static platoons, limiting their flexibility. Under the autonomous junction context, due to the long and fixed length of platoons, manoeuvring them through junctions causes many vehicles to be blocked and delayed momentarily. This static design can lead to excessive delays for the majority of vehicles. It is still questionable whether the benefits of platoons can outweigh their drawbacks, especially in high-traffic situations.

Yet other challenges in modelling platoons with autonomous junctions are the realism of the environment setup and study scale. Many simulations are designed in an idealised and restricted manner. For example, Jin et al. (2013) use of a 2-way 1-lane junction as an evaluation scenario, and Bashiri et al. (2018) simulate pre-generated and fixed-size platoons on a 4-way 1-lane junction. These studies explicitly exclude real-world elements, such as practical road geometry, no turning manoeuvres, high traffic demand, and heterogeneous vehicle types. Although the aforementioned studies have shown the potential of platoons with autonomous junctions, it remains unclear

whether platooning can perform well in highly realistic traffic environments where traffic is much more chaotic. Furthermore, many studies tend to focus on the microscopic level (vehicle-based level), such as those by (Bashiri and Fleming, 2017; Bashiri et al., 2018; Bisht and Shet, 2020; Calvert et al., 2020). However, focusing only on the microscopic level overlooks the very important effects at the network level (corridor or city-size scale) and does not provide a comprehensive enough evaluation of any methods developed. These studies may only demonstrate a one-sided view of the platoon's advantages.

Lastly, another essential element for autonomous junction management is the consideration of pedestrians. As traffic lights may become obsolete by the 2040s (Chen and Englund, 2016), managing pedestrian movement presents a significant challenge. While various technologies such as infrared, microwave, and image processing can detect the pedestrian presence, many studies on autonomous junction management still exclude pedestrians as a factor. The inclusion of pedestrians in the automation process can result in complex systems and performance drawbacks. In particular, simultaneously managing pedestrian and vehicle crossing at a junction is particularly complex and challenging as they share the same travelling space. For instance, previous research by Dresner and Stone (2006); Niels et al. (2020) proposed methods that account for pedestrian crossing in autonomous junctions. However, while they performed well in low-traffic scenarios, in high-traffic scenarios, these methods suffer significant traffic throughput drops.

To summarise, despite several studies incorporating the future use of CAVs, especially for the autonomous junction concept, substantial challenges still remain. These challenges cover both traffic optimization aspects, such as scalability, adaptability, robustness, assessment scope, and practicality, as well as computational considerations, including bottleneck issues. Addressing these challenges is essential before practically implementing autonomous junction management, especially within urban environments. Therefore, this thesis aims to tackle these challenges head-on by developing decentralised and robust algorithms tailored to CAV junction management. These algorithms are designed to dynamically adapt to complex traffic conditions in urban settings and accommodate miniature to large-scale assessment environments. The overarching goal of these proposed algorithms is to enhance the overall efficiency of junction management in anticipating futuristic real-world scenarios.

In the following sections, we present the specific research challenges that this thesis seeks to address.

1.1 Research Challenges

Our motivation for these research challenges arises from three key considerations:

First, with the emergence of CAVs, we aim to harness their potential to address specific issues in road traffic optimization. One key problem is scalability, when the traffic area grows larger and larger every day while certain traffic management may not be able to keep up with the demand in the future. Additionally, congestion remains a persistent challenge in urban areas and proving good traffic management to growing traffic demand is rather challenging. Our research seeks to leverage the capabilities of CAVs to alleviate these problems and enhance traffic flow.

Second, we recognize the importance of managing the computational loads associated with these systems as they scale to meet real-world requirements. The primary problem lies in addressing the computational bottlenecks that can lead to network congestion and operation delays. Striking a balance between computational efficiency, affordability, and maintainability is another key research goal.

Lastly, our research emphasizes the necessity of developing robust and adaptive systems capable of handling complex real-world scenarios. This includes the ability to respond effectively to road obstacles, ensuring road safety, and maintaining an efficient traffic flow. Additionally, the element of pedestrians is considered a crucial factor for implementing in the crowded city where they are prevalent. Our primary research challenge is centred on creating autonomous systems that can dynamically adapt to these imperative scenarios while sustaining the efficiency of traffic management.

At its core, our research is centred around optimizing junction management models, recognizing that road junctions are critical traffic bottlenecks in road transport systems (Chen and Englund, 2016). By enhancing junction traffic performance, we can have a profound impact on traffic flow, even from a single junction level to overall road network efficiency. We investigate different junction optimisation methods, either signalised or non-signalised junctions, along with the innovative applications that utilise CAV's capabilities. Accounting with the three considerations above, our literature highlights multiple existing challenges in autonomous junction management.

In more detail, the research challenges tackled in this thesis are as follows:

1. **Computationally-decentralised resource reservation:** The resource reservation for non-signalised junctions has been studied in various contexts, encompassing heuristic resource allocation, economic incentives, different types of junctions, and even networks of multiple junctions (see Section 2.3 for a literature review). Many resource reservation models have adopted a centralised-based approach due to its flexibility, simplified communication between agents, and adaptability to various scenarios. However, a common limitation of these existing approaches lies in their centralised nature, resulting in a significant computational bottleneck at the central control unit. Redundant computations and communications occur

when DAs fail to acquire reservations, leading to inefficiencies such as network congestion, operation delays, and scalability issues.

To address these challenges, a plausible approach involves distributing the main computation to individual vehicles, distributing the load to multiple nodes and leveraging the computation power of each one. Developing decentralised solutions to overcome these challenges remains an open area of research.

2. **Junction management with collision avoidance:** Road transportation is subject to various uncertainties, including potential obstructions in the junction. Ignoring this scenario can lead to inefficient, unreliable management systems and, even worse, pose risks to road safety. Therefore, enhancing the robustness of such systems becomes a crucial task, especially in the context of autonomous junctions, where automation is the sole reliability. One straightforward solution is to close the obstructed road or lanes, but this approach leads to considerable travel delays and, more significantly, leaves a substantial portion of the junction space unused. The assumption that obstructions completely block the road is rather impractical.

   Nevertheless, despite the challenges, there have been only a small number of studies that enable vehicles to utilize the entire junction space intelligently while avoiding collisions with obstructions during the crossing. Hence, there is a need for innovative approaches that ensure efficient utilization of space and ensure safety when confronted with obstructions, increasing the robustness of the junction control.

3. **Realistic evaluation environments:** An issue with many existing studies on autonomous junction management is their reliance on idealised assumptions, which simplify the scope of the study and avoid complexity. To do so, multiple realistic elements of road transport are deliberately excluded, e.g. the use of a homogeneous vehicle type, vehicle dynamics, simplified driver behaviour and artificial junction and road geometries. However, by omitting these elements, the performance assessment within their studies hardly reflects the true nature of the practical traffic environment, which is unlikely to show the comprehensive effectiveness of their method.

   To ensure that proposed methods can be effectively applied in practical scenarios, it is crucial to incorporate realistic evaluation elements. This not only aids in understanding the methods' capabilities but also helps identify optimal conditions and assess their adaptability to dynamic traffic scenarios.

4. **Dynamic platoons:** While platooning application has demonstrated numerous advantages in various contexts, including motorways, heavy-duty vehicles, and road junctions, many studies have been conducted under restrictive conditions,

mainly due to increased traffic capacity. These platooning models often lack essential dynamics. For example, platoons typically enter and leave the focus environment as static groups without the ability to join or break apart. Furthermore, these models often maintain a fixed platoon size throughout the evaluation, and platoons are formed in an ad-hoc and greedy manner. The rigidity of these models can lead to concerns regarding their function in a wide range of traffic conditions, as platoons may require excessive space or time slots to traverse junctions or roads, causing delays for other vehicles.

Despite the recognized benefits of platooning, these oversimplified models, operational constraints, and limited consideration of dynamic traffic scenarios may only present a one-sided view of their advantages. Furthermore, it is essential to investigate the degree of flexibility of platoon formation that allows platoons to adjust to various conditions and substantially enhance their adaptability, especially against dynamic real-world traffic scenarios.

5. **Multi-junction platoon control:** Many platoon application studies have primarily focused on assessing their performance at the microscopic level, which involves individual junctions and vehicle-based operations. While their approaches provide valuable insights into the behaviour and efficiency of platoons within junctions, they overlook critical effects that may manifest at a larger scale, often called the macroscopic level.

   Specifically, the challenge lies in understanding how platooning impacts traffic flow, congestion, and mobility on a larger scale, beyond isolated junctions. It is important to shift toward practical evaluation settings considering multiple junctions within a real-world road network, encompassing various traffic conditions and scenarios. A more comprehensive understanding of the macroscopic impact of platooning and its potential to enhance traffic flow in urban environments can be gained.

6. **Integration of pedestrians control:** Over the past decades, several techniques have been used to detect and locate pedestrians allowing many studies, either with conventional traffic signal control or autonomous junction control, to include pedestrians in their studied environments. Despite these technologies and the accurate information gained, optimising both pedestrians and drivers for the autonomous junction remains a significant challenge. Many studies have demonstrated that pedestrian right-of-way presents a significant burden to the system, leading to a drastic drop in traffic throughput. This is due to the need to share the road space with pedestrians, introducing complexities and challenges in traffic management. Addressing this challenge is crucial for the practical implementation of autonomous junctions in crowded scenarios, such as urban areas with a high prevalence of pedestrians.

1.2 Research Contributions

To address the above challenge, this thesis makes the following contributions:

- We propose a novel method for decentralised junction management in a traffic environment full of CAVs addressing Research Challenge 1. In our model, each vehicle is responsible for computational tasks related to generating crossing paths and resolving conflicts with other vehicles. At the same time, the junction manager is only responsible for validating requests from the vehicles. By distributing the computational workload in this manner, we can significantly reduce the computation burden on the junction manager, substantially improving the system's scalability. Moreover, our model retains an essential feature of the state-of-the-art discretisation method, i.e. allowing dynamic vehicle trajectories. Specifically, driver agents can reserve imperfect curves or line driving paths that change according to the junction geometry (asymmetric or tilted).

- We propose junction management with a multi-vehicle collision avoidance algorithm to enhance the robustness of traditional approaches, addressing Research Challenge 2. Our approach addresses the problem of having obstructions in the middle or at the entrance of the junction. With the existence of obstacles, the performance of traditional approaches drops considerably, which may lead to a road network problem, i.e., a gridlock. On the other hand, in the proposed approach, the vehicles are capable of intelligently calculating a collision-free crossing path and its safe time slot. The collision-free path utilises the available space in the junction area as much as possible. This capability prevents severe impacts on the system and ensures a smooth junction crossing, whether the obstruction is at the junction entrance or in the middle area. Overall, our approach offers a more reliable and efficient solution to junction management, providing good traffic performance even when junction areas are obstructed.

- We evaluate a wide range of scenarios utilising a state-of-the-art microscopic simulation tool, SUMO, which explicitly models many elements to reflect reality. The realism of the simulation is largely due to the detailed consideration of vehicle behaviour, partially addressing Research Challenge 3. For example, vehicle dynamics such as weight, speed, acceleration, braking, and centrifugal force are carefully considered. Additionally, driver behaviour is taken into account, e.g. lane-changing behaviour to ensure distributed road usage and prevent any lane from becoming overly occupied. Furthermore, we employ a car-following model that ensures all vehicles operate in a safe manner, maintaining a 2.5 m safety gap between vehicles while slowing down or speeding up. Ultimately, SUMO allows

us to simulate traffic environments with a high level of detail down to the individual vehicle level, providing an accurate representation of the movements of all vehicles throughout the simulation.

- We propose a new resource reservation junction management mechanism that forms the platoons dynamically to address Research Challenge 4. Our mechanism is developed to address ad-hoc platoon formation, which is done greedily and usually without consideration of system-level impact. In our mechanism, we consider that the larger the platoon, the more it will cause conflicts with the other vehicles, which is due to the platoon's large required time slots and space. Without optimising or constraining the platoon's size, the junction will experience more delays rather than reducing them. To resolve this issue, the manager agent is entrusted with an additional task calculating the benefits and costs of forming a platoon in terms of overall waiting time to maximise the benefits to the system. The maximum platoon size will be determined dynamically in real-time, constrained by how crowded the junction is.

- We generalise the autonomous junction management system with dynamic platoon formation to cover multiple junctions, specifically, a road corridor, to realise the performance of platooning at a macroscopic level. This contribution mainly addresses Research Challenge 3 & 5. Additionally, the simulation is calibrated with heterogeneous vehicle types, practical junction geometry and realistic traffic demand, utilising a real-world vehicle movement dataset. This is to replicate the ground truth traffic in Athens, Greece and extensively explore the impact of platooning not only at the individual junction level but also at the corridor level. Also, our method is evaluated against conventional traffic lights and FCFS under light and heavy traffic scenarios.

- We propose a pedestrian model under the autonomous junction environment utilising a concept of primary and secondary roads, defined by the amount of traffic volume. This is to address Research Challenge 6. To this end, we propose a waiting-time-driven decision tree that dynamically switches between four different operational phases/controls for specific conditions, utilising an advantage from uneven traffic volume between primary and secondary. The main objective is to maximise the traffic flow without discomforting the pedestrians (using a predefined waiting time limit as a hard constraint). We explore the compatibility of autonomous junction management within an urban setting, where the resources (space and time slots) are used with multiple road users, namely, vehicles and pedestrians.

The list of publications that has arisen from this thesis is provided in appendix A.

1.3 Thesis Outline

The chapters of this thesis are structured as follows:

- Chapter 2 introduces the background and related work in the area of junction management.
- Chapter 3 details the algorithms and structural model behind decentralised junction management.
- Chapter 4 presents our resource reservation junction management with dynamic platoon formation.
- Chapter 5 details the calibration process reproducing a highly realistic road corridor and evaluates our proposed method.
- Chapter 6 introduces our pedestrian model in autonomous junction environment for urban area usage.
- Chapter 7 serves as a conclusion for this thesis and some directions for any future work.

Chapter 2

Literature review

In this chapter, we present the literature review necessary to put the research objectives and challenges into context. This chapter is structured as follows. First, in Section 2.1, we discuss the future of ground transportation and its existing issues, i.e., traffic congestion, and we also address various research methods for junction management. Next, in Section 2.2, we detail traffic signal control approaches used to optimise the performance of signalised junctions (junctions with traffic lights). Moreover, we also detail several techniques for non-signalised junction management in Section 2.3.

Later, the uncertainties in the traffic environment and their counter plans are discussed in Section 2.4. Moreover, in Section 2.5, we discuss some mechanisms of the platoon applications and their context of studies along with their potential challenges. Lastly, Section 2.6 details the topic of pedestrians in autonomous junctions along with several approaches that handle pedestrian right-of-way.

2.1 The Future of Ground Transportation

According to a report from Alonso Raposo et al. (2019), a variety of revolutions in transport have been developed continuously, including automation, connectivity, decarbonisation, and shared mobility. Such developments start to redefine the future of road transportation.

Here, automation refers to a system that can autonomously perform real-time traffic management. Vehicles with automation capabilities are called Automated Vehicles (AVs). Normally, AVs are categorised into five different levels of automation which are driver-only, assisted, conditional automation, high automation and full automation (Alonso Raposo et al., 2019) (see Fig. 2.1). Particularly, these levels define who is in charge of the driving, i.e., humans or the machine, and they range from level zero where vehicles are entirely driven by humans to level five where vehicles are entirely

controlled by automated driving systems. Currently, these automated driving systems are still under development and testing due to many challenges, such as safety, operational delays and driving efficiency. However, different vehicle brands and models are already offering automation features level from one to three. For example, Tesla's Autopilot and Nissan's Propilot are considered to be level two automation while Audi A8 is the first car that can achieve level three automation (LeBeau, 2019). Moreover, Arbib and Seba (2017) estimate that by 2030, 95% of distance travelled in the US will be served by AVs. Due to these potentials, it is not difficult to imagine a road environment full of fleets of AVs.

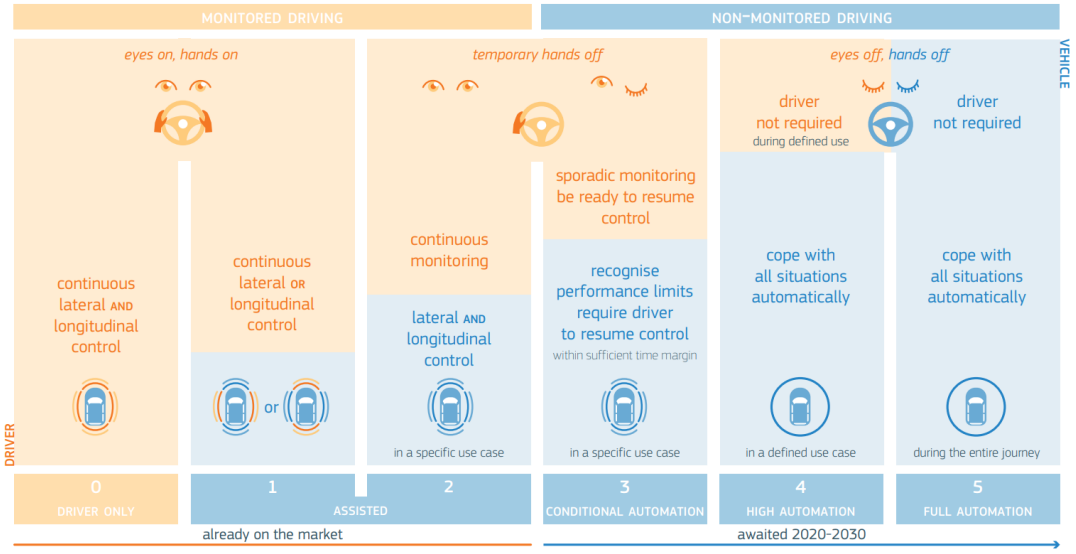

FIGURE 2.1: Levels of driving automation for on-road vehicles.
Source: Alonso Raposo et al. (2019)

Furthermore, another essential development is connectivity. Many vehicles nowadays are equipped with mobile-connected devices able to form connections with other devices via vehicular ad hoc networks (VANETS). With these connections, an individual vehicle can share the information provided by the sensors with other vehicles through vehicle-to-vehicle communication (V2V) or vehicle-to-infrastructure communication (V2I). Example of infrastructures that utilise these connection technologies includes real-time signage, toll charging systems, or junction management systems. A combination of these two communication types is referred to as V2X. To be specific, the AVs with these communication capabilities are usually called Connected Automated Vehicles (CAVs). To this end, this supportive information exchange technology enables the concept of a cooperative transportation system.

Moreover, decarbonisation is another aspect of road transportation development. Currently, many research departments are focusing on substituting fossil fuels with eco-friendly alternative ones such as hydrogen, biofuels, natural gas, and electricity, aiming to reduce greenhouse gas emissions and oil dependency (Alonso Raposo et al., 2019).

For instance, different models of electric vehicles (EVs), e.g., plug-in hybrid electric vehicles (PHEVs) and hybrid electric vehicles (HEVs), have been released to the market following that purpose. Unfortunately, these EVs have not been widely adopted yet due to the inadequacy of models and their high selling price compared to conventional vehicles. However, several researchers, e.g., (Sperling, 2018; Lutsey, 2018), believe that by 2025 a variety of EV models will be available to the market, and their prices will reduce to be close to the conventional vehicles. Moreover, a study from Blanco et al. (2019) shows that by the end of 2050, the number of EVs in use will reach at least 75 million.

Lastly, sharing mobility focuses more on the transportation strategy side, where users can gain temporal access to some transport modes as needed. Example of this innovation comes in various forms, such as car sharing, bike sharing, carpooling, and on-demand ride-sharing (Greenblatt and Shaheen, 2015). Nowadays, travellers are less likely to use traditional taxi services. Instead, ride-sharing or ride-hailing services are becoming more popular day by day since it is an upgraded version of taxis involving different innovations and technologies Zhong et al. (2022). Specifically, these services are alternatively referred to as 'Mobility-as-a-Service' (MaaS). (Commission, 2017) suggests that this innovative development is believed to reduce not only the amount of on-road vehicles but also the negative impacts of road transport, e.g. less traffic and emission.

Based on these four developments, it is not difficult to imagine a future where the road in urban areas will be full of advanced or "smart" forms of transportation. This could bring a significant impact on road transport, especially in lowering its cost. For instance, the operational cost of various types of ride-sharing could be significantly reduced due to autonomous driving vehicles. However, no matter how advanced transportation technologies are, the challenge of traffic congestion remains due to the rapid population growth and space limitation (Alonso Raposo et al., 2019). Traffic congestion has been a problem in various cities for many decades. In the US, congestion alone is accounted to cost approximately $305 Billion to the cities and drivers (Schneider, 2018). Moreover, in London alone, commuters spend more than 100 hours on the road due to the continuously congested road (Alonso Raposo et al., 2019). Without solving this issue, the full potential of future road transport cannot be realised.

Accordingly, many researchers tried to tackle the problem of traffic congestion by utilising vehicle communication, VANETs, and later introducing a new research field called Intelligent Transportation System (ITS). Moreover, since road junctions are considered to be the bottlenecks of the traffic flow, many proposed ITS models attempt to optimise the efficiency of the junction management (Chen and Englund, 2016). Normally, the junction management approaches are categorised into two types, signalised and non-signalised. We next discuss advanced methods in traffic light control used to optimise the signalised junction.

2.2 Traffic Lights Control

The most common way to manage bottlenecks at the junction is to use traffic lights. Originally, traffic lights or signals control can be traced back to 1914, when the first electric traffic lights were introduced and installed in Cleveland, Ohio, USA. At that time, the traffic lights alone could efficiently manage the traffic and successfully ease down the bottlenecks due to low traffic demand. In contrast, due to the rapid increase in the number of cars, population and the constraints of the cost of expanding road networks, traffic conditions have worsened. This inspires many researchers keen to improve the existing traffic light controls. Normally, the development of traffic lights is categorised into two types, fixed-time and traffic-responsive control.

The fixed-time traffic lights control is a method that defines lights' patterns based on the statistic of the past traffic flow. Due to the fact that similar traffic patterns repeat themselves every day, the daily traffic demands can be broken down into several sections based on time period, i.e. peak hours in the morning and evening, off peaks in the afternoon and nightfall. Then, the traffic light patterns for each section are formulated and applied, which switch the patterns corresponding to the time of the day. According to Heung et al. (2005), SIGSET (Improta and Cantarella, 1984) is one of the systems that is commonly used to determine fixed-time patterns. It can optimally determine the green time of each phase and the cycle time, even though the phase sequences must be fixed. However, this particular control lacks the input of real-time traffic, which is considered a major drawback. Due to this, traffic-responsive control was proposed to compensate for this drawback.

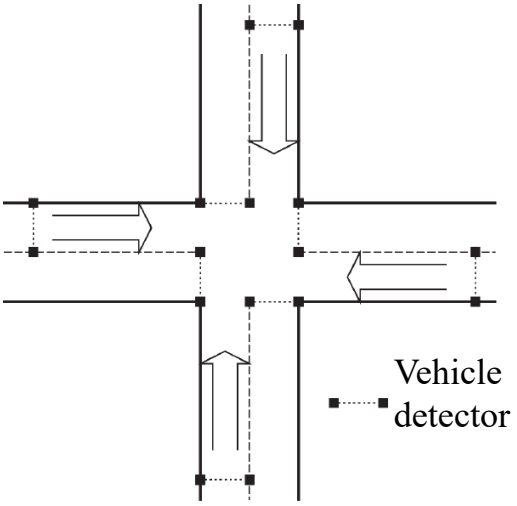

FIGURE 2.2: The example placement of vehicle detector.
Source: (Heung et al., 2005)

Specifically, traffic-responsive control is an online process that optimises the green time using real-time traffic data as input. With the extra hardware cost, especially the detectors (see Fig. 2.2), traffic-responsive control can adapt to variations in real time. The main idea is to predict the traffic demand at specific time and optimise the traffic light cycles corresponding to that demand in order to achieve a better performance, e.g. fewer delays and more throughput. The example of conventional traffic-responsive controls can be seen extensively such as vehicles-actuated signals (Akcelik, 1994), SCAT (Sims and Dobinson, 1980), SCOOT (Hunt et al., 1982), and RHODES (Mirchandani and Head, 2001) where the traffic demand is detected by loop detectors and surveillance devices. These approaches have been implemented in many cities worldwide, such as London, UK and Melbourne, Australia. In what follows, we highlight a few most recent approaches that relate to our work.

2.2.1 Optimisation Models

A number of studies, e.g., (Marcotte, 1983; Lo, 1999; Lin and Wang, 2004; Beard and Ziliaskopoulos, 2006; Abdelgawad et al., 2010; Aziz and Ukkusuri, 2012) have looked into the problem of traffic management from a mathematical perspective. In the work by Aziz and Ukkusuri (2012), a framework with dynamic traffic lights assignment was introduced in order to minimise the total queuing delay. This framework mainly addressed two important issues: the lost time caused by the phase switches and the cycle length. As a result, the framework was able to assign adaptive cycle lengths that respond to different traffic conditions while considering the route choice behaviour of drivers. On the other hand, other studies consider objective functions and constraints differently, e.g., (Abdelgawad et al., 2010; Kesur, 2010) where the impact of left-turn restriction and length of left-turn bays are accounted for and, and in (Kikuchi and Kronprasert, 2010; Li et al., 2010) maximum green-light phase settings were optimised.

2.2.2 Fuzzy Logic Approach

Fuzzy logic approaches address the issue that a simple logical value cannot possibly represent real-world traffic conditions and objectives. Fuzzy logic approaches define a degree of truth as a real number between 0 and 1 instead of the usual integer 0 and 1. Typically, fuzzy logic is used to represent the vagueness and imprecise information such as the degree of beautifulness or the level of sweetness. In a complex setting such as road transportation, several dynamic conditions, e.g., traffic patterns, weather conditions, road conditions and even levels of traffic congestion, are defined more precisely, and these values can be utilised through the fuzzy control system.

In more detail, several studies, e.g., (Pappis and Mamdani, 1977; Collotta et al., 2015; Trabia et al., 1999), have proposed the use of fuzzy logic in traffic light control. In (Pappis and Mamdani, 1977), a fuzzy controller for a two-phase junction (East-West/North-South) is introduced, and it is capable of dynamically managing the duration of the green-light phases. However, since it is specifically designed for the two-phase junction the turning movements of the vehicle have not been taken into account. In comparison, Collotta et al. (2015) introduced a controller for a four-phase junction where the turning movements are considered in the managing process as well. This allows the junction to respond to traffic conditions more effectively than the conventional one.

In addition, studies by (Trabia et al., 1999; Murat and Gedizlioglu, 2005) introduced an extension of Pappis and Mamdani's work called a two-stage fuzzy logic controller. The first stage determines which phase is needed to be handled, and the second stage is responsible for the green light duration management which is similar to (Pappis and Mamdani, 1977). Specifically, in the first stage, the ineffective crossing phase (forward or turning) is identified by an estimated number of waiting vehicles at the junction using induction loops and video cameras as detectors. However, these detectors provide inaccurate results which could lead to poor system performance. In contrast, to address this problem, Collotta et al. (2015) introduced multiple fuzzy logic controllers with an adaptation of IEEE 802.15.4 Wireless Sensor Network (WSN) that can provide the accurate number of vehicles in the queue. This allows them to handle the inaccuracy of vehicle detection and also enhances the controller's performance.

2.2.3 Machine Learning Approach

One of the most common optimisation approaches for the traffic lights controller is to use machine learning. For example, in (Spall and Chin, 1997), a System-wide TRaffic Adaptive Control (S-TRAC) is proposed, which is based on Reinforcement Learning (RL). This traffic lights controller uses RL with neural networks taking real-time traffic conditions and environmental parameters such as the weather condition as input and outputs a suitable timing for the signal phases. However, the performance is not guaranteed as both traffic and environmental conditions constantly fluctuate. Therefore, this traffic lights controller has to tune itself on a daily basis to maintain its performance.

Moreover, (Li et al., 2001) also proposed self-learning traffic light controllers integrating with neural networks. Their model is composed of two neural networks that frequently change their state between learning and working during the self-learning process. The self-learning process is completed with the decision made from a performance evaluation unit that evaluates the efficiency of two neural networks by measuring the traffic flow results. Simulation results show that this method can improve performance over

traditional traffic light control. However, one limitation of this work is that the initial setting for self-learning neural networks is hard to determine.

The main benefit of machine learning is utilising the big data of the traffic, especially when the world is extensively connected through the internet. In particular, the usage of devices can be captured and stored in real-time, producing a large amount of data recorded. The more data learned using machine learning techniques, the more intelligent such system will be. However, there is one drawback of using machine learning is the limited usage of a developed system. It is difficult to avoid bias in the dataset as different cities or countries have different driving behaviour, e.g. sometimes working hours are different. This prevents developed systems from working well when it comes to the general environment.

Due to the connectivity of many devices becoming more available day by day, another approach that has been used to optimise the traffic junction is the Multi-agent System (MaS) approach, which will be discussed next.

2.2.4 Multi-agent Systems Approach

The term agent refers to a computer system that is located in a certain environment and capable to perform so-called autonomous actions making changes to the environment and achieve the objective it is originally designed for (Wooldridge, 2009). Indeed, some researchers use the concept of multi-agent in the road traffic environment objectively to improve the traffic, i.e. reduce delays and congestion. Wiering (2000) was one of the first that proposed a multi-agent reinforcement learning integrated with traffic light control. They describe the use of RL algorithms for learning traffic lights controller to minimise the overall traffic delay with the integration of intelligent agents. Specifically, they considered both the traffic lights controller and vehicle drivers as agents. Then, they use a standard RL learning algorithm (Q-learning) (Watkins and Dayan, 1992) to adjust the behaviour of each agent. In this way, the system adapts to the needs of both the individual drivers and the controller. Nevertheless, the consideration of agent units can be different in various studies. For example, in true adaptive traffic signal control with RL proposed by Abdulhai et al. (2003), they only consider the traffic light controllers as agents while the vehicles are not.

In the same vein, de Oliveira et al. (2004) proposed an interesting multi-agent model to optimise traffic lights, but it is based on swarm intelligence. In their work, the junction controller is behaving like a social insect that is responsible for changing or performing a specific task, i.e., generating signal plans. However, each task has to be stimulated first. The stimulation usually comes from virtual pheromones constantly produced by the vehicles that are waiting for the green lights. The main drawback of this approach

is that it requires a significant amount of time to converge to a stable condition due to the high degree of dynamism in the traffic environment.

Next, we discuss another type of junction management called non-signalised junction management.

2.3 Non-Signalised Junction Management

As mentioned in Section 2.1, due to the rapid advances in the development of autonomous vehicles and several supportive technologies for vehicular communication, it is not difficult to imagine a road full of CAVs. In anticipation of such a future, many researchers have started investigating the idea of non-signalised junction management. Simply put, a non-signalised junction is a junction without traffic lights, and all the vehicle coordination relies on V2V and V2I communication. The access of the junction can be scheduled and queued using only connection between vehicles or vehicles with the infrastructure. One of the most well-known approaches used to manage non-signalised junctions (sometimes referred to as autonomous junctions) is a multi-agent approach proposed by Dresner and Stone (2008). The model interaction sequences between vehicle agents and the junction agent perform a resource reservation mechanism. The resource, in this case, refers to the space and time slots for access to the junction area. In particular, any vehicles that want to cross the junction have to ask for a time-slotted reservation from the junction agent. However, generally, the resource can be considered in a few different ways depending on the objective of each study. This resource reservation idea is mainly developed in two ways: centralised and distributed. In the next section, we discuss the centralised approach of resource reservation and its different adaptation objectives in more detail.

2.3.1 Centralised Resource Reservation

Generally, the centralised system is designed to have a central unit where most of the processes and actions are done or passed through this unit. For example, Dresner and Stone (2008) proposed a junction control mechanism called First Come First Serve (FCFS). that is shown to outperform the traditional traffic lights and stop signs. Specifically, the space in the junction is discretised into a grid of cells, and this method is normally called cell-based modelling. Each cell acts as a resource that can be reserved for a particular vehicle for a specific amount of time, depending on the vehicle's trajectory. To perform this reservation, they modelled the traffic as a multi-agent system including two main types of agents: Driver Agents (DAs) who drive the vehicles and Intersection Manager Agent (IMA or junction manager agent) who acts as a coordinator. Note that their model assumes that all vehicles on the road are CAVs.

Since our work is based on this work, we explain the resource reservation mechanism in more detail. Specifically, IMA and DAs exchange messages between each other to perform a scheduling decision when DA can access the junction. The IMA and DAs are interacting sequentially as follows:

1. When a vehicle enters the IMA communication range, the vehicle's DA sends a reservation request message containing all of the necessary parameters such as arrival velocity, arrival time, and vehicle characteristics to the IMA. The DA who performs this step is called a requesting DA.

2. Then IMA starts simulating an optimal trajectory of the requesting DA using the provided vehicle properties. The simulated trajectory is projected on the junction grid to generate a set of used cells with specific time stamps. This operation step can be referred to as "path prediction."

3. The IMA uses this set of cells to check a conflict against the previously granted reservations. The conflict occurs when at least one cell in the set is going to be reserved by two vehicles. An example of this situation is illustrated in Fig. 2.3.

   - If no conflict is found, a time slot reservation is granted to the requesting DA, and this DA begins crossing the junction. The granted response message contains some restriction details ensuring DA safe crossing. Meanwhile, IMA also holds the information of this reservation too.
   - Otherwise, the IMA rejects this request, and the requesting DA has to wait for a certain period before sending a new request.

   This can be referred to as "conflict checking".

4. Once DA completes its crossing DA notifies its departure back to the IMA. Then, the reserved cells are released.

Even though some DAs' requests may be rejected, a newcomer cannot gain a reservation overtaking the one that arrives earlier, following the FCFS rule. The DAs' arrival order determines the sequence of their access to the junction. The overall interactions between DA and IMA can be seen in Fig. 2.4.

Initially, this FCFS model was developed based on the assumption that vehicles drive with a static velocity until they complete their crossing. In a follow-up study by Dresner et al., improvements were made where some practical scenarios, left-turn and right-turn traffic, are accounted for. Moreover, the vehicles' velocity is not considered to be static. They are allowed to accelerate while crossing.

Furthermore, this resource reservation mechanism can be applied to other types of junctions as well. For example, in (Bento et al., 2012), roundabouts and crossroads

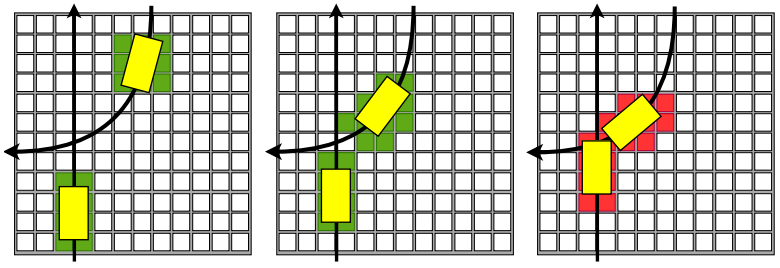

FIGURE 2.3: An example of conflict between two vehicles. The arrows represent the trajectories taken by each vehicle. These three images show cells (in green) that will be used by vehicles step by step, starting from the left. The right-most image shows a situation where a cell is going to be used by two vehicles.

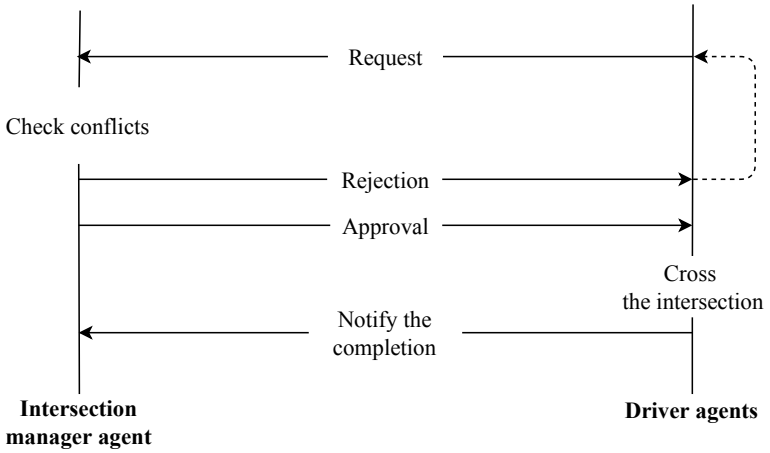

FIGURE 2.4: The interaction sequences between DA and IMA performing the resource reservation mechanism.

are considered. In the case of the roundabouts, the approaching vehicle must send its information to the junction agent to notify its arrival and intention and then wait for a response similar to (Dresner and Stone, 2008). The grid discretisation of roundabouts is illustrated in Fig. 2.5. In turn, in the case of crossroads, the junction agent must detect the driver's intention by itself.

On the other hand, several research studies attempt to integrate economic incentives with the resource reservation mechanism. Theoretically, in FCFS, vehicles acquire a reservation according to the order of arrival. However, from the economic perspective, individual people are more likely to value the limited resource, in this case, time, differently. For example, a driver who is in a hurry may value the junction passing more than ordinary commuters. Such value may be translated into a price that the driver may want to pay to cross the junction. This incentive-based approach has been studied in several works including (Schepperle et al., 2007; Vasirani and Ossowski, 2012; Carlino et al., 2013).

In more detail, Schepperle et al. (2007) introduced an auction-based junction management algorithm named Initial Time Slot Auction (ITSA). In their work, the vehicles

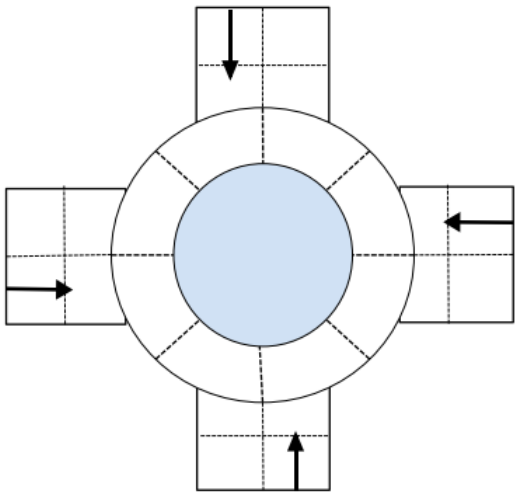

FIGURE 2.5: An example of space discretisation of a roundabout. The arrows represent traffic directions.
Source: (Chen and Englund, 2016)

crossing are not scheduled by order of arrival. They propose two types of auctions: basic auctions and auctions with subsidies. In the basic auction, only one vehicle in each direction is allowed to bid, and vehicles can join the bidding process when the preceding vehicle already has been allocated. Simply put, only the first non-allocated vehicle in each direction is allowed to bid. This means that no matter how long succeeding vehicles are in the queue, they do not affect the auction result. Therefore, to resolve this issue, the proposed auction with subsidies is introduced. The preceding vehicles are subsidised by the succeeding vehicles and join the auction as a group instead of an individual. The group with the highest bid will receive an allocated time slot and cross the junction together as a platoon. From their empirical evaluation, this subsidisation method can significantly reduce the overall waiting time and even outperforms the basic auction.

In the same vein, Vasirani and Ossowski (2012) proposed a market-inspired approach for junction management built upon the resource reservation mechanism of Dresner and Stone (2008). They extend the reservation mechanism by applying a combinatorial auction (Krishna, 2009) instead of the FCFS. As they empirically demonstrate, their auction-based algorithm can guarantee reduced delays to the drivers who value the resource the most, in this case, time. However, the overall delay is not guaranteed to be reduced especially in rush hour situations. This happens because some vehicles initially have a high bidding power value while ordinary vehicles, which can be considered as commuters, have a significantly lower bidding power. Additionally, they also scale up the junction management from the individual junction to a network of junctions and manage to balance the allocation of resources in a scalable way.

To summarise the benefits of the centralised non-signalised junctions, firstly, it can ensure that the available space in the junction will be fully utilised since it is occupied by different vehicles most of the time. The second benefit is that maintenance is rather simplified as the system was designed in a centralized fashion. Third, this resource reservation mechanism offers a high degree of flexibility for adaptation since it can even be applied to many junction shapes and also different types of junctions, e.g., roundabouts and crossroads. Fourth, many extensions can be made to include, for example, economic incentives and multiple junctions.

High Computational Load at The Central Controller.

However, despite the benefits of the centralised resource reservation method, it comes with a major drawback - a high computation load at the central unit or bottleneck issue. In this design, the central unit is burdened with not only communicating with DAs but also performing resource reservation operations, including path prediction and conflict checking. Furthermore, the IMA does not provide or suggest a possible conflict-free timing for DAs, resulting in the continuous receipt and rejection of numerous unsuccessful request messages. Consequently, the central unit faces a full computation load at all times, wasting the system's overall computation resource.

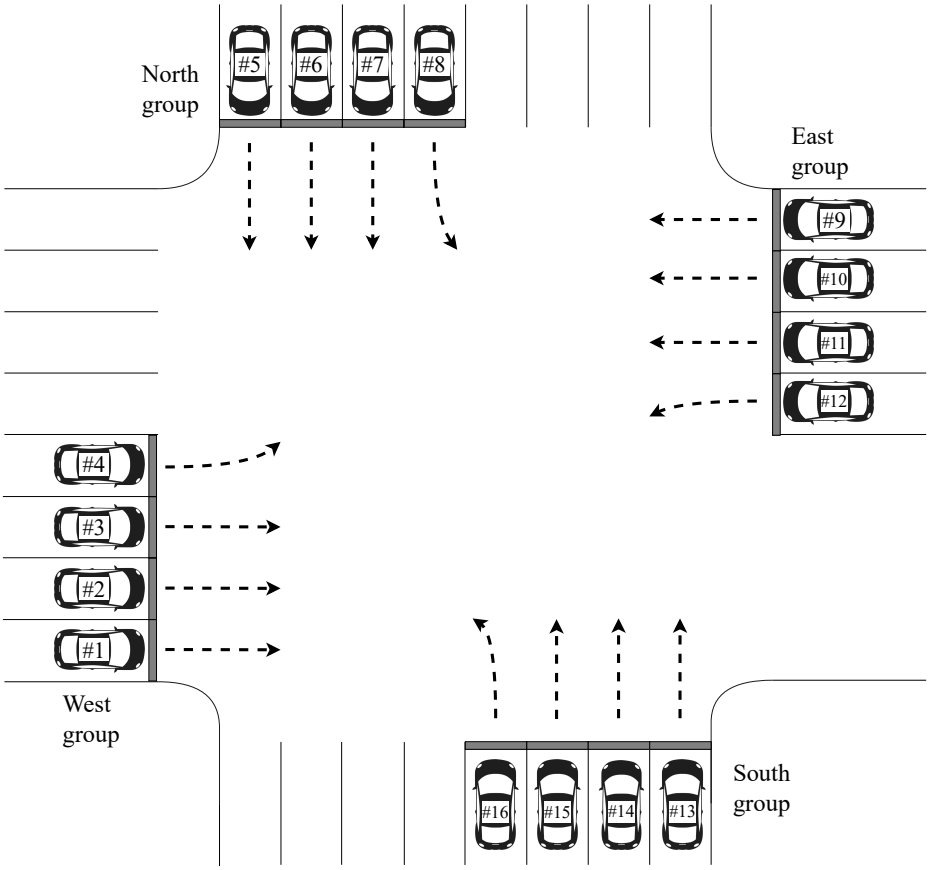

FIGURE 2.6: An example of a 4-way 4-lane junction with 16 cars would like to cross the junction simultaneously.

In order to demonstrate the computation load that the IMA faces, let us consider a case study involving a 4-way, 4-lane junction. Imagine that 16 vehicles enter the junction almost simultaneously in a clockwise manner, and they would like to cross the junction, labelled from #1 to #16. It is important to note that each inbound vehicle from the left wishes to make a left turn, leading to potential conflicts with the vehicles on the opposite side. We can categorize these vehicles into four groups based on their inbound directions: north, south, west, and east. The illustration of this situation can be seen in Fig. 2.6

As these vehicles approach the junction, they begin sending reservation requests to the IMA, which receives these requests consecutively, one by one. Upon receiving the requests, the IMA verifies them through internal simulations, specifically path prediction and conflict checking. The computational cost of these simulations depends on the granularity level of cell-based discretization, with higher levels of discretization leading to increased computational expenses. Initially, the IMA does not have any existing reservations. Therefore, the West group receives immediate approvals for their reservation requests, while the requests from the North, East, and South groups are rejected due to conflicts arising from the West group's reservations. Consequently, these vehicles will slowly approach the stop bar, awaiting their turn to receive approval.

Afterwards, every DA at the stop bar that unsuccessfully acquires reservations has the ability to send consecutive requests to the IMA, as stated in (Dresner and Stone, 2008). The IMA cannot approve the requests by the north group as their requested spaces are currently occupied by the vehicles in the west group. At the same time, the east and south groups could not retrieve any reservation approvals either, as the IMA strictly follows the FCFS principle (the north group had arrived first). The IMA will continuously receive these requests and run a significant number of internal simulations without returning a single approval, thereby facing an expensive computation load for a certain amount of time. These redundant requests before a successful one are considered wasteful efforts, leading to the IMA expending all the computation resources without any meaningful outcome.

To further demonstrate the total computation load of the IMA, assuming the speed limit for crossing the junction is 7 m/s (approximately 25 km/hr), and the longest crossing trajectory is 30 m, the DAs cross the junction at a constant speed of 7 m/s. Therefore, the vehicles would need a maximum of 30/7 = 4.28 s to cross the junction. Since the computation load occurs when there are repetitive unsuccessful reservation requests, we focus on the north, east, and south groups. Each group needs approximately 4.28 s to wait for the occupied space to become free. During this time, the IMA will continuously run internal simulations to handle their requests. This means the IMA will be under full load for approximately 4.28 s for each of these three groups.

In total, the IMA will experience a full computation load for approximately 4.28 x 3 = 12.84 seconds when handling these four groups of vehicles in the given scenario. While this study case only considers 16 vehicles, in real-world scenarios, the number of vehicles is much higher. For instance, during rush hour, the traffic demand can reach anywhere from 6,000 to 10,000 vehicles per hour, depending on the area. This raises a concern about the vast amount of computation resources that would be wasted in such busy scenarios.

In conclusion, allowing the DAs and IMA to engage in wasteful communication could lead to the IMA operating at full load at all times, which may have several negative effects on the computation devices. These effects could include performance degradation, high power consumption, overheating, and other potential issues that might impact the system's reliability and maintenance costs. Additionally, it is crucial to consider the impact of the number of exchanged messages. The continuous back-and-forth communication between the IMA and DAs not only consumes computational load but also results in a significant number of messages being sent and ultimately wasted. Such issues can lead to overcrowding of communication channels, potentially causing delays or latency to other wireless devices sharing the same medium.

Next, we discuss distributed approaches for the resource reservation mechanism, another efficient method that purposely reduces reliance on the central unit.

2.3.2 Distributed Resource Reservation

Even though the centralised approaches offer simplified management in terms of design, configuration, monitoring and troubleshooting, they have some disadvantages. As stated, one of them is the computation bottleneck at the central unit, which can lead to scalability issues where the central unit is shouldering more computation if the traffic management scope expands. The factor of scalability in traffic optimisation is considered to be crucial as it ensures that advancements can benefit a wide range of road users, rather than being limited to specific groups or scenarios.

To mitigate such issues of the centralised approaches, the concept of performing resource reservation in a decentralised or distributed manner has been introduced. This idea leverages the advancement of communication among vehicles or V2V sharing global data that is essential for resource reservation. With the shared data, individual node (in this case, vehicles) has the capability to perform some calculation internally, distributing the burden across multiple units and utilising each one's computational power. By doing this, the bottleneck issues can be alleviated, reducing the potential reliance on a single node and opening up the opportunity to expand the scope of the system.

To this end, Naumann et al. (1997, 1998) were among the first to propose a distributed reservation approach using the concept of tokens. A token, in this context, represents an occupancy of the resource. Specifically, when two vehicles attempt to cross the junction, and a conflict occurs, a specific token will be given to only one vehicle, granting a right to use that conflicted area. The token holder vehicle constantly broadcasts its occupancy to other vehicles, and once this vehicle has left the conflicted areas, this token will be released. At the same time, other vehicles in the waiting area continuously listen to the broadcast messages and look for any available tokens in order to avoid any potential conflicts. This decentralised approach is shown to be free from any deadlocks and collisions through set-up experiments. Moreover, the authors also increase the level of complexity of the system by proposing a priority-based fairness token reservation mechanism. In addition, this mechanism is designed to serve different types of vehicles with different priorities. For example, vehicles such as ambulances, fire engines, and police vehicles have a higher crossing priority than ordinary commuters.

Furthermore, another distributed junction management approach was proposed by Vu et al. (2018). They proposed a model based on multi-agent coordination, Distributed Constraint Optimisation Problem (DCOP). The challenge in their work is how to convey data to each node and maintain data consistency, referred to as message passing. Vehicle nodes necessitate sharing some essential data, and it must be ensured that the data is consistent (convergence point), preventing any undesired outcome. To do so, they employ the max-sum ADVP algorithm since it is proven to outperform several message-passing algorithms known for its speed in reaching convergence point (Vu et al., 2018). Upon utilising an advanced message-passing algorithm, they also ensure that traffic performance is optimised.

Moreover, the scale of their model focuses on both a single junction and a network of junctions. Even with a large study scope like the network of junctions, the max-sum ADVP can guarantee to converge within an acceptable rate, demonstrating its scalability advantage. Furthermore, the interesting element of this work is the choice of representing conflict areas as they opt to represent them with a set of fixed points. They consider their environment as static, and vehicles must cross the junction on fixed paths. Essentially, with certain assumptions, they simplify the conventional cell-based area discretisation (Dresner and Stone (2008)) while still maintaining its full use. The advantage of this simplification lies in the size of messages passing between vehicle nodes, reducing the computation and communication cost of the overall system. An example of the static conflict points is illustrated in Fig. 2.7.

Even though this work can address the bottleneck problem presented in the centralised resource reservation mechanism, it can introduce challenges under certain simulation settings. In particular, prioritising emergency vehicles can lead to longer average waiting times for other vehicles. Moreover, since this approach heavily relies on exchanging

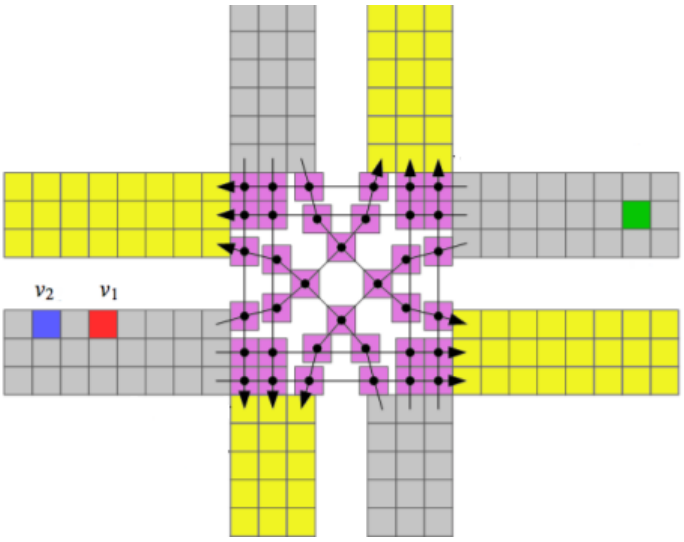

FIGURE 2.7: A cell-based junction model with static conflict points (boxes with dot). The arrows represent possible trajectories of vehicles.
Source: (Vu et al., 2018)

messages between driver agents, it incurs a high communication cost and a significant amount of time to reach a stable state or converge, especially in crowded junctions.

In summary, distributed systems come with their challenges and disadvantages. One of the drawbacks is increased complexity. Managing multiple nodes across a network can be challenging, leading to potential difficulties in configuration, monitoring, and troubleshooting. Additionally, such systems may experience data consistency problems, as synchronization across nodes can be complicated, leading to potential data conflicts and undesired behaviours. In essence, distributed systems require intricate design to mitigate or address such challenges proactively.

Moreover, the failures can also come in a physical manner. In practice, the failure of hardware is one of the common incidents in traffic scenarios. This failure can lead to vehicle breakdowns and even accidents that could result in road obstruction. Therefore, in the next section, we will delve into uncertainties within the traffic environment and discuss strategies and countermeasures to address them."

2.4 Uncertainties in Traffic Environment.

We have mentioned several traffic management approaches that can dynamically adapt to the vary in traffic conditions, e.g., machine learning and multi-agent-based traffic lights (Section 2.2.3 and 2.2.4), and also the junction models that anticipate the potential full CAVs traffic environment. However, most of the studies in the traffic management field aim to optimise traffic flow or system efficiency, while only a few consider the

fact that the real world is full of unforeseen incidents, such as emergency access or accidents.

To deal with the emergency, one common solution for this is to add a priority level to each vehicle. Specifically, for example, in (Naumann et al., 1998; Vu et al., 2018), several important vehicles, such as ambulances, police vehicles, and fire engines, are labelled as high-priority vehicles. These high-priority vehicles always have a right to cross the junction first, ensuring faster navigation for these vehicles through the traffic network. Still, the exchange cost is the increased average delays on non-emergency vehicles.

Nevertheless, another type of unforeseen incident can come in the form of road obstructions, often attributed to vehicles' hardware failures or accidents. To address this issue, (Iša et al., 2006) proposed a controller model manipulating traffic lights in an environment of non-autonomous vehicles, and their model is adaptable to accidents at any unpredictable spots. They applied RL on the traffic lights controller and obtained better flow performance in both normal situations and situations where accidents occur. In the simulation, the accidents were modelled to occur at random points on the lane with the highest vehicle input rate. Once an accident occurs, the entire street (more than one lane) will be marked as defective, and it is impossible for any vehicles to go through. In their study, all the vehicles that attempt to go to the accident spot will be rerouted in a different direction without interrupting the overall planning since a specific destination was not defined initially. In this way, the defective street will not cause overflow queuing or any deadlocks and allow traffic to flow normally.

Although they can address the issue of blockage from accidents and maintain a good traffic flow, a shortcoming is that a significant amount of space in the defective street is left unused. The model assumes that the entire road becomes unusable due to the accident, without considering the actual size of the road obstruction. In practice, there may be opportunities to employ the unaffected portions of the road for vehicle passage, thereby potentially achieving better traffic flow. The consideration of obstruction size offers a promising way for optimization, particularly employing obstruction avoidance in the operation. Addressing this challenge holds the potential to further enhance traffic management strategies.

2.5 Platoon Applications

There have been considerable developments in the field of vehicle automation. Most notably, several cars nowadays are equipped with a smart radar-based system called Adaptive Cruise Control (ACC)), This system allows vehicles to automatically maintain a safe gap to the vehicle in front automatically. However, in the early state of ACC, inconsistency and stability seem to prevent explicit road traffic improvements, inspiring many transportation researchers studied on this issue. For instance, Rajamani (2011)

shows that ACC needs a headway time of roughly one to two seconds as the safety gap, which is the same gap when compared to human drivers. Similarly, (Shladover, 2007; Bergenhem et al., 2010), work on the evolution of ACC called Cooperative Adaptive Cruise Control (CACC). Their studies focus on improving communication among the involved vehicles reducing their latency and aiming for real-time communication. As a result, CACC can provide better safety even when vehicles drive at close distances. This enables the development of *platooning* to be a more efficient traffic management strategy.

Specifically, the term *platooning*, in this thesis, indicates cooperative driving applications in which automation is combined with connectivity, allowing vehicles to synchronise their movement and manoeuvre among themselves in order to behave as a group (Segata et al., 2014). In recent decades, platooning has become a well-known topic in transportation-related studies due to its promising benefits. For example, Fernandes and Nunes (2011) proposed a safe and stable operation for intra-platoon information management that significantly impacts platoon stability. While Öncü et al. (2014) have looked at platooning from a network perspective, incorporates the effect of sampling, hold, and network delays due to the imperfection of wireless communication, such as transmission delays, limited bandwidth and overlapped communication channels. In addition, Shladover et al. (2012) study the impact of platooning on the freeway showing that with moderate to high percentage market penetration CACC can significantly increase traffic capacity. For more specific purposes, Liang et al. (2015); Liang (2014) apply platooning with the heavy-duty vehicles aiming to reduce fuel consumption due to less air drag.

Furthermore, the platooning idea has also been applied to more specific locations that are considered the main traffic bottlenecks, especially in urban areas. For instance, Lioris et al. (2016) uses platooning on the junctions and demonstrate that when platooning is integrated into a small network of 16 junctions, their approach can achieve 200 to 300% increase in traffic capacity (compared to fixed-time traffic lights). Similarly, Mamouei et al. (2018) have demonstrated the benefits of platooning in terms of increasing road capacity, reducing delays and decreasing pollutant emissions and fuel consumption. Additionally, a study in Calvert et al. (2020) conducts a platooning field test on five consecutive traffic lights, showing an average of 5% reduction in travel times.

However, within the context of traffic junction optimization, some may suggest that conventional signalized traffic control systems can already generate platoon-like behaviour, often referred to as "consequential platoons." These are formed by queuing vehicles before releasing them collectively, resembling platooning to some extent. Nevertheless, it is important to note that platoons created through CAV communication capabilities offer significant advantages over consequential platoons, primarily due to one key factor: uniformly small time headways.

To deliberate the benefits of platooning in more detail, a study by Lioris et al. (2016) provides explicit insights. They evaluated junction performance by comparing a typical signalized junction with platoon-enabled communication capabilities to one without, where the red-green signal control remained unchanged. Their results reveal substantial improvements when CAV platooning is introduced, including reductions in queuing length and queuing delays, as well as a substantial increase in throughput. This research lays down a crucial point – CAV platoons and consequential platoons are fundamentally different. CAV platoons, with their emphasis on maintaining short headways, offer substantial advantages and greater benefits in terms of traffic efficiency.

Subsequently, many researchers in modelling autonomous junctions for CAVs have turned their attention towards platooning, taking advantage of the platoon's cooperative controls. For example, Jin et al. (2013) focus on the autonomous junction for CAVs and propose multi-agent junction management with platooning. A heuristic platoon formation is used in their work where the vehicles will be grouped as soon as they enter the junction communication range. A simulation on a 2-way junction shows that even with a heuristic algorithm. Their method can shorten average travel time up to 8% compared to the non-platoon-based method (Dresner and Stone (2008)). Bashiri and Fleming (2017) propose a stop-sign algorithm that allows platoons to cross the junction one at a time, reducing the fuel consumption by up to 13% and advancing the prior work by giving priority to the platoon that has the highest waiting time in Bashiri et al. (2018). Consecutive study where a rule-based platooning approach for intelligent traffic lights is presented, mainly prioritising the highest cost platoons to reduce overall travel time proposed recently Bisht and Shet (2020).

Interestingly, in the field of automated junctions for CAVs, platooning can be interpreted differently. Instead of having vehicles follow each other in the same lane, a platoon can be formed using vehicles on any incoming lanes to cross the junction as a group. The particular method is referred to as "1-dimension platoon" or "virtual platoon"[1]. Several examples of this particular platoon can be seen in Masi et al. (2018); Kwon and Chwa (2014); Medina et al. (2017); Zhou et al. (2022). Particularly, in Zhou et al. (2022), the space before entering the junction is divided into three operating rooms comprising of the cooperative zone, buffer zone, and virtual gate (see Fig. 2.8). The idea behind this method is to precisely manipulate the speed of grouped vehicles to cross the junction as smoothly as possible, thereby reducing stop-and-go delays. However, the main drawback of this method is that it requires a huge space to perform, which is around 100m. This would work well in rural areas or motorways where junctions are far spaced-out, but applying this method to urban ares where space is more limited could be problematic.

However, the aforementioned studies are usually simulated and evaluated in idealised scenarios, which are rarely seen in real-world traffic. Several essential elements are

[1] https://drive.google.com/file/d/1qTEkkJzI5aenuHQD107yakwEf1fKSIdq/view?pli=1

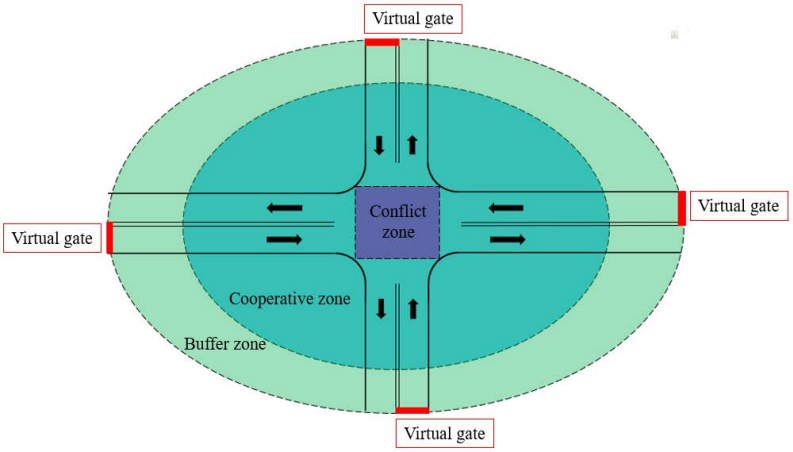

FIGURE 2.8: This figure shows operation zones used to manage virtual platoons. Source: (Zhou et al., 2022)

rarely considered, including road geometry, turning manoeuvres, realistically high traffic demands, heterogeneous vehicle types, and a variation in vehicle speed. For example, in Jin et al. (2013), the studied junction is an artificial 2-way-1-lane simulating relatively low traffic demands. Similarly, Bashiri et al. (2018) use a 4-way-1-lane junction with pre-generated platoons assuming the groups have been formed as arrival with unchangeable size. Even in the recent study in Zhou et al. (2022), the setting of the scenario is quite simple only demonstrating the superiority of their method against the non-platoon-based state of the art. Additionally, many studies, e.g. (Bashiri and Fleming, 2017; Bashiri et al., 2018; Bisht and Shet, 2020; Calvert et al., 2020), implicitly and unrealistically assume that vehicles are identical.

As such, while many studies demonstrate that platooning works well in idealised and deterministic junction management scenarios, it is unclear whether it works equally well in practice. More importantly, many of the aforementioned studies are done at the microscopic level showing the performance only in small-scale aspects. This is likely to overlook the important effects on the system at a larger scale. The challenge here is to expose platooning to highly realistic scenarios which opens up a good opportunity to fully understand the actual impact and performance of platooning at the macroscopic level, especially when traffic demands change during the day time.

2.6 Pedestrian with Autonomous Junction

In reality, the presence of pedestrians is rather an essential element that cannot be ignored. Pedestrians can play a critical role in the autonomous system as they greatly affect the performance of junctions. Specifically, pedestrians negatively impact performance by increasing the waiting time of vehicles and also decreasing the junction's

throughput. Additionally, safety is also an aspect needed to be considered as well. Otherwise, the autonomous junction will not be practical in such crowded areas, i.e. the urban city. Therefore, autonomous junctions must operate in a way that ensures the safety of pedestrians while still maintaining a good flow.

Unfortunately, many studies in junction modelling often exclude the consideration of pedestrians and prioritise their work on optimising the traffic throughput and delays. Only a few works are addressing this challenge. One example is the work from Dresner and Stone (2006). They attempt to address this challenge of pedestrians by introducing a heuristic solution allowing pedestrians and bicycles to cross the junction. However, their approach is still not responsive to the change in pedestrian demand. Similarly, Kothuri (2014) introduces a responsive traffic light control objectively to minimise vehicles' delay and amount of stopping, but the result shows that it ends up increasing waiting time for pedestrians. Many researchers conclude that the general problem in consideration of pedestrians is the lack of connectivity among pedestrians or between pedestrians and infrastructure, making it difficult to accurately determine their presence, demands, positions, or intention. Hence, such systems cannot be more dynamic in responding to the change in pedestrian demand. To address this, a few computer vision studies propose methods for pedestrian detection, see surveys in Gavrila (2001); Gandhi and Trivedi (2006, 2007); Rasouli and Tsotsos (2019). Many alternative detection technologies were suggested, including infrared, microwave, usage of mobile devices, and image processing. Using these innovative technologies enables researchers to acquire useful data, and one of the valuable pieces of information is the estimated number of pedestrians.

By having such valuable information, handling or considering the presence of pedestrians in traffic circulation become conceivable. For example, Niels et al. (2020) proposes demand-responsive pedestrian phases where the algorithm is designed to limit the maximum waiting time of pedestrians. A micro-simulation is used to conduct experiments with different sets of pedestrian demand scenarios. Despite their demand-responsive mechanism, in extremely high traffic demands, their approach ends up increasing vehicle delays. The major issue is that they handle and reserve time slots for pedestrians one by one. It appears that the right-of-ways for pedestrians are given too often, and delays are pushed away towards vehicles instead. Moreover, Chen et al. (2020) also introduce a system that takes pedestrian into account and simulates various traffic scenarios. They demonstrate a trade-off between vehicles (drivers) and pedestrians, in which "delays of pedestrians and vehicles are negatively correlated". Recently, Wu et al. (2022) proposed an automated transport unit for the autonomous junction, called "Automated Pedestrian Shuttle" or APS. In their work, pedestrians are also considered in reservation circulation, acting as a shuttle car. Fig. 2.9 shows the cell discretisation used in this work where the junction area is divided into cells in a similar

manner to (Dresner and Stone, 2008). These cells represent reservations for both pedestrians and vehicles that are used to identify any potential conflicts. The shuttle can carry pedestrians across the street in vertically, horizontally and diagonally. However, their work is yet to evaluate with the practically heavy traffic.

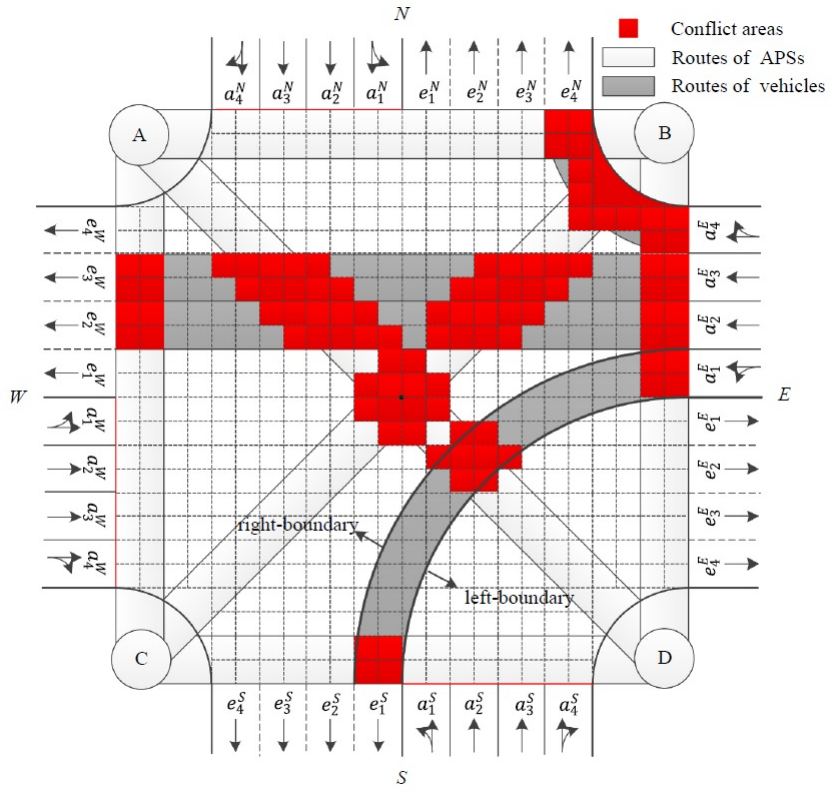

FIGURE 2.9: This image shows the discretised area within the junction which will be used for both APS and vehicles.
Source: (Wu et al., 2022)

In conclusion, even though many studies of autonomous junction management with pedestrians can be seen, there is still no consensus on how to handle pedestrians appropriately. It appears that pedestrian consideration is rather a sensitive and complex matter for autonomous junction modelling since it causes several drawbacks to such system, e.g. less throughput and more vehicle delays. The challenge here is to create a pedestrian-friendly system that also concerns other road users. In other words, we have to balance the level of service between drivers and pedestrians and explore the advantages and disadvantages of such system, determining which situations work best.

2.7 Summary

We have conducted a comprehensive literature review on both signalized and non-signalized junction management. Various studies have proposed different types of

traffic light control systems for signalized junctions, which have shown significant improvements in traffic flow. However, with the rapid advancements in CAV technologies, the concept of fully autonomous driving has emerged. In this context, traditional signalized junctions may not fully exploit the potential benefits offered by CAVs. As a result, the idea of non-signalized junction management using a multi-agent system has gained attention, where all junction crossings are entirely dependent on communication among CAVs.

Recent studies have explored traffic management models based on this non-signalised approach, employing either centralised or distributed strategies. One of the most commonly known centralised junction management approaches is the FCFS method, which is regarded as the state of the art in many studies. One of the main advantages of FCFS is its flexibility to accommodate various extensions, such as different types of junctions and multiple junctions. Additionally, it can handle dynamic vehicle trajectories effectively.

However, there are notable drawbacks to this approach, namely the high computation load burden or bottleneck issue on the central unit, leading to scalability and potential hardware degradation concerns. These concerns raise the necessity of exploring alternatives that can mitigate these shortcomings while preserving the valuable properties of FCFS. One such approach is addressing Research Challenge 1, which pertains to decentralised resource reservation junction management. This innovative approach aims to tackle the challenges posed by FCFS, reducing the reliance on the central unit while maintaining its beneficial features.

Furthermore, we discuss the uncertainties of real-world traffic where various unexpected incidents could happen. One form of uncertainty is an emergency case which has already been addressed in a few studies by introducing a priority level for each vehicle to the system. Some vehicles such as police cars, firefighter trucks, and ambulances, are labelled as important vehicles, and they always have a right to cross the junction before ordinary vehicles. Besides, another form of incident could come in the form of an accident that causes an obstruction and results in defective streets or lanes. One of the solutions that considerably softens the impact is a vehicle rerouting solution. However, this rerouting is yet to be considered as the optimal solution since it significantly worsens overall travel delays where certain roads are left unused. Only a few existing works focus on Research Challenge 2, junction control with collision avoidance where the performance is reasonably maintained even when obstructions exist.

Moreover, we review the applications of the platoon that utilise CACC technology allowing vehicles to maintain a constant gap between each other and drive as a group or *platoon*. Many positive results can be seen throughout a variety of transportation studies such as reducing COx & NOx emission, increasing the traffic, and shortening overall delays on autonomous junctions even with ad-hoc platoon formation. However, a

number of challenges of platooning are still presented which involve the dynamism or flexibility of platoon controls, including joining or breaking platoons and defining the optimal platoon size. To date, there is little work addressing platoon control in a dynamic manner (Research Challenge 4). Moreover, several introduced platoon control models rarely consider real-world traffic elements, i.e. realistic road geometry, high traffic demand, heterogeneous vehicle types, etc. It is not clear that such studies can perform equally well in practice. Therefore, Research Challenge 3 & 5, namely realistic evaluation environment and multi-junction platoon control, are necessary to be addressed.

On top of that, we review an essential matter of pedestrians when it comes to implementing autonomous junctions in urban areas. In the past few decades, technologies in computer vision have evolved significantly, which majorly helps researchers to detect and estimate pedestrian presence, and position, and even able to predict their intention. With such valuable information about pedestrians, many researchers in autonomous junction modelling started considering pedestrians in their design, introducing pedestrian-adaptive models. However, taking pedestrians into circulation is quite sensitive to a system like an autonomous junction as it causes a huge disadvantage, especially the traffic throughput, due to the shared road usage of pedestrians and vehicles. Therefore, Research Challenge 6 is a great challenge considering the prevalence of pedestrians in practical urban areas.

After we have discussed the related works relevant to our study along with existing challenges, we continue to present our contributions addressing these challenges, beginning with our resilient junction management approach with the computationally decentralised mechanism.

Chapter 3

Resilient Junction Management with Computationally Decentralised Mechanism and Collision Avoidance

In this chapter, we introduce a computationally decentralised approach to junction management to alleviate the high computational load experienced by the central agent unit, often observed in centralised methods such as those presented in (Dresner and Stone, 2008; Vasirani and Ossowski, 2012). We also integrate collision avoidance strategies into our proposed method to ensure smooth traffic flow in the presence of road obstructions. This method directly addresses Research Challenges 1 and 2.

In this contribution, we consider the work of Dresner and Stone (2008) as the state of the art where all of the computation is responsible by one central unit. Their algorithm is designed as a multi-agent system composed of an intersection management agent (IMA) or junction manager agent as the central unit and driver agents (DAs). To coordinate vehicles crossing the junction, the IMA and DAs perform a resource reservation mechanism by exchanging request and confirmation messages with each other (see Section 2.3.1 for more detail). Additionally, we introduce a crucial consideration—road obstructions—which is often overlooked in existing autonomous junction studies.

Although there are two types of agents in the system, the junction agent is the one who is responsible for most of the computation, path simulation/prediction and conflict checking. On the other hand, the DAs are only responsible for sending their properties and arrival time to the IMA and do not perform any computation. Moreover, in the process of preventing conflict between DAs' reservations, many repetitive communications between the IMA and DAs are being made without achieving any meaningful

outcomes. This leads to redundant messages exchanged and a high computation load at the IMA at all time. The example impacts of these issues are overcrowding the communication channel, causing message delays and also affecting the central controller in terms of degradation. Therefore, we propose a decentralised junction management approach where the IMA is no longer responsible for any computation except granting a confirmation and holding a piece of information keeping track of the current reservation state.

Furthermore, we address road obstructions by allowing DAs to navigate through the junction intelligently, utilising available space efficiently while avoiding obstructions. While doing so, we ensure that the collision avoidance movements cause minimal effect on other DAs by reducing on-spot lane changing and maintaining smooth traffic flow even in obstructed scenarios.

The structure of this chapter unfolds as follows. Firstly, we delve into the background of collision avoidance. Next, we detail the modeling of traffic elements, including obstacles, vehicles, lanes, and junctions. Subsequently, we expound on the behavior of driver agents within our system, elucidating their responsible operations: path prediction, collision avoidance algorithms, and conflict resolution. Following this, we elaborate on the behavior of the junction manager agent and the interaction design with driver agents, addressing inherent issues. We then outline how we uphold the First-Come-First-Serve principle within our system. A comprehensive empirical evaluation follows, assessing our proposed method in terms of computational loads and robustness against obstructions. Finally, we engage in a discussion, evaluating the performance of our method and acknowledging encountered limitations during development.

This chapter is structured as follows. Firstly, we discuss the background of collision avoidance. Secondly, we describe the modelling of traffic elements, i.e., obstacles, vehicles, lanes, and junctions. Thirdly, we describe the behaviour of the driver agents within our system, detailing their responsible operations: path prediction, collision avoidance algorithm, and conflict resolution. Fourthly, the chapter elaborates on the junction manager agent's behaviour along with some issues that shape the interaction design with the driver agents. Fifthly, we then outline how we uphold the First-Come-First-Serve principle within our system. Following this, we continue with an empirical evaluation, assessing our proposed method in terms of computational loads and robustness against obstructions. Lastly, we engage in a discussion, evaluating the performance of our method and acknowledging encountered limitations during development.

3.1 Background of Collision Avoidance

Our collision avoidance implementation requires a high level of vehicle autonomy. Therefore, we draw inspiration from research in collision avoidance, safe navigation, and path planning in multi-robot systems, such as (Kim and Kwon, 2015; Pudics et al., 2015; Rebai et al., 2009; Savkin and Wang, 2014; Savkin and Li, 2018). Each of these studies proposes various approaches to tackle the problem.

For instance, Kim and Kwon (2015) introduced an algorithm using an ordinary camera, while Pudics et al. (2015) utilised a 360-degree camera as a sensor. In contrast, Savkin and Li (2018) relied solely on a 2D range finder, which provides simpler processing and proved to have sufficiently accurate measurements (Pudics et al., 2015). Given our purpose to maintain algorithm simplicity and computational efficiency, we build upon the approach presented in Savkin and Li (2018).

In (Savkin and Li, 2018), a unicycle robot equipped with a 360-degree 2D range finding sensor navigates an environment filled with obstacles. Using sensor information, the robot can identify several tangent lines between itself and the safe distance from the obstacles. These tangent lines are perpendicular to the safe distance, which is drawn further from the edge of the obstacles. The robot then selects a single tangent line to follow and heads toward the corresponding edge. Once the robot's distance to the obstacle's edge is close to a predefined safe distance, it maintains that safe distance while moving along the boundary of the obstacle (see Fig. 3.1). The robot remains in a closed area while continues exploring the area until it has fully covered it. Once the space is fully explored, the robot completes and terminates its operations.

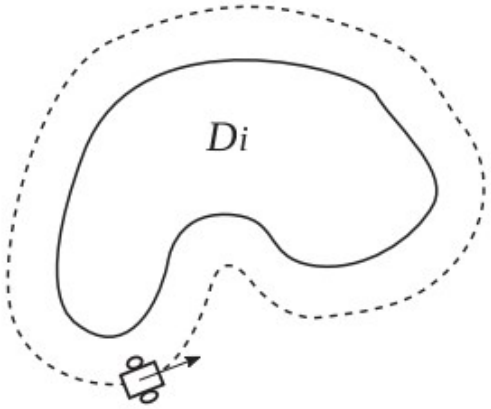

FIGURE 3.1: An example movement, dashed line, of a robot that keeps a safe distance around an obstacle D_i
Source: Savkin and Li (2018)

However, it is essential to recognise that the context of our problem differs significantly from that of (Savkin and Li, 2018). In our domain, vehicles are confined to the limited space of the junction area, determined by their size. Unlike the robot, vehicles have a

specific objective to avoid collisions with obstructions within the junction while navigating towards their intended destinations. Additionally, vehicles remain inside the junction for only a brief period before exiting. The context of our problem imposes several constraints that need to be incorporated and adds to the complexity of the collision avoidance task.

Next, we describe the traffic model used in this chapter, especially the obstacle model.

3.2 Traffic Model

Our model is built as a sophisticated multi-agent system comprising two main components: the IMA and the DAs, drawing inspiration from the state-of-the-art model presented in (Dresner and Stone, 2008). In this system, the DAs take on the role of driving the vehicles and engaging in communication with the IMA, while the IMA serves as the controller stationed at the junction, scheduling the access of the DAs. Moreover, as one of our key focuses is addressing obstructions, we have also defined a model of obstacles within our system.

In the next sections, we provide detailed models of obstacles, vehicles, lanes and junctions, which together form the foundation of our autonomous junction management system.

3.2.1 Obstacles

In our system, obstacles are defined as physical objects that hinder or block trajectories within the junction, directly impacting the movement of vehicles. We specifically focus on static obstructions that may be present either within the junction area or at the entrance to the junction. These obstacles can include structures like sinkholes and road constructions, which can pose challenges to smooth vehicle flow.

For the purpose of this development, we simplify the consideration of obstacles by making the following assumptions:

1. Free Shape Representation: Obstacles are represented as free shapes. In our model, circles are used to act as safe rings around the obstacles. These circles provide a clear indication of the area that vehicles have to manoeuvre around to ensure safety while crossing the junction.
2. Static Obstacles: Each obstacle remains stationary throughout the management process. Dynamic obstacles, such as moving pets or persons, are not included in this model. Their size and position are predefined and do not change during the simulation.

3. Limited Obstacles' Position: Obstacles are restricted to two possible locations – they can be present either:
   (a) within the junction area or
   (b) at the exit of inbound lanes.
4. IMA's Knowledge of Obstacles: The IMA has complete knowledge of the obstacles' positions and their corresponding safe rings. This information is readily available to the IMA, enabling it to convey details of the obstacles to all DAs.

For the definitive representation of the obstacles, let O represent an obstacle within our system. Each obstacle is characterised by specific properties, including its position pos_O, obstacle radius $r(O)$, safe ring radius $r_{safe}(O)$, the circle around the obstacle denoted as $C(O)$, and the circle of the safe ring denoted as $C_{safe}(O)$. We assume that the DAs within our model have full knowledge of the obtacle's $C_{safe}(O)$ as they enter the junction, which is essential to perform the obstruction avoidance task.

3.2.2 Vehicles

Let t be the current time step and $A_t = \{a_1...a_n\}$ be a set of driver agents in our system at time t. Each $a_i \in A_t$ is modelled with its own properties: position pos_i, velocity v_i, width w_i, length l_i, and the orientation θ_i. All agents in A_t have the same value of maximum velocity v_{max}, minimum gap between the agent and the leading agent and accelerating rate α. We assume that each agent also has a knowledge of its current lane. For the vehicles' properties, the actual values are provided in Section 3.6.1.1.

At each time step, pos_i will be updated based on the agent's velocity and orientation following the equations below:

$$\frac{\partial x}{\partial t} = v_i \cdot cos(\theta_i) \ , \ \frac{\partial y}{\partial t} = v_i \cdot sin(\theta_i)$$

where x and y are the horizontal and vertical positions of the agent. Specifically, these equations are used to specify the precise orientation fo the vehicles while crossing the junction, especially when the DAs turn left or right.

Furthermore, an essential element in modelling vehicles is their driving behaviour. Since we assume all vehicles in our system are CAVs, their driving behaviour differs from that of human-driven vehicles. All CAVs in our model follow the car-following model from Krauß et al. (1997), which is based on the concept of "Let vehicles drive as fast as possible while maintaining perfect safety[1]" without exceeding v_{max}.

[1] https://sumo.dlr.de/docs/Definition_of_Vehicles%2C_Vehicle_Types%2C_and_Routes.html#default_krauss_model_description

With this model, one crucial parameter that significantly affects driving behaviour is the reaction time. We set the reaction time of CAVs to 0.25 seconds. Although CAVs can have reaction times close to 0-0.1 seconds (Xie et al., 2019), we opted for 0.25 seconds to reduce computational load and accelerate the simulation time. This reaction time is still faster than SUMO's default human reaction time, which is set to one second. By utilising this model, we can accurately represent the responsiveness of CAVs in our simulations, demonstrating their advantages over human-driven vehicles, especially how they decelerate and maintain their safe headways. Despite the fast reaction time of CAVs, they cannot drive extremely close to each other, and the shortest headway of 2.5 m must be followed.

3.2.3 Lanes

We model the lanes as a simple representation of the structural position, length, width, stop bar, orientation or direction in radius, and flow type (inbound or outbound). Each lane carries a *line segment*, which is one or multiple line equations representing: the position, centre and direction of the lane. The usage of multiple line equations is to represent curving roads. The entry point and exit point are also defined for each lane. We also define a set of possible lanes for the vehicle to exit the junction. For instance, the possible lanes of the left-most lane are lanes that support agents moving forward and turning left, and vice versa for straight-going and right-turn ones. Next, the different lane settings are used to model the junction.

3.2.4 Junction

The junction is modelled with four different groups of incoming and outgoing lanes (North, West, East, and South), and a centre area in the junction is called *grid*. The trajectory of a vehicle crossing the junction is called a *path*. This *grid* is divided into a number of cells (κ), which will be used to detect the conflict of path in requesting reservation process in our model.

Our focus in this work revolves around two junction configurations. The first configuration is a replica of the junction presented in Dresner and Stone (2008), enabling us to create a traffic environment similar to the state-of-the-art approach for performance comparison. This configuration is depicted in Fig. 3.2.

Subsequently, we move beyond the ideally configured junction in the state-of-the-art approach and focus on more generic junctions. This junction is of particular interest as it closely resembles real-world scenarios found in urban areas like New Delhi, Bangkok, or Manhattan, where primary roads intersect with minor/secondary roads. In such cases, the presence of obstructions can significantly impact traffic flow, as the small

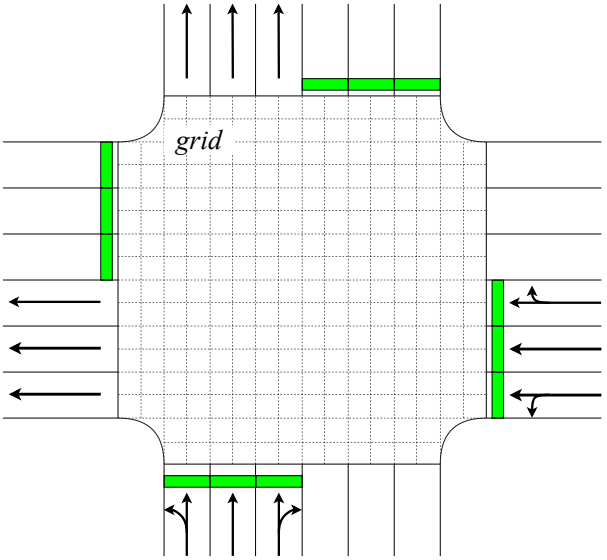

FIGURE 3.2: This illustrates a 3-lane junction model where the space in the middle of the junction is discretised into cells/grid. The arrows represent the directions of the traffic flow. This layout is modelled after the junction in (Dresner and Stone, 2008)

junction size may limit traffic capacity and weaken the traffic management capabilities. Consequently, even a single obstruction can considerably disrupt traffic, allowing us to examine the impact of obstructions in greater detail.

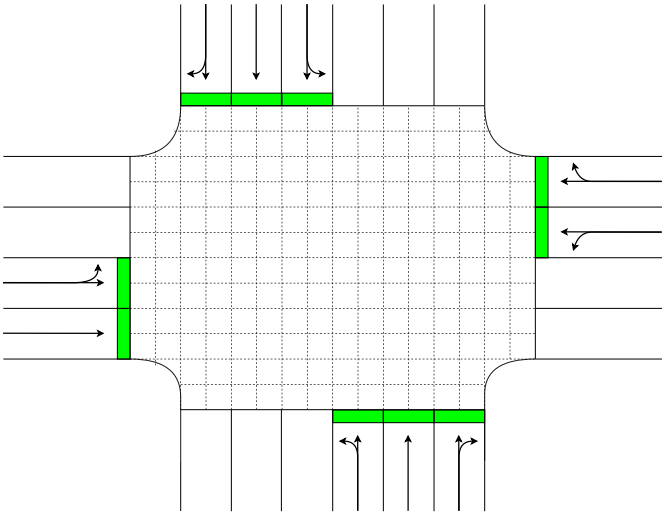

FIGURE 3.3: This figure illustrates the topology of a junction in Manhattan between Park Avenue South and East 23rd Street. The arrows represent the directions of the traffic.

As a result, we model our second junction configuration based on a real junction in Manhattan, specifically the intersection between Park Avenue South and East 23rd Street (Fig. 3.3). It is essential to emphasise that our method is not limited to these specific junction configurations. With slight adjustments, it can support various other junction layouts as well.

Additionally, an important element that affects the model of the junction is the obstacles. Obstacles within our consideration can be positioned either inside the junction area or at the exit of the inbound lanes. The interesting position of the obstacle is when it is located at the exit of the inbound lane. The vehicles in that lane necessarily change lanes to avoid unnecessary blockage before they begin crossing. Without preemptively anticipating the blockage and allowing the simulated vehicles to react on the spot, this will interrupt the traffic flow, resulting in huge travel delays.

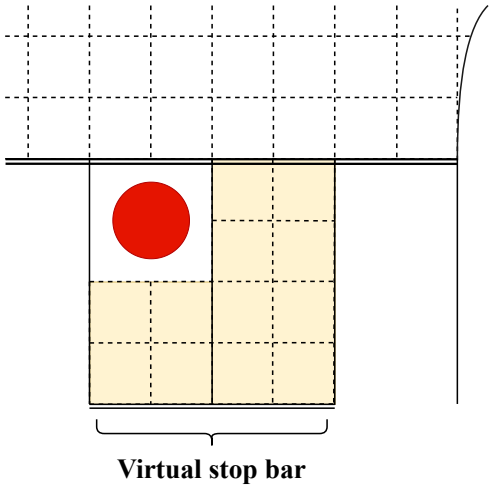

FIGURE 3.4: The extended area whenever an obstacle is positioned at the exit of the inbound lane. The circle represents the obstacle, and the highlighted area is where the grid is extended. The virtual stop bar is also located as depicted.

To address this, we propose a modification to our cell-based approach extending the junction grid to cover the area around the obstacle according to the position (see Fig. 3.4). The main benefit of this extended section is to have the lane-changing movements combined with the crossing movements, allowing those movements to be reserved as one. As a result, DAs can make a reservation immediately after arriving at the queuing area, potentially ignoring any delays that the on-spot lane-changing movements may cause. Note that, to aid the queuing, a virtual stop bar is defined in front of the extended grid.

Furthermore, to keep track of the space reservation, the junction agent also holds a reservation map in the form of a 3-D matrix that basically determines cells' occupancy. The first and second dimensions represent the grid of cells, while the third dimension contains a list of integers representing reserved timestamps per cell. For an implementation perspective, this reservation map is recorded in a dictionary structure[2] as shown below:

```
reservation_map = {cell_1 : [t+1, t+2, ...],
                   cell_2 : [t+1, t+2, ...],
                   cell_3 : [t+5, t+6, ...],
```

[2]A data structure that stores a collection of key-value pairs

```
cell_4 : [],
  .
  .
cell_κ : [t+31, t+32, ...]}
```

CODE 3.1: Data structure of reservation map

Note that the reservation map contains reserved time slots for each cell, but there is a possibility that cells have not been reserved yet, which is recorded as an empty list, "`[]`". Given this reservation map, $\texttt{cell}_1$ will be reserved at time slots of `[t+1, t+2, ...]`, and vice versa for $\texttt{cell}_2$ to $\texttt{cell}_\kappa$. The incrementation of the time value here can be decimal depending on the time step of the simulation.

By having this reservation map, the system can determine how long each cell will be reserved and schedule vehicle access accordingly. The size of the reservation map will decrease continuously as the simulation runs, and it will grow only when the IMA records a newly granted reservation. Note that the third dimension only grows up to the latest reserved timestamps.

Our model was designed as a multi-agent system similar to Dresner and Stone, but we improve the method by altering the interaction between the IMA and the DAs. The several tasks of the IMA are pushed to the DAs to reduce the computation workload of the IMA and minimise the message exchange. We start describing our algorithm with the behaviour of driver agents and also its responsible operations being pushed from the IMA.

3.3 Driver Agents Behaviour

This section describes actions and procedures that the DAs take in order to make a time-slotted reservation with the IMA. The algorithms beyond the communication with IMA are also explained later, as they are essential for the reservation mechanism.

The DAs in our model can accelerate freely as long as they satisfy the vehicles' constraints (car-following model), as in Section 3.2.2. To acquire a reservation, the DAs must execute several operations within the resource reservation mechanism, which are summarised as follows:

1. Receiving information on speed limits, stop line positions, the target lane (i.e., which is important to path prediction) and reservation map (i.e., which cells in the junction are already reserved) from the junction agent once they arrive at the communication range of IMA.

2. Approach the junction. Once the position is five metres from the stop line, continue the next step.
3. Initiating path prediction, which will be detailed in the next section.
4. Resolving the conflict that may arise against other DAs' reservations.
5. Sending a requesting message containing the vehicle properties and the predicted path to the IMA, and then wait for a confirmation.
    - If the request is rejected, go back to step (1).
    - If the request is confirmed, begin the crossing.
6. Notify the junction agent that they have left.

The choice of operations in our algorithm is driven by the aim to achieve the FCFS principle. Further detailed reasons behind these choices will be discussed in Section 3.5.

A key feature of decentralisation in our algorithm is the delegation of computations to the driver agents, specifically in operations (3) and (4) - path prediction and conflict resolution. By assigning these operations to the driver agents, we minimise the computational burden on the junction agent. The driver agents are responsible for determining a conflict-free path from their current position to the target lane. This redistribution of computation reduces the overall computation cost at the junction agent.

However, this decentralisation comes with the cost of additional message exchanges between the IMA and the DAs. In operation (1), the IMA broadcasts essential information to facilitate operations (3) and (4). This information exchange is crucial to avoid repetitive communications and computations whenever conflicts occur, which are prevalent in the original FCFS method. While there is an added communication cost, it is outweighed by the advantage of significantly less repetitive operations at both IMA and DAs.

Moreover, it is important to note that rejection in operation (5) serves the purpose of preventing potential conflicts arising from concurrency issues. By rejecting conflicting requests, we maintain the integrity and safety of the system's operation. This aspect will be discussed in detail in Section 3.4. More importantly, the DAs constantly follow the car-following model. Without approval or rejection from the IMA, the DAs continue to approach the junction as if there are red lights.

In the next section, we delve into the algorithms executed by the driver agents within our autonomous junction system. Specifically, we will discuss the processes of path prediction and conflict resolution, which are integral to the efficient and safe operation of the system.

3.3.1 Path prediction

Path prediction involves the driver agents' ability to calculate a path from their current position to the desired target lane. An essential component of the path prediction operation is the *lane following* function introduced in Dresner and Stone (2008). Specifically, DAs simulate their movements step by step from their position to the target lane, either turning or moving straight. It basically modifies the orientation of agents, θ_i, mimicking a steering mechanism while modifying position, pos_i, moving towards the assigned target lane. As a result, these modified θ_i and pos_i will affect the agents' positions as described in Section 3.2.2. While moving, the agents' orientations are kept changing towards the target lane's direction until orientations align. This function allows agents to predict their optimal path from their current position to the target lane. In case an agent $a_i \in A_t$ performs this path prediction, the result will be a predicted path:

$$p_i = \{< pos_i^{t+1}, \theta_i^{t+1} >, ..., < pos_i^{t+s}, \theta_i^{t+s} >\} \tag{3.1}$$

where $t + s$ is an end timestamp of this predicted path. To be specific, this predicted path contains a set of vectors that determines the position and orientation of a_i at each time step. Here, $t + 1$ refers to the next time step in the simulation, and the purpose of the value of 1 is only to simplify the equation. The actual simulation time-step can be different values depending on the settings.

3.3.2 Path Prediction with Obstructions

Here, we explain how the DAs can effectively perform path prediction even in the presence of obstructions. This capability allows the DAs to specify an obstruction-free path, which is then used in the conflict resolution operation to ensure non-conflict navigation through the junction.

Even though obstacles exist within the junction grid, the DAs need to detect when collision avoidance is necessary, as the obstacles may not always obstruct their paths. Hence, we introduce a fundamental function called "obstruction detection" to fulfil this requirement.

Obstruction Detection Function

Upon reaching the junction, the DAs execute the "lane_following()" function to obtain an initially-predicted path obtaining p_i (see Eq. 3.1). The obstruction is detected if the distance between one of the positions in the predicted path and the centre of the

obstacle's circle is less than $r_{safe}(O)$ (the safe ring radius around the obstacle). Mathematically, this condition is represented as follows:

$$D(x) \text{is a distance function between } pos_i^x \text{ and } pos_O \tag{3.2}$$

$$\exists(x \in \{t+1, ..., t+s\})[D(x) < r_{safe}(O)] \tag{3.3}$$

An example of this situation is illustrated in Fig. 3.5. This function is later referred to as $obstruction\_detection()$, which returns the blocking obstacle as an object.

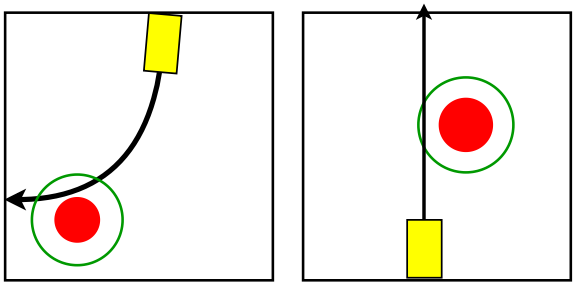

FIGURE 3.5: An example of a blocked or obstructed path, where the red circles represent the actual obstacles, and the green circles represent the safe area/rings around the obstacles.

Once a potential obstruction is detected using this function, the initially-predicted path is discarded, and the DAs initiate the path prediction process using a collision avoidance algorithm instead. This algorithm determines a new path for DAs, ensuring they deviate from their normal trajectories to avoid colliding with the obstacles.

Collision Avoidance Algorithm

In this section, we present our algorithm for enabling the DAs to navigate to the target lane while avoiding collisions. Our approach is based on the model proposed by Savkin and Li (2018), where a unicycle robot explores an environment with obstacles. However, we have modified their algorithm to suit road traffic environments where space is limited, movement is regulated, and the exit point is fixed at the target lane.

The obstructed-free path of each DA can be broken down into three main stages:

1. Avoiding Blockage: The DA follows the tangent line between its entry point and the safe ring of obstacles[3] $C_{safe}(O)$ to avoid collisions.
2. Following the Safe Ring: The DA moves along the $C_{safe}(O)$ curve until the obstruction is no longer detected (using the $obstruction\_detection()$).
3. Moving to the Target Lane: The DA steers and moves forward from the end position in stage (2) towards the entry of the target lane.

[3]The size of $C_{safe}(O)$ is adaptable to different vehicles' width.

Stage (3) serves as a recovery mechanism, ensuring that the agent returns to the initial aiming direction while utilising the least possible space. Fig. 3.6 provides an example of these movements.

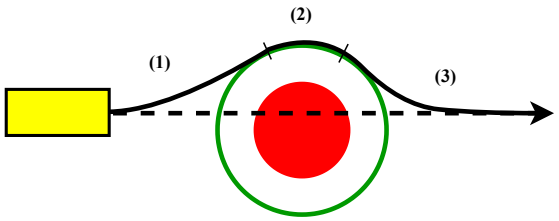

FIGURE 3.6: The predicted path considers the obstacle, where the solid line represents the path from the collision avoidance algorithm while the dashed line represents the normal predicted path.

With this collision avoidance algorithm, our driver agents can predict a path from their current position to the target lane, either with or without obstructions. The path is recorded in vector form, similar to Eq. 3.1. Subsequently, the DAs proceed to perform conflict resolution to determine conflict-free time slots, allowing them to send reservation requests to the IMA.

3.3.3 Conflict Resolution

In this section, we outline the process of conflict resolution within our system. Conflict resolution plays a role in ensuring the safe movement of vehicles by detecting and addressing potential conflicts that may arise against other/previous reservations.

Even though the predicted path is already determined, the DAs cannot begin junction crossing immediately. When multiple DAs navigate the junction simultaneously, there is a high possibility of their paths overlapping, leading to potential conflicts (see Fig. 3.7). To address this, we have implemented a conflict resolution algorithm.

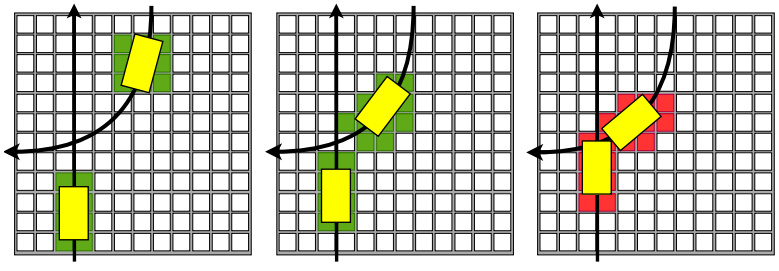

FIGURE 3.7: Usage of discretised cells for detecting conflicts between two paths. The green-fill or red-filled rectangle specifies the desired cells of the individual vehicle. These images represent a continuous movement of two vehicles, and it is evident that the rightmost image shows the overlap between the desired cells of two different vehicles.

First, the conflict resolution starts with each DA reconstructing its vehicle's body per time step $h \in \{t+1, ..., t+s\}$. Using information such as length l_i, width w_i, and movements from the predicted path p_i (pos_i^h & θ_i^h). The DA projects the vehicle's body onto the junction grid to identify the cells it intends to use at each time step. We record these cells in a variable called "desired cells" or "to-be-reserved cells", which is stored in a dictionary structure as shown in Code 3.2.

```
desired_cells_i = {cell_1 : [t+1, t+2, ...],
                   cell_2 : [t+1, t+2, ...],
                   cell_10 : [t+5, t+6, ...],
                   cell_11 : [t+5, t+6, ...],
                      .
                      .
                   cell_73 : [t+31, t+32, ..., t+s]}
```

CODE 3.2: Data structure of desired cells. Note that the cell numbers here are used for explanation purposes only.

Specifically, this record structure indicates that agent a_i requires $\texttt{cell}_1$ at time slots `[t+1, t+2, ...]` and so on. Note that the total number of cells on the grid is κ, which means the set of all possible cells is $\{\texttt{cell}_1, \texttt{cell}_2, \texttt{cell}_3, ..., \texttt{cell}_\kappa\}$. However, the desired cells variable does not record all the possible cells in the grid; only the cells that the DAs required are recorded. This is to save up the memory storage and computation load.

To identify potential conflicts, the reservation map is used to compare against the DAs' desired cells. A conflict is detected if any cell from the desired cells has a time slot that overlaps with the reservation map. To resolve this, a current operation is to have DAs wait for a certain time: we refer to this time value as "waiting time". The calculation of waiting time can be seen in Code 3.3.

```
1  current_time = t
2  conflict_time = 0
3  for k in keys(desired_cells_i):
4      intersected_time = max(desired_cells_i[k] ∩
5                             reservation_map[k])
6      conflict_time = max(conflict_time, intersected_time)
7  waiting_time = conflict_time - current_time
```

CODE 3.3: Calculation of the waiting time

The `keys()` function returns all the keys within the dictionary structure. In our case, $\texttt{keys(desired\_cells}_i\texttt{)}$ returns all the cells that agent a_i needs given path p_i. Then, a

loop is used running through all the desired cells to determine potential conflicts. In line 4, `intersected_time` denotes the latest time slot where a desired cell intersects with a previously reserved cell from the reservation map. The actual conflict time is then determined by taking the maximum value of `intersected_time` among all desired cells. This waiting time guides the DAs on how further they should wait before proceeding. Therefore, to calculate the waiting time, we subtract the current timestep of the simulation from the conflict time. We refer to this function as $conflict\_resulution(p_i, a_i)$ returning a value of waiting time.

Particularly, this waiting time is used to shift the timestamp of the predicted path, slowing down the DAs' trajectory/access to the junction to avoid conflicts. Assuming that the calculated waiting time is denoted as wt, the updated predicted path is denoted as $\hat{p}_i$ and is given by:

$$\hat{p}_i = \{< pos_i^{t+wt+1}, \theta_i^{t+wt+1} >, ..., < pos_i^{t+wt+s}, \theta_i^{t+wt+s} >\} \tag{3.4}$$

However, this adjustment might lead to new conflicts, so the entire conflict resolution process, including the shifting of timestamps, is repeated until the predicted path is free from any conflicts. In that case, the predicted path p_i is replaced with the shifted one $\hat{p}_i$ every time the conflict resolution operation is carried out. Simply put, the waiting time keeps accumulating until no conflicts remain.

One key aspect of our approach is that the DAs themselves are responsible for performing conflict checking/resolution, rather than relying on the IMA. This empowers the DAs to ensure that their predicted paths do not lead to any conflicts before submitting the reservation request, considering the broadcasted reservation map in hand. Consequently, this conflict resolution strategy significantly reduces the computation load on the IMA and minimises the number of rejected messages compared to the original work by (Dresner and Stone, 2008).

With the conflict-free predicted path, DAs continue their operation in step (5), sending a requesting message containing the predicted path and desired cells to the IMA. The path included in the message in the requesting message is highly likely to be conflict-free. Our decentralised approach simplifies the tasks of the IMA, reducing them to two actions: verifying the request and sending a confirmation. The next section will provide a detailed explanation of the junction agent's behaviour.

3.4 Junction Manager Agent Behaviour

In this section, we explore the behaviour and functionality of the IMA within our autonomous junction system. The IMA is responsible for coordinating the actions of the driver agents and ensuring the smooth and safe operation of the junction.

Firstly, the operation of the IMA commences by broadcasting messages to any DAs that enter its communication range. These messages contain essential information for DAs' path prediction and conflict resolution, including speed limits, stop line positions, possible target lanes, and the reservation map. Unlike the original FCFS approach, the junction agent is no longer responsible for predicting the path of requesting agents. This prediction is now included in the requesting message itself, referred to as the *requested path*.

Upon receiving the requested path, even though this path has gone through conflict resolution on the DAs' side, the IMA still validates its conflict-free compatibility before providing an approval or rejection response to the requesting agents. The necessity and operations behind this validation will be explained in Section 3.4.1. Later, the IMA expects to receive completion notifications from the DAs after granting access. This is to ensure success in scheduling the access of each DA and also closing up the interaction cycle between IMA and certain DAs. Overall, the interaction flow between the driver agents and the junction agent in our algorithm is shown in Fig. 3.8.

Next, we explain the decisions behind designing the junction manager agent's behaviour. Mostly, these decisions are primarily influenced by the challenges associated with relying on wireless communication within the system, namely concurrency issues and indefinite message waiting.

3.4.1 Concurrency Issues

As we have transferred the conflict resolution responsibility to the DAs, the need for confirmation messages, which were essential in traditional FCFS approaches, may no longer be necessary. This is because the predicted paths requested by the DAs are now more likely to be conflict-free, as the DAs have already resolved any potential conflicts on their end. However, it is important to note that while the DAs may perceive their paths as conflict-free, the IMA, from its perspective, may still identify potential conflicts that were not accounted for by the DAs.

With multiple agents (in this case, the DAs) accessing and utilising shared data, which is the reservation map held by the central unit (the IMA), inconsistency can occur when the data is provided to multiple agents simultaneously. During the resource reservation mechanism, to indicate the current state of the junction's reservations, the reservation map is updated instantaneously. However, due to message exchange delays, there is a chance that the DAs receive different versions of the reservation map, leading to data inconsistency and synchronisation problems. This concurrency issue can result in incorrect, undesired outcomes and vehicle crashes.

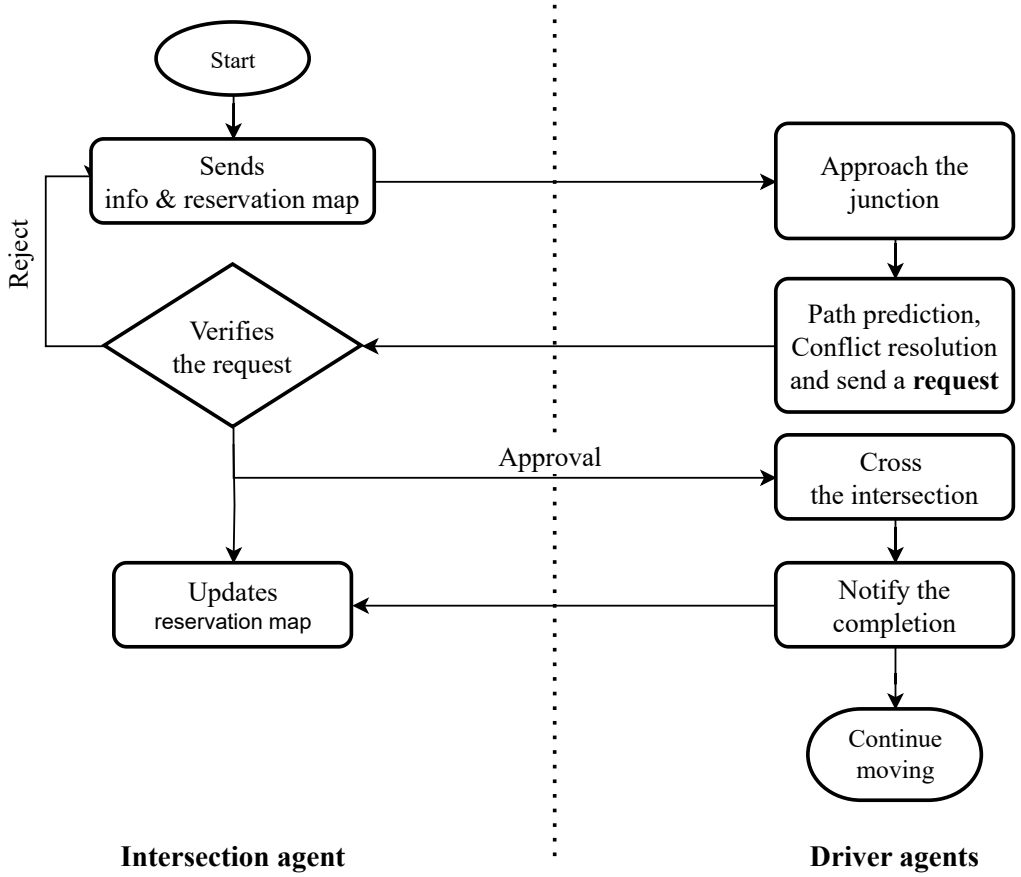

FIGURE 3.8: This flowchart illustrates the overall interaction between intersection agent (junction agent) and driver agents.

To address the concurrency issue and ensure safety and synchronisation, a permission-granting step becomes necessary. This step ensures that the DAs perform their operations based on the correct and consistent information from the IMA before proceeding with their actions. Therefore, the system can maintain the consistency of the shared data and safety among the vehicles.

In particular, in the permission-granting step, the IMA verify the validity of the requested path against the current state of the reservation map. The IMA simulates the movements of the requesting agent, retrieving the `desired_cells`$_i$ and the previously granted agents based on their timestamps while double-checking for any conflicts. Given the data structure of the reservation map in Code 3.2.4 and the desired cells in Code 3.2, the double-checking can be done by using instructions only lines 3 to 5 in Code 3.3. As this process is done on the IMA side, it utilises the latest reservation map held by the IMA.

However, potential conflicts may arise if a requesting message from agent a_{i+1} arrives after the reservation map has already been updated by accepting a request from agent

a_i. If a conflict is found, the validation process stops, and this reservation request becomes invalid. In such cases, the request from agent a_{i+1} will be rejected, and the agent will need to initiate the resource reservation mechanism from the beginning.

On the other hand, if no conflicts are detected, the IMA sends an approval message back to the requesting agent, which means that the cells specified in $\texttt{desired\_cells}_i$ are now reserved for the requesting agent. The reservation map is updated accordingly.

To demonstrate how the reservation map records, assuming that the reservation map is initially empty and the desired cells are as stated in Code 3.2, the updated reservation map ($\texttt{reservation\_map}'$) will be as follows:

```
reservation_map' = {cell_1 : [t+1, t+2, ...],
                    cell_2 : [t+1, t+2, ...],
                    cell_3 : [],
                       .
                       .
                    cell_9 : [],
                    cell_10 : [t+5, t+6, ...],
                    cell_11 : [t+5, t+6, ...],
                    cell_12 : [],
                      .
                      .
                    cell_72 : [],
                    cell_73 : [t+31, t+32, ..., t+s],
                      .
                      .
                    cell_κ : []}
```

CODE 3.4: Updated reservation map

3.4.2 Indefinite Message Waiting

Moreover, it is crucial to acknowledge another issue that can arise with this design, specifically the problem of indefinite message waiting. This problem can occur at two different timings: (1) when DAs send a request message to the IMA and wait for a response, and (2) when the IMA waits for DAs to provide a notification of the completion of their journey.

Firstly, let us examine the causes and impacts of situation (1) on the system. In some cases, a request message may be interfered with and not reach the IMA (e.g., electromagnetic interference or physical obstacles). Consequently, the IMA cannot process or serve the request and send a response back to the DA. Currently, the communication

design between the IMA and DAs lacks a fail-safe mechanism to address this issue, meaning both agents remain unaware of the missing message.

Furthermore, we are going to detail the impact of this situation. If DAs' reservation requests do not receive approval from the IMA, the DAs are uncertain about the safety of their requested time slots to proceed. While conflict resolution techniques may use a reservation map to identify conflict-free time slots, it cannot be assumed that it is always safe to proceed without explicit approval. Due to the missing approval message, the affected DAs cannot move from their current positions to cross the junction. Consequently, any DAs positioned behind the affected vehicle will also face indefinite waiting. Although other DAs may be able to perform strategic lane changes to access lower queuing lanes, access to the junction itself becomes limited (one less inbound lane). This impact is reflected in the system's junction throughput and traffic capacity, which are significantly reduced. In the worst-case scenarios where multiple vehicles are waiting for messages indefinitely, a gridlock is going to happen.

Secondly, in situation (2), if the IMA does not receive the completion notification from DAs, the interaction cycle will be incomplete. Without receiving this notification, the IMA does not know whether the requesting DAs have left the junction or completed their journey or not. The IMA only performs broadcasting and acknowledges the DAs existence only when receiving their request. To maintain good synchronisation between the IMA and DAs, we must ensure that IMA will not indefinitely wait for any DAs' completion notification.

To address the issue of indefinite message waiting, we propose implementing a timeout mechanism. We assume the slowest vehicle's speed equals the average walking speed, approximately 1.2 m/s. The timeout duration can be calculated based on the longest trajectory within the junction divided by the walking speed. For instance, if the longest trajectory is 20 meters, the calculated timeout would be $20/1.2 \approx 17$ (16.66) seconds. This timeout duration represents the maximum time for the slowest vehicle to cross the junction. After providing the DAs with approval, the IMA will not wait for notification of completion from the DAs for longer than the specified timeout.

In the event of no completion notification received from DAs, after the specified timeout period (e.g., 17 seconds), the IMA will no longer expect any further completion notifications from the DA that failed to send the notification. This ensures that the IMA can drop any loads and proceed without waiting indefinitely for missing notifications. It is impractical to keep waiting for a message from DAs that have already left the junction and could be outside the IMA's communication range. Moreover, there could be a chance that DAs have not received the approval in the first place, so the notification of completion will never be sent.

In case the DAs have not received the approval, they cannot proceed, while the IMA believes that the DAs are crossing the junction. To prevent repetitive requests from

being sent to the IMA while the IMA waits for notification of completion, the timeout at the DAs is necessary. In a similar manner, the DAs will also have the same timeout period to wait for approval or rejection from the IMA. This allows DAs to stop waiting at the same time when the IMA closes its expectation for the notification of completion.

Particularly, if there is no feedback (approval or rejection) within a specified time (e.g., 17 seconds), the DAs will send a request to the IMA again. The consistent timeout on both the IMA and DAs prevents repetitive requests from being sent to the IMA, which would otherwise consume unnecessary communication resources and computation loads.

Nevertheless, it is essential to note that each DA's timeout is registered or recorded for their specific reservation and operates in parallel. Dedicated computation units or threads are generated to handle the timeout for the individual requesting agents. This approach allows the autonomous junction control to operate normally without being interrupted by the timeout for reservation reject/approval and completion notifications. By incorporating these timeout mechanisms, the system ensures timely and efficient communication between the IMA and DAs, resolving indefinite messages waiting on both ends within a reasonable timeframe. As a result, the throughput of the junction and traffic capacity can remain stable.

In the next section, we are going to explain how we achieve the first-come-first-serve principle within our approach, which is rather different from how it was traditionally implemented in the state of the art.

3.5 First-Come-First-Serve Principle

After defining all the operations and functions for both DAs and IMA, this section delves into the modeling of the First-Come-First-Serve principle, namely FCFS. In our simulation, IMA broadcasts essential information such as speed limits, stop bar positions, obstacle's position & safe ring, and the granularity of cell-based discretisation, as well as the reservation map to all DAs in the area (refer to Fig. 3.8). Once DAs enter the broadcasting range of IMA, they acknowledge the presence of IMA preparing operations for autonomous junction. While the interaction between IMA and DAs has been explained earlier, implementing the FCFS principle in our model is not a straightforward task.

Specifically, being the first to enter the communication range of IMA does not guarantee being the first to access the junction. To design the FCFS principle effectively, we must consider several issues that can impact practicality and traffic performance. These issues include (1) the uneven distribution of road space, (2) and serving order. The following sections will provide detailed explanations of the causes and effects of

these issues and how they are addressed to ensure a reliable implementation of the FCFS principle.

3.5.1 Uneven Distribution of Road Space

The issue of uneven distribution of road space arises when vehicles queueing on multiple lanes of the same road experience uneven growth. This problem can occur with the original method proposed by Dresner and Stone (Dresner and Stone, 2008). Their FCFS approach operates by granting reservations to DAs in sequential order as they enter the communication range of the IMA. Once DAs receive a reservation, they remain in their designated lanes without the ability to change lanes until accessing the junction.

By operating in this way, no issues are evident when using their environment and demand settings, which is a 4-way, 5-lane junction with low traffic input. In their simulation, the road's capacity is significantly large, highly likely exceeding the actual traffic demand. The absence of high demand reduces the number of conflicts between DAs reservations, resulting in shorter waiting times. With fewer conflicts, the vehicles can simultaneously proceed through the junction, which results in short vehicle queues. Here, the lane-changing capability is rather not essential.

However, in situations with extremely high traffic demand, the problem of uneven distribution of road space emerges due to differences in release speed or throughput among the lanes. The number of conflicts between DAs' reservations is the main contributing factor to this discrepancy, as more conflicts lead to longer waiting times. This means that at least one lane is going to have vehicles queueing longer than the rest. An example is that turning lanes are likely to have a lower throughput than the straight-going lanes due to slow turning speed (for safety purposes) and conflicts with the opposite side of the road.

The lack of flexibility in lane changing can lead to excessively long queues forming in a single lane, potentially blocking the entry of newly generated vehicles. Furthermore, in scenarios involving multiple junctions, the queueing can overflow from the outbound lanes of nearby junctions, also known as a spillover issue. Specifically, this issue reduces the number of possible outbound lanes leading to the throughput of these nearby junctions being compromised. As a result, the issue of uneven distribution of road space not only impacts the specific junction where the long queue forms but also has consequences for nearby junctions, especially in terms of traffic throughput aspect.

Therefore, it is crucial to address the issue of uneven distribution of road space not only for the affected junction but also to ensure the smooth functioning of the surrounding junctions and maintain optimal traffic flow in case of an expansion of the study scope. More importantly, we must ensure fair and efficient utilisation of the available road space.

To do so, we allow some freedom for DAs to make a decision according to their positioning while approaching the junction. In our model, DAs do not immediately send a reservation request upon entering the IMA's communication range. This intentional design allows DAs to take advantage of the available road space by adjusting their speeds and lane positioning. One common action DAs takes is changing lanes and filling up short queuing lanes, thereby optimising the utilisation of the road space. By making these strategic adjustments, the DAs contribute to a more efficient traffic flow and help alleviate congestion.

It is important to note that in the customised simulation, DAs do not have the ability to change lanes. Instead, we assume that DAs enter the simulated environment corresponding to their intended junction access trajectory. For example, DAs always enter a straight-going lane to cross the junction straight, and vice versa for turning DAs. This assumption poses no issues as the traffic demand remains low in the customised simulation cases. However, in the SUMO simulation environment, each DA follows a strategic lane-changing model, enabling them to accelerate or decelerate while changing lanes and balance the queue among the lanes effectively. This lane-changing model is specified in (Krajzewicz et al., 2002).

After allowing DAs to adjust their lane positioning, DAs are programmed to send a reservation request once they are approximately five meters from the stop line, corresponding to the front bumper position. This positioning awareness is facilitated by the essential information provided in the IMA's broadcast message. We refer to this specific range as the "waiting area". However, it is important to note that, due to the flexibility of lane changes and the inconsistent throughput among different lanes, DAs may no longer approach the junction in the same sequence as they entered the IMA's communication range. This adjustment sacrifices the ability to serve the DAs' requests sequentially to the arrival order. Therefore, our approach redesigns how the IMA handles the serving order or sequence of the DA reservations.

3.5.2 Serving Order

This section is going to explain how we view and handle the serving order within our approach. We first explain the drawbacks of serving the request using the arrival order and then proceed to detail our solution to address them.

Strictly maintaining the serving order to the communication range arrival presents some drawbacks that can potentially hinder the system's performance instead of improving it. To illustrate this point, let us consider an example where Agent A enters the IMA's communication range before Agent B. Agent A intends to make a right turn and slowly follow the queue at the junction. This lane typically has a lower throughput due to the slower turning movements required for safety purposes. On the other hand,

Agent B is positioned on the opposite side of the road, has arrived in the waiting area and wishes to go straight, but its crossing path and time slots overlap with Agent A's turning path.

If the system strictly adheres to the communication range arrival order, Agent B would be unable to cross the junction despite arriving at the waiting area first. This situation explicitly creates a bottleneck in the system, as the higher throughput of the straight-going lanes is compromised by the lower throughput of the turning lanes. This issue can be considered counterproductive and directly impacts traffic performance. Furthermore, Agent B would lose its fastest and conflict-free opportunity to cross the junction, if it follows the arrival order rule. By only following this rule, the autonomous junction would be unable to reach its full potential despite advanced communication capabilities between the DAs and IMA.

Due to the potential bottlenecks that arise when strictly following the communication range arrival order, we recognise the necessity for a change in our approach. To address these issues and maximise system performance, we have redefined the concept of "Come" in the context of the First-Come-First-Serve principle. Instead of using the communication range entry as arrival order, we now define "Come" as the point when the DAs send reservation requests after arriving at the waiting area. Essentially, we translate FCFS as "the first to arrive at the waiting area is the first to be served." This modification allows for a more practical and efficient implementation of the FCFS principle in the autonomous junction.

Under this new interpretation, the system treats the earliest reservation request that arrives in the waiting area as the first comer, ensuring that it is served first. For instance, even if Agent A entered the communication range of the IMA before Agent B, if Agent B manages to reach the waiting area before Agent A, it will be Agent B who sends the request to the IMA ahead of Agent A. By doing this, we ensure that the slower throughput lanes do not hold back the vehicles in the high throughput lanes. This system aligns with the order of arrival at the waiting area, effectively mitigating potential bottlenecks from varying lanes throughputs. We successfully alleviate the bottleneck challenge while retaining the fundamental concept of FCFS.

3.6 Empirical Evaluation

This chapter serves as a comprehensive evaluation of the proposed autonomous junction system, aiming to assess its performance and effectiveness. Through this evaluation, our objective is to gain a deeper understanding of the system's capabilities, identify its strengths and weaknesses, and make informed decisions for further improvements.

Our approach is assessed through two primary evaluation phases: a customised simulation evaluation and an evaluation within the SUMO framework. The initial phase represents a foundational assessment of our approach to investigate the pros and cons compared with the state of the art (not yet taking obstacles into account). Given that the IMA's computations are initiated by incoming messages, we measure the amount of message exchange between our approach and the traditional approach. With this, we can specify the computational workload between these two approaches.

Subsequently, our evaluation with SUMO places a special emphasis on traffic scenarios involving obstructions. This phase focuses on simulating different obstruction avoidance strategies, followed by performance analysis. To quantify the effectiveness, a pivotal metric in this assessment is queuing length, an indicator of how swiftly the junction can accommodate and manage traffic under various obstruction scenarios.

We next proceed to describe the evaluation with customised simulation.

3.6.1 Evaluation with Customised Simulation

Specifically, in this phase, we compare our computationally-decentralised approach with the centralised counterpart (FCFS) using a customised simulation tool that we have developed.

It is important to note that this study represents our initial exploration into autonomous junction management. As such, we have made some modifications to the seminal state-of-the-art approach, particularly in terms of agent interactions. Instead of utilising the open-source AIM simulation by Dresner and Stone (2008), we opted to create our own customised simulation. This decision was driven by the desire for greater programming flexibility and implementation adaptability. By developing our own simulation framework, we were not only able to adjust traffic inputs and junction configurations but also tailor the core algorithm to align with our approach. The primary modifications of the algorithm lie in the interactions between agents and the responsibilities of the resource reservation operations. Attempting to understand and modify the coding, especially within the core algorithm of the AIM simulation codebase would have been impractical.

We design the simulation after the AIM simulation proposed by (Dresner, 2006) incorporating four primary components: vehicles, lanes, and the junction, as initially defined in Section 3.2 (obstacles are not included yet). Particularly, this customised simulation is capable of generating the traffic situation of a 4-way junction with changeable variables, e.g., number of lanes, size of lanes, maximum vehicles' velocity, acceleration rate, vehicle input rate, vehicle properties, and state of traffic lights (red and green). Note that the state of traffic lights supports FCFS algorithm, not ours. In the original FCFS, red and green lights are used as release indicators. Moreover, the simulation

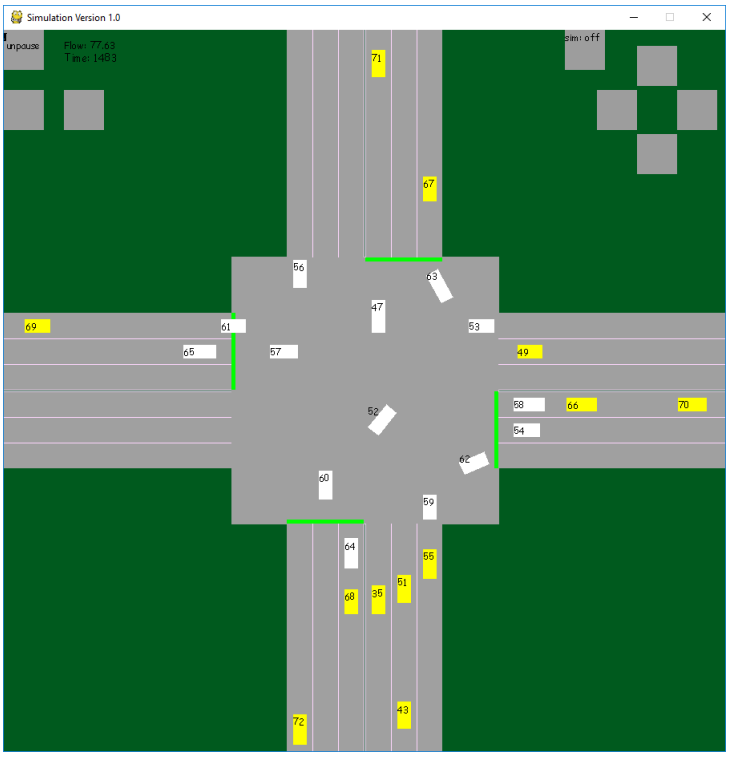

FIGURE 3.9: This is the screenshot of our customised simulation. A square button on the top-left is a pause button, and next to that button is the flow rate of the system. Rectangle boxes represent vehicles, and the white ones are the vehicles with granted reservations.

can also output traffic flow in either real time or a generated csv file. The flow rate is measured in a unit of *vehicles per minute* (vpm). A screenshot of our simulation can be seen in Fig. 3.9. We chose the measurement unit to be *vpm* because the duration of the simulation is quite short. However, this short-period evaluation is sufficient to record the difference in performance between FCFS and our decentralised method.

3.6.1.1 Experiment Settings

To provide a fair comparison between our approach and the centralised one, we aligned our settings with those presented in the seminal work by (Dresner and Stone, 2008).

In particular, we define a 3-lane junction and the width of each lane is 3.2 metres. The vehicles are 1.9 m wide, and $\in [3,5]$ m long. The speed is limited to $v_{max} \in \{12,18\}$ m/s and the acceleration rate is $\alpha = 3\ m/s^2$. The minimum gap between vehicles is 2.5 m. The input flow rate is 160 vpm, which generates 0.66 vehicles per second. The lane is chosen randomly between the three lanes. Vehicles have an equally 33.3% chance to continue straight, and turn left or right. The time step for simulation is 0.25 seconds. This means the predicted path contains four instances with one second of travel. To comprehensively evaluate the system's performance, the simulation was executed with

the aforementioned settings over the course of 10 individual runs, each spanning 10 minutes.

3.6.1.2 Experiment Results

The experiment showed that our decentralised approach has a similar performance as the original FCFS in terms of maximum flow rate, approximately 155 vpm. This customised is an observation tool for the researchers to investigate the behaviour of the DAs in action, which can ensure that the system operates correctly as intended. These exercises ensure that our implementation does not cause any undesired anomalies such as vehicular omissions, collisions, or vehicles caught in indefinite waiting loops. Particularly, the impact of such issues would be reflected in a drop flow rate performance, as these concerns directly influence the rate of vehicles entering the junction. As a result, the flow rate performance over time can be seen in Fig. 3.10.

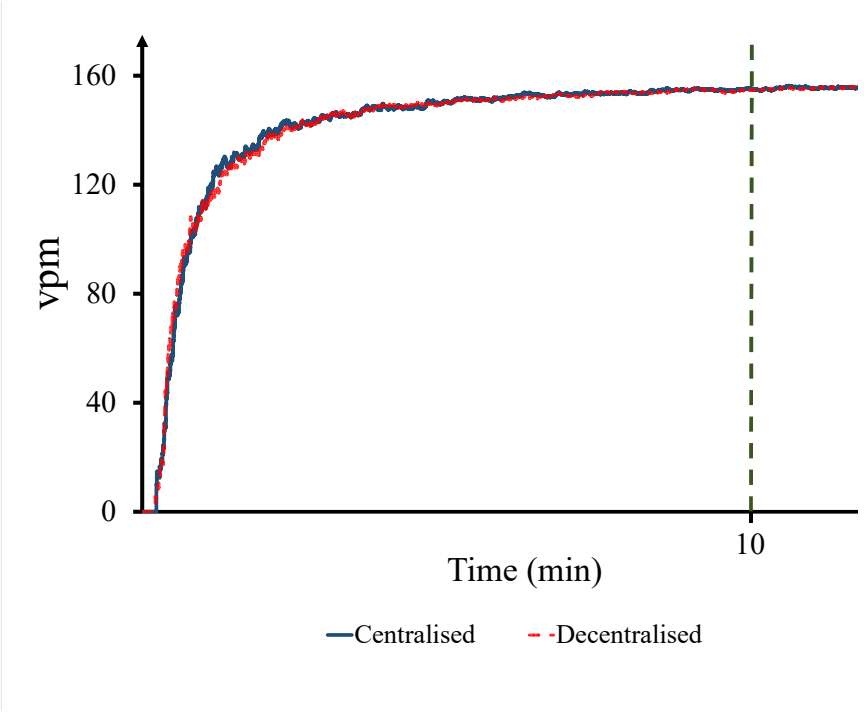

FIGURE 3.10: A line graph shows flow rate performance between the centralised state of the art and our decentralised model

Subsequently, we move to measure the number of exchange messages between two methods: FCFS and our method. The number or frequency of exchange messages serves as a reliable indicator of the computational load associated with each method. By quantifying the number of interaction cycles facilitated through these message exchanges, we gain insight into the computational workload of the agents within the system. An interaction cycle denotes the completion of a full resource reservation mechanism loop, thus allowing us to determine the agents' computational engagement.

To examine this aspect, we have extended the simulation duration to one hour, as 10-minute simulations cannot sufficiently highlight significant differences in the number of exchange messages between the two methods. Through a recording process across ten simulation runs, we obtained the average number of exchange messages under three distinct traffic volumes: 1,500, 3,000, and 6,000 vehicles per hour. The average number of exchange messages across ten runs is shown in Table 3.1. We next provide a detailed analysis of the results, focusing on the relation of the number of exchange messages to the computational workload of such autonomous methods.

Number of vehicles (veh/hr)	**FCFS**	**Our method**
1,500	5,260	6,012
3,000	13,362	12,011
6,000	41,480	24,080

TABLE 3.1: The average number of exchange messages in comparison between FCFS and our comutationally-decentralised method, under different numbers of input vehicles.

To begin with, from the results, it can be seen that there is no complete superiority from any of the two methods. A notable advantage of the seminal FCFS approach is its requirement for a smaller number of exchange messages to complete a resource reservation mechanism cycle (three messages) On the other hand, our method requires another broadcast message at the IMA side that makes a total of four messages in order to complete one cycle of the resource reservation mechanism. In low-traffic scenarios, wherein vehicular density attains a rate of 1,500 veh/hr, our method causes an increment in the average number of exchange messages relative to FCFS, approximating a differential of 14%. Looking at these prompt results, it appears that our method might not be the optimal option for operating under such low-traffic situations due to the dependency on the high number of messages.

Furthermore, we shift our focus to the scenario of 3,000 veh/hr. In this scenario, the state-of-the-art FCFS cannot maintain its superiority over our method as the count of exchange messages turns out to be higher. Despite having fewer exchange messages per reservation cycle, the FCFS seems to require more messages than our method. The key reason these results appear this way is the conflicts among reservation requests. By using a typical FCFS algorithm, the DAs must repeatedly submit new requests every time that they get rejected by the IMA. This causes multiple and identical requests to be sent until acquiring only a single approval. Note that within these experiments, we limit the number of repetitive requests sent to only one message per conflict that occurs, reducing unreasonable and unnecessary burdens on the computational hardware and also speeding up the simulation. Additionally, a single DA can submit more than two requests to acquire approval, as avoiding one conflict might lead to another. Theoretically, without this constraint, the actual number of messages will be extremely bigger. Nevertheless, even with the constrained results, the FCFS continues to use a

greater number of messages than our method, implying that the FCFS's total computation workload in this scenario is rather heavier than our proposed method. Numerically, the average number of exchange messages of FCFS exceeds that of our method by $\approx 10\%$.

In the same vein, in this 6,000 veh/hr scenario, the results show that the number of exchange messages with FCFS method is significantly higher than our method. Given this amount of traffic input, the system was experiencing a large number of potential conflicts among reservations, particularly more amplified than in the previous scenario. Essentially, this scenario has more chaotic traffic demand mimicking a junction in urban areas. As a result, the average number of exchange messages of FCFS climbs to $\approx$ 41,000, while our method only utilises $\approx$ 24,000 messages. Specifically, the FCFS used messages approximately 41% higher than our method.

Unlike the initial results from the 3,000 veh/hr scenario, here, the cost of one additional message incorporated in our proposed method shows to be beneficial. It can alleviate the impact of potential conflicts in terms of the number of exchange messages bypassing the repetitive exchange messages that happen in the original FCFS.

Next, we proceed to discuss and analyse the results comprehensively. The results of the number of exchange messages will be translated and interpreted into the amount of computation workload on both the IMA and DAs sides. This is to answer the main objective of this work – the reduction of computational load for the autonomous junction mechanism's central controller.

3.6.1.3 Computational Load Analysis

With the average number of exchange messages for both the FCFS and our method acquired, the subsequent step involves quantifying the precise computational load borne by both the IMA and DAs. This investigation is particularly significant given that the essence of the resource reservation mechanism centres around path prediction and conflict checking/resolution operations. Thus, our focus here is shifted towards quantifying these operations to uncover the underlying computational discrepancies between the two methods.

Firstly, we begin by quantifying the computational requirements for the path prediction operation. This operation's computation relates to the number of iterations needed to pinpoint and specify the positions and orientations of the DAs as they navigate the junction until they reach their designated target lanes. Key factors influencing this determination include the distance of travel and the velocity of the vehicles.

Consider a scenario accompanying this explanation: an agent a_i embarks on a straight journey from one side of the junction to the opposite side at a constant speed denoted as

v_i. In this context, let us denote w_{cell} as the width of individual cells, and k_{a_i} as the count of cells intersected by a_i's path (considered it as a line segment). With these defined parameters, the travel distance amounts to $w_{cell} \cdot k_{a_i}$, requiring $(w_{cell} \cdot k_{a_i}) / v_i$ iterations to complete the entire path prediction operation. Consequently, the computational cost or load inherent in the path prediction operation adheres to an $\mathcal{O}((w_{cell} \cdot k_{a_i}) / v_i)$.

Furthermore, we continue to specify the computational load from the conflict checking/resolution. The heart of this operation involves an exhaustive scan through all the `desired_cells`$_i$ to identify any conflicts with cells in the reservation map (as seen in Code 3.3). As such, the computational burden directly correlates with the total number of cells within `desired_cells`$_i$, represented as n_{dc}. Consequently, the computational complexity for conflict resolution is in the order of $\mathcal{O}(n_{dc})$.

Moreover, let us specify the computation load for each type of agent with respect to each method. For FCFS, since all of the resource reservation operations are done within the IMA. The computational load of IMA is the order of $\mathcal{O}((w_{cell} \cdot k_{a_i}) / v_i) + \mathcal{O}(n_{dc})$, which triggers every when it receives the reservation request. While DAs do not perform any computations or calculations, only sending requests to the IMA. Hence, DAs bear no computational workload.

On the other hand, with our method, the DAs retain the information on the reservation map and then perform path prediction and conflict resolution before even a single message is sent out. Hence, the computational load of DAs is $\mathcal{O}((w_{cell} \cdot k_{a_i}) / v_i) + \mathcal{O}(n_{dc})$. Once the reservation request is sent out from the DA's side and the IMA receives this particular request, the IMA is going to perform a validation process (similar to the conflict resolution but without the shifting process). This validation process is extremely important, specifically addressing the concurrency issues that might happen. Due to this, the IMA requires another computation of $\mathcal{O}(n_{dc})$. Table 3.2 sums up the computational load of each method separate by the agent types.

Agent types	**FCFS**	**Our method**
IMA	$\mathcal{O}((w_{cell} \cdot k_{a_i}) / v_i) + \mathcal{O}(n_{dc})$	$\mathcal{O}(n_{dc})$
DA	–	$\mathcal{O}((w_{cell} \cdot k_{a_i}) / v_i) + \mathcal{O}(n_{dc})$
Total	$\mathcal{O}((w_{cell} \cdot k_{a_i}) / v_i) + \mathcal{O}(n_{dc})$	$\mathcal{O}((w_{cell} \cdot k_{a_i}) / v_i) + 2 \times \mathcal{O}(n_{dc})$

TABLE 3.2: This table compares the computational load of each agent type on different automation control methods. The total value denotes the total computation load on both IMA and DA per one complete reservation cycle **without considering conflicts**

At first glance, the total computation of our method is higher than the FCFS when considering both agents. In particular, the difference amounts to $\mathcal{O}(n_{dc})$. However, this order of computation has not taken an element of reservation conflicts into account yet.

The element of reservation conflicts plays an important role in determining the computation load difference between FCFS and our method. Specifically, in the FCFS approach, conflicts are addressed by rejecting the requesting message and allowing DAs to repeatedly submit messages until their requested path is conflict-free. Under FCFS, the IMA treats each message as distinct, behaving as a trigger to execute the entire reservation mechanism. This implies that a higher occurrence of conflicts results in an increased influx of requesting messages, consequently elevating the computational burden on the IMA.

Let us consider the recorded amount of messages in Table 3.1. In the context of the 6,000 veh/hr scenario, the average message count is 41,480. However, notably, the notification of complete is only sent once per leaving vehicle. If we focus solely on messages that pertain to triggering the resource reservation mechanism, then we must deduct the 6,000 messages associated with this notification.

The number remains at 35,480 messages, which is still substantially high considering the number of inbound vehicles. There are different causes for this high number which will be outlined next. This number of messages composites two distinct types: DAs' request and IMA's response. We know for a fact that one reservation cycle requires these two exchange messages. Therefore, the count of reservation cycles can be deduced by taking half of the remaining message count, amounting to 17,740 cycles. Still, the number does not align with the inbound vehicles, which is only 6,000. The main cause of this phenomenon is the conflicts between reservations, rendering multiple requests sent from the individual DA. By dividing the remaining message count, 17,740, by the amount of inbound vehicles, 6,000 veh/hr, yielding a value of approximately 3 (2.95) requests/veh. This value implies that after the initial request message is sent, two conflicts occur, and two more requests are sent as follow-ups until DAs eventually acquire the approval.

Considering the potential reservation conflicts, our focus now shifts towards estimating the practical computational loads born by both DAs and IMA under different junction control methods. The earlier calculations revealed that in a scenario involving 6,000 veh/hr with the original FCFS approach, a single DA necessitates a minimum of three request messages to secure approval. These three messages initiate three cycles of the resource reservation mechanism, resulting in a computational load for IMA denoted as $3 \times [\mathcal{O}\left(\left(w_{cell} \cdot k_{a_i}\right) / v_i\right) + \mathcal{O}(n_{dc})]$.

In contrast, our proposed design resolves conflicts internally on the DAs' side. The conflict resolution is carried out three times to address two reservation conflicts, with the final conflict resolution acting as a verification step for the DA. Notably, the DAs do not need to recompute the path prediction multiple times when conflicts arise. Consequently, in this context, the computational load for DAs stands at $\mathcal{O}\left(\left(w_{cell} \cdot k_{a_i}\right) / v_i\right) + 3 \times \mathcal{O}(n_{dc})$. On the IMA side, conflicts do not affect the computational process; it

merely executes one request validation, thereby retaining the computational load at $\mathcal{O}(n_{dc})$. Table 3.3 presents a summary of the computational loads, taking reservation conflicts into account.

Agent types	**FCFS**	**Our method**
IMA	$3 \times [\mathcal{O}((w_{cell} \cdot k_{a_i})/v_i) + \mathcal{O}(n_{dc})]$	$\mathcal{O}(n_{dc})$
DA	–	$\mathcal{O}((w_{cell} \cdot k_{a_i})/v_i) + 3 \times \mathcal{O}(n_{dc})$
Total	$3 \times \mathcal{O}((w_{cell} \cdot k_{a_i})/v_i) + 3 \times \mathcal{O}(n_{dc})$	$\mathcal{O}((w_{cell} \cdot k_{a_i})/v_i) + 4 \times \mathcal{O}(n_{dc})$

TABLE 3.3: This table compares the computational load of each agent type on different automation control methods. The total value here denotes the total computation load on both IMA and DA per one complete reservation cycle with **two conflicts**

Analysing the table, the disparity in computational load between the two methods becomes evident. However, determining which method exhibits a lower total computation workload per reservation cycle might seem unclear at this point. It is important to note that the assumption of limiting conflict-resolving requests to just one during message counting was applied. This approach allows us to estimate the conflict occurrences in the experiments, although it sacrifices the ability to accurately retrieve the true rate of request/veh. The true rate of requests per vehicle depends solely on the computational speed of the machine running the simulations. A faster machine translates to a higher request/veh rate. This phenomenon arises because the IMA can process and return rejections swiftly, prompting DAs to send repetitive requests faster, even though most of them are unsuccessful. Practically, in the original FCFS, resolving a single conflict necessitates more than one reservation mechanism cycle.

From the table, The difference in the computational load of the two methods can be seen. It might be a bit unclear which method has the lower total computation workload per reservation cycle. Please note that an assumption that limits the number of requests resolving the conflict to one was used during the message counting. This allows us to estimate the number of conflicts within the experiments but sacrifices an ability to retrieve the true rate of requests/veh. The true rate of request/veh solely depends on the computation speed of the machine running the simulations. The faster machine is, the higher the requests/veh rate will be. The reason is that IMA can compute and return rejects faster, allowing the DAs to send repetitive requests faster, which are mostly unsuccessful as the space is being occupied. Practically, in the original FCFS, one conflict is likely to require more than one reservation mechanism cycle to be resolved.

For instance, let us consider three repetitive messages sent in an attempt to resolve conflicts – three messages/conflict. With the previously estimated value of two conflicts per vehicle, the total reservation mechanism cycles amount to seven cycles: one for the initial contact and another six for conflict resolution (three for each conflict). Applying

this context, the computational load on the IMA under FCFS increases to:

$$7 \times \left[\mathcal{O}\left(\left(w_{cell} \cdot k_{a_i}\right) / v_i\right) + \mathcal{O}(n_{dc})\right] \tag{3.5}$$

In contrast, our method eliminates the need for repetitive messages; DAs preemptively address conflicts internally before submitting a request. Consequently, the reservation cycle occurs just once, accompanied by a varying number of conflict resolution cycles contingent on potential conflicts (refer to the total load of our method in Table 3.3).

To put it simply, the computational load of FCFS and our method scale up based on the frequency of conflicts that occur within the systems. However, the load of these methods scales at a different pace. On the one hand, the computational load of FCFS scale up extremely rapidly as it allows the repetitive request messages to trigger the whole reservation mechanism. On the other hand, the load of our method scales up more gradually than the FCFS. This is because only the conflict resolution/checking operation is repeatedly executed, not the whole reservation mechanism. This highlights that our computationally decentralised approach manages overall computational loads more efficiently than FCFS, even when accounting for both agent types.

Furthermore, another significant consideration revolves around the dimension of time. In the context of the FCFS design, the IMA embodies a solitary node shouldering the entire computational load. It finds itself in the position of sequentially handling all incoming requests. Consequently, during periods of high traffic, the IMA becomes overwhelmed with an excessive amount of request messages, driving it to operate consistently at maximum capacity. This can potentially lead to instances where the IMA cannot respond to each DA simultaneously due to the ongoing processing of other requests.

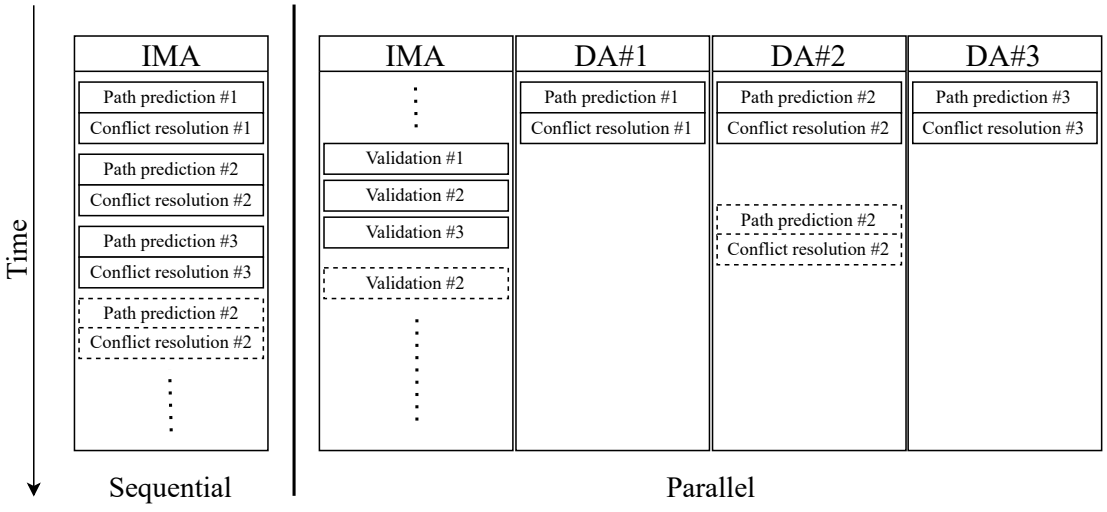

FIGURE 3.11: A comparison of burden tasks between FCFS and our proposed method categorised on an agent basis with respect to time. The #1 to #3 denote computation association to individual DAs.

To provide a more detailed perspective, let us compare the sequential nature of the FCFS method with the parallel approach proposed in our method. This comparison is depicted in Figure 3.11. It is important to note that this illustration focuses solely on computation workload and does not account for communication timing, assuming instant communication.

Consider a scenario where three DAs sequentially send reservation requests to the IMA. On the left side of the figure, we represent the computation workload of the IMA, encompassing path prediction and conflict resolution for each DA (DA #1 - #3). During this sequential processing, the DAs do not perform any calculations; thereby, there is no representation showing the DAs' responsible tasks. The dashed rectangle indicates a possible scenario where DA #2's request is rejected, prompting it to resend the request.

On the other hand, the right side of the figure illustrates the tasks assigned to the IMA and the DAs in a parallel processing scenario. Here, the computational load is distributed among multiple DAs, each performing internal calculations. Then, the IMA validates the requests from the DAs. Similar to the sequential scenario, dashed rectangles denote possible request rejections, but these rejections arise due to concurrency issues in practice rather than conflicts between reservations.

While the parallel method (in this figure) shows only a minor timing gap compared to the sequential method, real-world scenarios typically result in a higher rate of conflicts in the FCFS's sequential approach. This amplifies the computation burden on the IMA, overwhelming it with a massive number of requests, leading to more significant delays that slow down the responses. In essence, during busy periods causing numerous conflicts, our parallel method offers reduced IMA workload, enabling faster response times compared to the FCFS's sequential operation.

In summary, our method introduces a distinctive advantage – the distribution of computational loads among multiple DAs. This architectural choice facilitates parallel computation by DAs, allowing the IMA to react or respond on a request basis only. As a result, the IMA is relieved from the burden of constant full-load operation. This benefit is primarily attributed to the restructured design, wherein DAs now undertake the resource-intensive task, i.e., conflict resolution (due to its repetitiveness). Therefore, our system offers the IMA a lighter operational profile, enabling it to swiftly deliver rejections or approvals without being strained by continuous heavy computation loads.

3.6.2 Evaluation with SUMO

This evaluation section primarily centres on traffic scenarios featuring obstructions. We begin with a comprehensive breakdown of our implementation of autonomous junction control within the SUMO simulation environment. Here, we delve into the interface for SUMO and the foundational structure of our system.

Subsequently, we detail the methodologies employed to incorporate obstacles into the SUMO simulation, a process necessitated by the absence of pre-existing functions catering to obstruction objects. Moving forward, the strategy employed for positioning these obstructions will be described. The distinct obstruction scenarios are crafted through strategic placement, each inducing varying levels of impact and consequence. This strategy enables the creation of diverse scenarios, amplifying the depth of our evaluation.

Lastly, we describe our experiment settings and present the experiment outcomes. This exploration aims to provide a comprehensive understanding of the performance and efficiency of the implemented approach across these obstructed traffic scenarios compared to different junction controls.

3.6.2.1 Implementation with SUMO

This section provides a fundamental implementation of our system using SUMO. Note that within this evaluation phase, we refer to our model as Resilient Intersection Management with Collision Avoidance or RIMMCA, as the objective of this phase is not the computational load. The term "computationally decentralised" is not included in the preferred name.

Our objective is to benchmark our RIMMCA model against two distinct junction control methods: traffic lights (TL) and FCFS. To accomplish this, we utilise the open-source traffic simulator SUMO (Simulation of Urban MObility) Krajzewicz et al. (2002). The client-server-based Traffic Control Interface (TraCI) Wegener et al. (2008), an integral component of SUMO, enables the development of external applications that can apply real-time control over simulated vehicles. In particular, this interface operates in a client-server architecture, with the simulation functioning as the server and external applications interacting as clients. Various setting files, encompassing network configuration, routing, traffic light programs, vehicle types, and additional elements, such as vehicle counters or detectors, can be provided as inputs to the simulation.

On the client side, multiple commands can be executed to retrieve information and trigger actions on simulated objects. This versatility allows for actions like obtaining a vehicle's speed or changing the state of a traffic light. The main use of the retrieved information is to form communication messages that interconnect between two types of agents and perform their internal operations (e.g. conflict resolution).

Leveraging this capability, we implement FCFS and RIMMCA algorithms within the simulated junctions using the available TraCI. To achieve this integration, two critical modifications were introduced to the SUMO environment:

1. Vehicles can progress beyond the stop bar, even when traffic lights are red. This is to allow DAs to follow their reservation at a specific point in time. [4]

2. Vehicles are allowed to disregard their car-following models while crossing the junction. This relaxation facilitates a closer spacing between vehicles, optimising the traffic capacity of FCFS and RIMMCA algorithms.

Our approach emphasises an object-oriented design for the client side, where various Python classes correspond to distinct SUMO objects. The junction class represents the IMA, functioning as the central unit responsible for scheduling DA access. The vehicle class represents DAs traversing the simulated region, endowed with unique attributes like ID, velocity, width, length, and orientation (for more information, refer to Section 3.2.2). This class is also responsible for the calculations related to path prediction, obstruction detection, and conflict resolution (refer to Fig. 3.3.2). These computations can be performed by utilising the broadcasted information from the IMA as DAs enter its communication range.

Additionally, we create an obstacle class to simulate obstructions within the system. These obstacles could be positioned in any available space within the junction area by specifying their properties: position, radius, and safe ring. However, the process of obstacle placement required a distinct treatment, which will be elaborated upon in the subsequent discussion.

3.6.2.2 Obstacle Placement

Given that SUMO lacks direct support for obstacle placement, we explain how we manage obstacles within the simulated environment. The obstacles are categorised into two distinct types: in-lane and in-junction obstacles, based on their positioning and their impact on the traffic.

In the case of in-lane obstacles, our methodology entails the placement of an immobile vehicle at a specific position within a lane, consequently occupying a portion of the lane's available space. Typically, this obstacle type explicitly blocks vehicles manoeuvring at the lane's exit. Notably, we leverage existing SUMO vehicle objects to create these obstacles. The advantage of this approach lies in its responsiveness: all simulated vehicles automatically react to the obstruction by adjusting their behaviour, such as strategic lane changes and best lane selections. This natural responsiveness obviates the need for algorithmic manipulation for this obstacle type. As a result, both the TL and FCFS algorithms are evaluated without necessitating collision avoidance extensions.

[4]The red traffic lights is considered within the car-following model, and they serve the purpose of assisting the DAs in waiting and decelerating while waiting for their access time slots

On the other hand, the scenario of in-junction obstacles presents a distinctive challenge. In contrast to in-lane obstacles, placing an immobile vehicle to represent in-junction obstacles would yield unintended consequences. Specifically, the car-following model by (Krauß et al., 1997) could trigger emergency braking for crossing vehicles within the junction. In the context of TL, this could potentially lead to a deadlock. To address this issue, we adopt an alternative approach. Our solution involves employing a point of interest–a circle-shaped object provided by SUMO–to symbolise in-junction obstacles. While this object lacks physical relevance to drivers' perceptions, it effectively serves as a representation of the obstruction. This approach mitigates the risk of unexpected behaviours arising from the car-following model, especially emergency breaking, ensuring a smoother simulation run.

Furthermore, considering that only a few researches have explored the implications of obstructions within junction areas, a new evaluation framework was necessary. We conducted comparative assessments by benchmarking our algorithm against a basic collision avoidance extension of TL and FCFS. This extension involves the closure of lanes obstructed by the object throughout the simulation. In particular, only the lanes that have paths/trajectories obstructed or intervened by the obstacles will be closed, while the others remain open. We refer to this lane-closing solution as TL-LC (Traffic Lights with Lane Closure) and FCFS-LC (First-Come-First-Served with Lane Closure), respectively.

The experimental framework is structured around two primary categories: in-lane obstruction and in-junction obstruction scenarios. Each category serves as a distinct context for assessing the impact of obstacles on traffic performance.

For the in-lane obstructions, our objective is to deliberately create situations that negatively influence traffic throughput. To achieve this, we position an obstacle at the exit of the leftmost lane (as illustrated in Fig. 3.12a). In a left-hand driving environment, this particular lane often experiences extensive queuing due to slow left-turn movements. Placing an obstacle at this critical point has the potential to significantly disrupt overall traffic flow and throughput, providing valuable insights into the impact of obstructions on the system.

For the second category, we aim to study in-junction obstructions that block multiple lanes. To this end, we position the obstacle in the midst of the junction, thereby generating scenarios where three lanes are obstructed simultaneously (as depicted in Fig. 3.12b). This controlled three-lane blockage serves as an indicator of the obstruction's impact on traffic throughput, while still allowing junction control mechanisms to maintain a reasonable flow. Importantly, our evaluation framework confines the placement of obstacles to positions that result in the blockage of precisely three lanes. Beyond this configuration, the dynamics become less predictable and more sensitive to the junction's non-symmetric design, making it difficult to sustain stable outcomes.

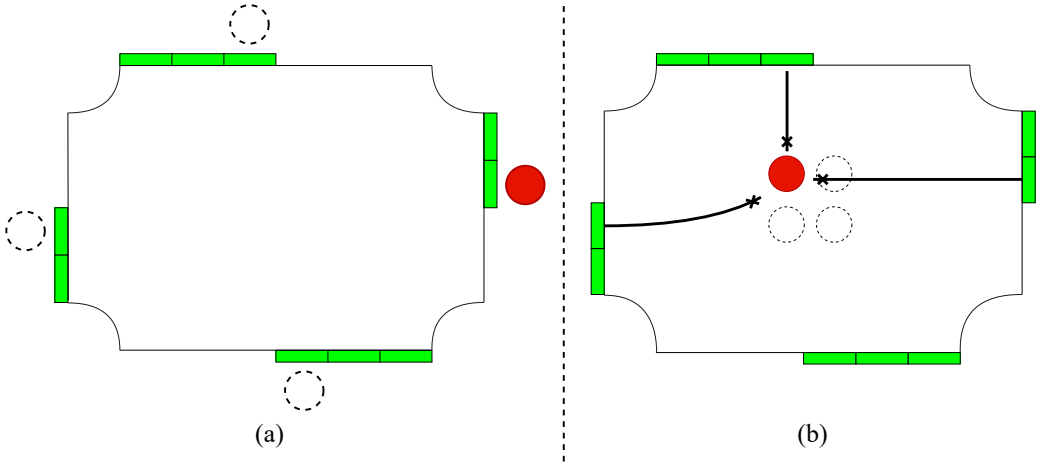

FIGURE 3.12: This is an example placement of obstacles where red solid circles represent the obstacle while the dashed circles represent the possible positions of the obstacle. Situation (a) shows an example case where the obstacle is placed at the exit of the inbound lane. While in situation (b), the obstacle is placed within the junction area that blocks multiple crossing paths.

In the next sections, we describe the experiment settings that aim to replicate real-world scenarios. These will provide comprehensive insights into our RIMMCA model's performance under different obstruction scenarios, demonstrating its efficiency in managing diverse obstruction challenges.

3.6.2.3 Experimental Settings

In this experiment, our objective is to replicate a practical environment that closely mirrors real-world traffic conditions. This is to help us comprehensively explore the impact of obstructions on non-ideal traffic scenarios. To do so, we reproduce a junction resembling the layout found in Manhattan, setting it apart from the idealised junction configuration in (Dresner and Stone, 2008).

Our targeted junction, situated in the city of New York, offers access to authentic traffic data through New York State Department of Transportation (2016). This dataset serves as a valuable resource, enabling us with useful information about the average daily traffic counts for individual roads during the year 2016. Given this dataset, we are able to derive estimates of traffic flow within our simulation. Specifically, we establish a baseline flow of 2,300 vehicles per hour, a figure that experiences a 10% surge during peak hours, reaching a peak rate of 2,500 vehicles per hour.

By utilising these traffic counts, we can allocate this traffic flow across the two relevant roads—Park Avenue South and East 23rd Street—according to their real-world usage patterns. This allocation reflects 67.46% of the total flow on Park Avenue South and 32.54% on East 23rd Street, mirroring their respective contributions to the overall traffic volume. To be specific, the distribution of vehicles across various trajectories within

each road is handed out in Table 3.4. To demonstrate the meaning of this table, for instance, let us consider Park Avenue South: 67.46% of the total vehicles are generated on this road, of which 33.73% continue straight, while 16.865% opt for either a left or right turn. This allocation enables us to generate a practical vehicle flow in each simulation run.

Road	**Vehicles Share**			
	straight	left	right	total
Park Av South	33.73%	16.865%	16.865%	67.46%
East 23rd Street	6.508%	13.016%	13.016%	32.54%

TABLE 3.4: This table presents the distribution of simulated vehicles across various trajectories for each road within the junction environment. The percentages indicate the proportion of vehicles generated on a specific road that follow distinct trajectories, i.e., straight or making left or right turns.

To specify the DAs vehicles properties, they include a width of 1.8 m, length of 5 m, maximum speed of 11.18 m/s ($\approx$ 40 km/hr), acceleration of 2.6 m/s^2, deceleration of 4.5 m/s^2, maximum junction crossing speed is 7 m/s ($\approx$ 25 km/hr), time step 2.5 s, and minimum gap of 2.5 m. The size of the obstacles is $r(O) = 2.5\ m$, $r_{safe}(O) = 4.5\ m$. The communication range of the IMA is 40 m. Note that only one obstacle will be placed per simulation run.

To clarify the effectiveness of the specified 40-meter communication range in cooperative transmission between the IMA and the DAs, we need to assess its functionality. The purpose of broadcasting this message is to alert the DAs about the presence of autonomous junctions and enable them to make necessary preparations accordingly. In our approach, DAs submit a request when they are within five meters from the stop line (as discussed in Section 3.5). This leaves a substantial buffer of 35 meters, allowing ample room for DAs to acknowledge the IMA's existence.

As vehicles decelerate and approach the stop line within these five meters, four key tasks are required to complete: (i) path prediction & conflict resolution on the DAs' side, (ii) DAs sending a reservation request, (iii) IMA verifying the request, and (iv) IMA sending an approval back to the DAs. Tasks (i) and (iii)[5] are operational tasks, assuming that they take 10 + 10 = 20 ms or 0.02 s to compute. Tasks (ii) and (iv) are communication tasks, and we assume a communication frequency of 2.4 GHz with a minimum practical transfer rate of 20 Megabits per second (Mbps). Let us assume the size of the request and approval packages is one Megabit each, requiring a transfer time of $2 \div 20 = 0.1$ s. In total, DAs require 0.12 s to secure their reservation. Assuming they approach the junction at a maximum crossing speed of 7 m/s, it would take $7 \times 0.12 = 0.84\ m$ to fully secure the reservation within the 5-meter gap. This calculation

[5]Even though the IMA's validating computation time is minimal, it can accumulate and become substantial in some scenarios, hindering the immediate response to requests. The computation time of IMA is estimated with a high value to anticipate such scenarios.

demonstrates sufficient distance for maximum crossing-speed vehicles when encountering challenging scenarios, factoring in worst-case computation delays for both agent types and a significantly slow connection speed of 2.4 GHz.

Furthermore, with the 35-meter buffer and assuming a maximum message size of one Megabit, broadcast messages would take a mere 0.05 s transmission time to travel from the IMA to the DAs. Considering the maximum speed achievable by DAs, which is 11.8 m/s, it becomes evident that DAs would traverse a distance of only $11.8 \times 0.05 = 0.59$ m during the 0.05-second transmission time. There is plenty of road space for DAs to perform any strategic lane changing or decelerating to anticipate queuing or stop lines.

Hence, the chosen 40-meter communication range ensures the successful transmission of essential messages, establishing knowledge of DAs and sufficient for DAs to attain their reservation for our proposed autonomous junction control.

3.6.2.4 Experimental Results

In this subsection, we present the results and discussions of our experiments. Each simulation run spans a duration of 1 hour, during which randomly generated departure time and vehicle trips based on estimated traffic flow and usage weight are employed (see Table 3.4). To ensure accuracy, each experiment is repeated at least 20 times, and the reported values below represent the averages across these runs.

Traffic Metrics

We assess our system using three traffic metrics: flow rate (measured in *vehicles/hour* or *veh/hr*), queue length (measured in *metres*), and delays (measured in *seconds/vehicle* or *s/veh*). To be more specific, the flow rate denotes the number of vehicles that progress through the junction within a time frame. The queue length is the average queue length on the inbound roads measured throughout the simulation run. Lastly, delays here denote how long DAs are at full stop (speed less than 0.1 m/s).

No Obstructions

Initially, we conduct obstruction-free experiments to record the optimal performance of traffic lights, FCFS, and RIMMCA. The results demonstrate a clear superiority of both FCFS and RIMMCA over traffic lights. Specifically, the optimal performance of traffic lights produces a flow rate of $1,569.8$ veh/hr, a queue length of 33.28 m, and delays of 93.43 s/veh. In contrast, FCFS and RIMMCA have considerably higher performance, a flow rate of $2,402$ veh/hr, a queue length of approximately $0.13 - 0.15$ m, and delays of just 0.01 s/veh. Notably, flow rates and delays are recorded at the end of each simulation run.

Having established this baseline, we move on to experiments involving the two obstruction types: in-lane obstructions and in-junction obstructions. We begin with the in-lane obstruction scenarios.

In-Lane Obstruction Scenarios

In this phase, we construct four distinct in-lane obstruction scenarios by strategically placing obstacles at various positions (as depicted in Fig. 3.12a).

The results reveal a significant degradation in the performance of both TL and FCFS. Specifically, TL experiences a sharp drop in flow rate to $1,023.75$ veh/hr, a queuing length increase to 44.85 m, and delays escalating to 162.01 s/veh. Similarly, FCFS encounters a notable performance drop, with a flow rate of $2,371.75$ veh/hr, a queuing length of 11.2 m, and delays of 1.61 s/veh. On the other hand, RIMMCA demonstrates high resilience in the face of in-lane obstructions, maintaining its performance close to the optimal point. It sustains a flow rate of $2,380$ veh/hr, a minimal queuing length of 0.25 m, and small delays of 0.03 s/veh.

The discrepancies in performance among TL, FCFS, and RIMMCA are attributed to their capability of handling the obstacles at lane entries. In TL and FCFS, the sudden need for lane changes due to obstructions can disrupt the traffic flow, essentially impacting the smoothness of the traffic and resulting in longer queues and higher delays. In contrast, RIMMCA's advantage becomes apparent here: its reserved path already anticipates the collision avoidance movements, behaving as lane-changing manoeuvres. This approach ensures vehicles can access the junction without on-the-spot lane changing, maintaining the flow's continuity. To illustrate this, Fig. 3.13 shows the average queuing lengths over time for TL, FCFS, and RIMMCA.

By using the optimal performance as a baseline, the traffic lights and FCFS offer a flow rate $\approx$ 65% and $\approx$ 98% of the optimal point, while RIMMCA offers 99% of the optimal point. From the results, it can be seen RIMMCA can outperform both traffic lights and FCFS even with the presence of in-lane obstacles.

We continue to the experimentation of in-junction obstructions. This consideration of in-junction obstructions is a rather important evaluation of autonomous junction control as it can critically impact the traffic flow of the junction. Specifically, multiple inbound lanes will be blocked by the obstacles. The next section will demonstrate how different junction control methods respond to this challenging scenario.

In-junction Obstruction Scenarios

Building upon the described placement of obstacles, the exploration of in-junction obstruction scenarios involves positioning obstacles at four key locations (as shown in Fig. 3.12b). These distinct scenarios present a different challenge to both Traffic TL

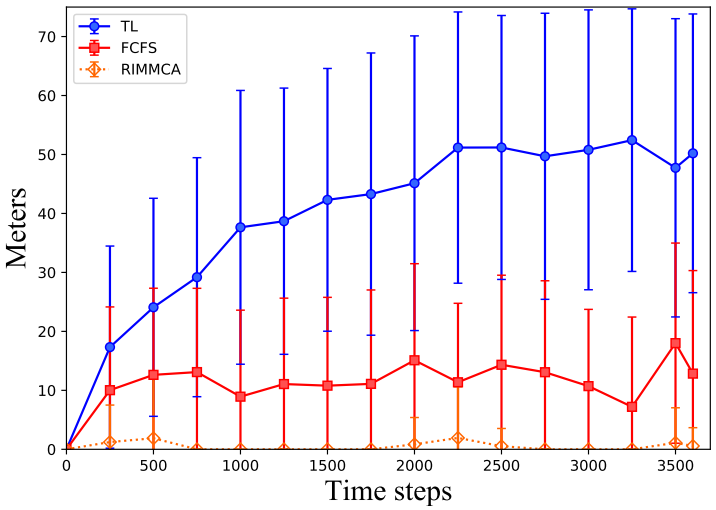

FIGURE 3.13: Average queuing length results in the standard deviation of four different in-lane obstruction scenarios (20 runs per scenario).

and FCFS methods, even though they employ similar lane-closing solutions. The subsequent shows a comparative analysis of the performance of these methods alongside our proposed RIMMCA within this complex context.

Starting with TL-LC, the performance demonstrates improvement compared to the in-lane obstruction scenarios, providing a flow rate of $1,416.7$ veh/hr, a queuing length of 42 m, and delays of 118.82 s/veh. FCFS-LC, on the other hand, records a flow rate of $2,170$ veh/hr, accompanied by queuing length and delays, measuring at 1.3 m and 0.6 s/veh, respectively.

Meanwhile, our RIMMCA experiences a slight performance drop compared to the in-lane obstruction case with a flow rate of $2,260$ veh/hr, queuing length of 2.73 m, and delays of 4.7 s/veh. This marginal decrease can be attributed to the blockage caused by the in-junction obstacle positioning, resulting in the obstruction of multiple lanes. The comparison graph of queuing length among the three junction control methods in the presence of in-junction obstacles is depicted in Fig. 3.14.

In comparing FCFS-LC and RIMMCA, FCFS-LC reveals a superior average queuing length. This is attributed to the overlapping of required spaces between obstructed-free paths (resulting from collision avoidance) and normal paths. As a result, unobstructed DAs are sometimes unable to promptly reserve their time slots due to the spaces being occupied by other vehicles manoeuvring to avoid obstructions. This increases waiting times for unobstructed DAs and elevates overall waiting time. However, despite the minor trade-offs in queuing length and delays, RIMMCA still surpasses both TL-LC and FCFS-LC in terms of flow rate. Outstandingly, RIMMCA manages to maintain a flow rate of approximately $\approx$ 94% of the optimal point, while TL-LC and FCFS-LC achieve only $\approx$ 90% of the optimal point.

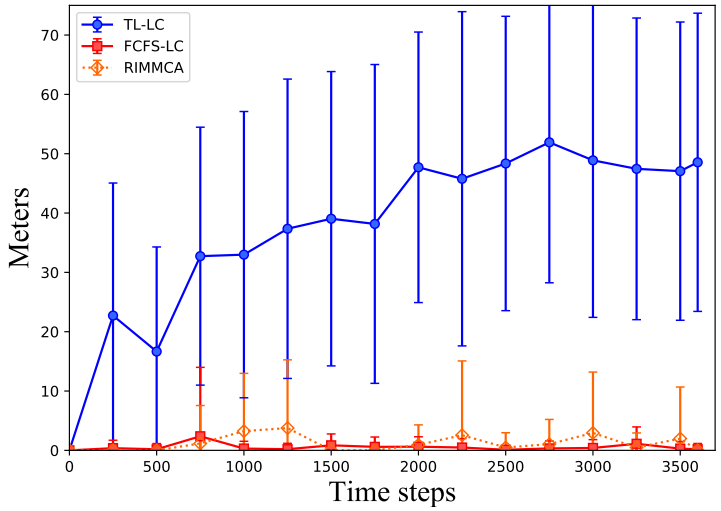

FIGURE 3.14: Average queuing length results with a standard deviation of four different in-junction obstruction scenarios (20 runs per scenario).

An overview of the comparative performance between the three junction control methods is encapsulated in Table 3.5. This experiment provides useful insights into the capacity of each control algorithm to effectively anticipate the challenges posed by in-junction obstructions.

	Flow Rate (veh/hr)			**Queuing Length (m)**			**Delays (s/veh)**		
	TL	**FCFS**	**RIMMCA**	**TL**	**FCFS**	**RIMMCA**	**TL**	**FCFS**	**RIMMCA**
Optimal	1,569.8	2,402	2,403	33.28	0.13	0.15	93.43	0.01	0.01
In-lane	1,023.75	2,371.75	2,380	44.85	11.2	0.25	162.01	1.61	0.03
In-junction	1,416	2,170	2,260	42	1.3	2.73	118.82	0.6	4.7

TABLE 3.5: This table shows a performance comparison in three metrics: flow rate (veh/hr), queuing length (m), and delays (s/veh) regarding TL, FCFS and RIMMCA in different experiment scenarios.

In conclusion, the evaluation of our RIMMCA alongside TL and FCFS control methods in the context of both in-lane and in-junction obstructions has highlighted the robustness and adaptability of RIMMCA. In scenarios involving in-lane obstructions, RIMMCA demonstrates superior performance compared to TL and FCFS, maintaining higher flow rates, shorter queuing lengths, and lower delays. Integrating collision avoidance mechanisms into RIMMCA's reserved paths allows smoother traffic flow by minimising the disruptions caused by on-the-spot lane changes. Furthermore, even in the more challenging in-junction obstruction scenarios, where the obstructions affected multiple inbound lanes, RIMMCA continues to outperform TL-LC and FCFS-LC in terms of flow rate. However, there are slight trade-offs in queuing length and delays for RIMMCA due to the overlapping between obstruction avoidance paths against normal paths. This finding shows the potential adaptability of our proposed method in managing complex and challenging situations in the real world involving various obstruction scenarios, e.g., a sinkhole or a construction site.

3.7 Discussion & Limitations

One limitation of this work is the challenges posed by simulation tools. Our initial search for a simulation platform to integrate the FCFS algorithm while allowing modifications to align with our computationally decentralised concept was unsuccessful. We developed a customised simulation tool that enabled us to examine the computation load disparities between state-of-the-art methods and our proposed approach. However, the customised simulation tool is imperfect and has several limitations. It lacks realism in various traffic components such as traffic and vehicle dynamics, vehicle weight, lane-changing behaviour, and queuing dynamics. This becomes apparent in cases of extremely high-traffic scenarios, where such inaccuracies could affect the results.

Continuing to enhance this customised tool even further would require an unreasonable amount of time and effort, diverting us from other aspects of the autonomous junction control. This prompted us to adjust the second phase of our empirical evaluation, utilising the open-source microscopic simulation tool, SUMO. By utilising SUMO, we benefit from a high-fidelity representation of infrastructure and an accurate configuration of vehicle dynamics and behaviours. The simulation can closely replicate real-world vehicle dynamics, allowing us to investigate the performance of our approach in more detail. However, SUMO's programming language interface, Python, brings its own challenges.

Due to its unique format or syntax design, Python offers many advantages allowing users to learn and understand its instructions, commands and logic easily. Users from various backgrounds can understand the code easily, as these commands are run sequentially, theoretically, line by line. However, this property of Python has its drawbacks. Due to its nature, it is difficult to write the code in a parallel manner. Writing in parallel would be more suitable to properly simulate the agent in practical situations as each agent should operate on their own time. The concurrency issues are certainly going to happen, and many agents are going to behave unexpectedly, which is due to the problems that will arise, e.g., inconsistency of shared data. Therefore, we opted for sequential agent execution to simplify system complexity, but this choice limits our ability to directly quantify the frequency of concurrency issues.

Furthermore, measuring the computational load of agents presents a challenge due to hardware inconsistencies and the non-parallel system design. The IRIDIS high-performance computing facility, our primary experimental runtime environment, allocates different hardware nodes to each request, introducing variability in runtime results. Given a similar simulation setting and traffic configuration, different hardware will produce different runtime results since each one has a different clock speed, let alone background tasks that might slow the running of the simulation. To address this, we indirectly assessed computational load by counting exchange messages. Even with

precise message counts, numerous assumptions were necessary to estimate the computational load for each method.

Additionally, it is important to highlight that the second phase of our empirical evaluation, building upon the core algorithm of the customised simulation (see Section 3.6.1), presents challenges in code optimisation. The implemented code has not yet been properly optimised, leading to unstable vehicle movements within the simulation. The major cause of this issue lies in the SUMO design that all simulated vehicles follow fixed or guided paths integrated into each traffic element. However, the deviation from these paths for collision avoidance purposes explicitly violates SUMO's original design. This misalignment between the DAs' desired paths and fixed path causes vehicles to have jumpy movements, in which SUMO try to realign them onto the fixed paths, contributing to a notable standard deviation in the simulation results.

With regard to the results, the results within the context of obstruction experiments reveal an anomaly between delay and flow rate outcomes. Ordinarily, delays and flow rates share an inverse relationship, where high delays correspond to lower flow rates. Yet, our results seem to show otherwise. An observative investigation has been made to indicate a potential cause for this phenomenon: FCFS may lead to vehicle stuttering, unintentionally avoiding complete stops. This unique behaviour could contribute to the unexpectedly low delay values associated with FCFS. For example, see Table 3.5, under in-junction obstruction scenarios, FCFS shows lower delays than RIMMCA while still lagging behind RIMMCA in terms of flow rate performance. In further experiments, another metric will be used instead, namely travel time. Travel time is the time used to move from one point to another that essentially includes all the delays while travelling.

Despite certain works that have been done with this project, we decided to shift our research direction away from obstruction avoidance due to various reasons. Firstly, we acknowledge the importance of addressing real-world uncertainties, particularly with safety and practicality as the ultimate goals. However, the context of obstructions, especially static ones, remains relatively infrequent. Addressing these obstacles appears more as an extension or refinement of existing methodologies rather than a novel algorithmic innovation. Besides, at the moment, SUMO does not fully support obstruction placement, and it needs to be hard-coded from scratch, which significantly limits the modification of obstruction. This presents certain difficulties to continue improving the work or conducting more experiments involving obstruction avoidance. We, therefore, consider this work as an extension of our research that solely addresses the uncertainty and shift the research direction towards the concept of platooning. It holds the potential for advancing autonomous junction control by leveraging the connection capability of CAVs to improve traffic efficiency even further.

Chapter 4

Dynamic Platoon Formation

We propose a novel resource reservation junction management mechanism designed to dynamically form the platoons, thereby addressing Research Challenge 4. This innovative approach aims to enhance the platoon formation process beyond the greedily heuristic method utilised in the seminal work of Jin et al. (2013). The existing heuristic approach lacks a comprehensive assurance of the advantages that platooning can offer to the traffic system. In their approach, platoons are formed as vehicles enter the communication range of the central unit and manoeuvre across the junction. However, this straightforward methodology does not consider potential conflicts or losses to the system arising from the platoons' massive required time slots.

However, to resolve this issue, we entrust the IMA with the additional task of calculating the benefits and costs of forming a platoon in terms of overall waiting time to maximise the benefits to the system. Particularly, the benefit calculation is executed in real-time taking the changes in the traffic environment into account, thereby dynamically determining the optimal size of the platoons. Moreover, the cell-based approach (described in Chapter 3) is also improved in order to support different-sized platoons effectively.

This chapter is structured as follows. Firstly, we present an adjustment of the vehicle model to support the formation of platoons. Secondly, an explanation of our dynamic platoon formation algorithm and the resource reservation is provided. Thirdly, we explain how the driver agents' protocol is modified to support the platoon. Lastly, we evaluate our algorithm against the non-platoon state of the art.

4.1 Traffic Model

We base our model on the original FCFS model in Dresner and Stone (2008) that considers a road junction managed by an IMA. The IMA is able to grant its resource (time-slotted space in the junction) to each DA—to coordinate each vehicle's movement. Most of the models are similar to chapter 3 except for the vehicles and the junction. We then describe how we model the vehicles and the junction.

4.1.1 Vehicles

The model is similar to the vehicles model in Section 3.2 comprising of fundamental properties such as length, width, maximum speed etc. However, to model the platoon, we add a new property named $status_i$ to represent or label one of three states of a vehicle in our system, which are:

1. *leader* – A leader of a platoon that has complete knowledge of the follower(s)' properties.
2. $follower$ – A follower in a certain platoon.
3. *null* – An ordinary DA does not belong to any platoon, later referred to as a standalone agent.

Nevertheless, the vehicles in our system are initially labelled as *null*. Unlike existing work by Jin et al. (2013); Bashiri and Fleming (2017); Bashiri et al. (2018) where all vehicles are labelled as they enter the IMA's communication range, in our approach, platoons are formed dynamically only when they are queuing and meet certain conditions ensuring the benefits to the overall system. These conditions will be specified in Section 4.2.

4.1.2 The junction

This section entails the junction model within this work that goes beyond the scope of the previous study and objectively leverages the platooning capabilities.

Unlike various platooning works such as Jin et al. (2013); Bashiri and Fleming (2017), which focused solely on ideal junction geometries, our study takes a step further by considering more realistic junction scenarios. To achieve this, we utilise the SUMO simulation, allowing us to model a complex junction configuration. In line with the methodology established in Chapter 3, we centre our experimentation around a representative junction found within the Manhattan road network, specifically intersecting between Park Avenue South and East 23rd Street.

Additionally, we enhance the communication coverage of the IMA by extending it from 40 meters to 55 meters. This deliberate extension naturally increases the workload of the IMA, yet the previous 40-meter coverage was relatively limited. To illustrate, consider an example scenario in which the minimum vehicle length is 5 meters, and a minimal gap of 2.5 meters is maintained. In the 40-meter communication coverage context, the longest queue within certain lanes is constrained to approximately $40 \div (2.5 + 5) \approx 5\ veh/lane$. Moreover, the very first vehicle often secures its reservation, thereby making it unavailable for platoon formation. Consequently, only four vehicles remain eligible for platoon formation.

On the other hand, the decision to broaden the coverage aims to encompass more vehicles within the platoon formation consideration. With a 55-meter coverage, the vehicle count per lane reaches around $55 \div (2.5 + 5) \approx 7\ veh/lane$, thus enhancing the potential for platoon formation. With this coverage, the platoon formation potential grows with six available vehicles. Therefore, not only the likelihood of capitalising on the benefits of platooning will increase, but it also widens the scope of assessing the platoon's impact across multiple lanes.

Note that the IMA operates with the assumption that it has knowledge of all vehicles' positions and other properties within the communication coverage.

We next define our junction management mechanism that dynamically forms platoons while minimising overall waiting time and ensuring safe passage for DAs and platoons.

4.2 Dynamic Platoon Formation Algorithm

Having defined the model elements, we next define our junction management mechanism that dynamically forms platoons while minimising overall waiting time and continue describing the safe passage reserving process for standalone DAs and platoons.

Due to the platoon synchronisation movements, follower DA(s) can cross the junction in a less interrupted manner than standalone ones, reducing unnecessary braking and accelerating. Thus, the travel time and delays are reduced cumulatively with every platoon formed. The larger the platoon size, the greater the reduction. However, by doing this, the system may suffer cumulative delays elsewhere, i.e. so-called *externalities*. For instance, as the platoon requires large reserved space and time slots, forming one on a lane may prolong any future reservations and cause multiple DAs to wait, leading to substantial delays in the flow. Therefore, forming a platoon greedily, i.e. as soon as a DA enters the communication range (as in Jin et al. (2013)), is unlikely to be optimal or beneficial. To alleviate this issue, we propose a dynamic platoon formation to ensure the *cost* efficiency of every platoon formed.

The *cost* in our study refers to the overall waiting time across the junction, which is greatly affected by the existence of platoons. Specifically, the effect is categorised into two types: (i) the reduction of waiting time λ_{plt} and (ii) the increase in waiting time T_{plt}. As a platoon forms, the reduction comes from the reduced stop-and-go movements, while the increase comes from the cumulative delays of multiple DAs on the other inbound lanes.

By virtually grouping consecutive queuing DAs into multiple groups and calculating their λ_{plt} and T_{plt}, the algorithm can decide on platoon formation in real-time. If any groups of DAs make the reduction higher than the increase, i.e., $\lambda_{plt} > T_{plt}$, those DAs will be logically merged together, effectively forming a platoon. DAs in a platoon will adjust their driving behaviour according to $status_i$ (see later Section 4.3). It should be noted that merging can be done only in two scenarios: merging two standalone DAs together and creating a new platoon, or merging a standalone DA with an existing platoon, thus making it longer. We emphasise that, using our approach, the platoon size is non-static and can be *extended* dynamically as long as a net benefit is obtained (unlike Jin et al. (2013); Bashiri and Fleming (2017); Bashiri et al. (2018)).

Example cases of platoon formation can be seen in the left part of Fig. 4.1. Here, lane01 shows a typical case where two vehicles are one possible platoon. In lane02, the IMA is considering extending the size of an existing platoon by including the vehicle behind. In lane03, the vehicle with granted reservation is no longer considered in the platoon formation while the rest still do. Lastly, in lane04, the IMA considers vehicles even when they are still moving. Furthermore, the right-hand side of Fig. 4.1 shows the overall process of DAs handling platoon formation, including sharing some basic information with the IMA, retrieving a labelling response from the IMA, and adjusting their driving behaviour.

Next, we explain how the IMA calculates the cost of platoon formation, starting with the reduction of waiting time, λ_{plt}.

4.2.1 Calculating the Reduction of Waiting Time

Whenever a platoon *can* be formed, i.e. a DA approaches another queuing DA that shares a similar route, the IMA can suggest two possible choices:

(i) Merging the approaching DA with its preceding one(s) into a platoon.

(ii) Doing nothing and leaving the approaching DA as a standalone one.

Assuming the DAs in the platoon cooperate via CACC, as suggested in Segata et al. (2014), the follower DAs only follow the trajectory and speed recommended by their leader. Once the leader acquires a reservation, in situation (i), the follower DAs are

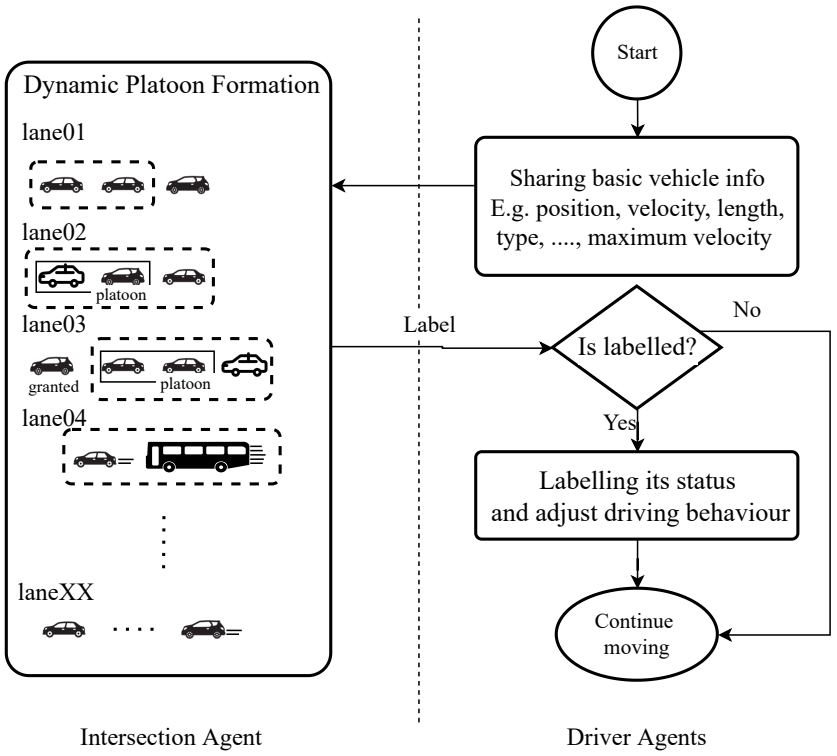

FIGURE 4.1: The interaction between the IMA and DAs where vehicle icons represent DA in certain lanes. Dashed rectangles represent a group of DAs possible for platoon formation, while the solid rectangles represent DAs that are already in platoons. Square dots represent additional lanes and DAs being considered in the platoon formation algorithm.

less likely to perform stop-and-go movements to cross the junction, which is where the reduction in waiting time comes from. In situation (ii), however, the approaching DA must slow down and wait for its reservation slots, which is more time-consuming. Therefore, the amount of reduced waiting time is calculated from the travel time difference between situations (i) and (ii).

To calculate this time difference, an estimated time of arrival (ETA) and estimated clearance time (ECT) – the amount of time vehicles need to manoeuvre and completely leave the junction – will be used. We first focus on computing the travel time between two points that incorporates ETA and ECT.

DA Travel Time: We build on Jin et al. (2013) to calculate the time that the DA requires to travel a certain distance without exceeding v^{max}. The function, called $cruise(d, a_i)$, is defined as:

$$cruise(d, a_i) = \frac{v_i^{max} - v_i^t}{\alpha_i^{max}} + \frac{1}{v_i^{max}} \left[d - \frac{\left((v_i^{max})^2 - (v_i^t)^2\right)}{2 \cdot \alpha_i^{max}} \right] \tag{4.1}$$

where d is the travel distance, and v_i^t is the velocity of an input DA a_i at time t. The first term denotes the acceleration time, and the second term denotes the time that the

vehicle travels at a constant speed (v_i^{max}). We next expand on this to compute the ETA and ECT.

ETA: In Jin et al. (2013), a vehicle's ETA is rather simple which does not account for any braking, as the settings are only on a two-way junction with light traffic. Our model involves more complex situations with higher traffic volumes, where vehicles regularly slow down approaching a stop line. Hence, a DA's ETA has to account for braking. Let d_i^{line} be the distance between the stop line and the front bumper of a_i, d_i^{brake} be the braking distance (to stop before the stop line) and t_i^{brake} be the braking time. Respectively, the DA's and platoon's ETA are given by:

$$t_i^{EA} = cruise\left(d_i^{line} - d_i^{brake}, a_i\right) + t_i^{brake} \tag{4.2}$$

$$t_{plt}^{EA} = cruise\left(d_i^{line}, a_{lead}\right) \tag{4.3}$$

In the case of the platoon's ETA (t_{plt}^{EA}), the preceding agent can be either an ordinary agent or a platoon member. The input a_{lead} in Equation (4.3) is the actual platoon's leader of a new platoon or the *extended* one assuming the DA joins with another preceding it.

However, in such heavy traffic scenarios, the calculation of ETA becomes challenging. The straightforward calculation outlined above might not suffice, as it does not account for the presence of preceding vehicles. Particularly in high-traffic demand scenarios, i.e., rush hour, the junction's throughput decreases, leading to extended queue lengths.

At the moment, the ETA calculates in a manner that no vehicles are waiting in front and aims to stop at the stop line. However, under high traffic loads, approaching DAs no longer manage to halt at the stop line; they are instead forced to come to a stop behind the existing queue. This dynamic necessitates the calculation of ETA to anticipate this altered behaviour.

To address this, we introduce the concept of the Estimated Time of Entry (ETE). In this context, let's denote an approaching DA as a_i and its preceding DA as a_{i-1}. Assuming a_{i-1} has fully reached the stop bar, this implies that a_i cannot immediately reach the stop bar and must wait until a_{i-1} has fully entered the junction area (i.e., the entire vehicle body crosses the stop line). The ETE, therefore, serves as an indicator of whether the conventional ETA calculation can be applied or if adjustments are necessary.

Furthermore, to facilitate this challenging ETA calculation, we introduce a recursive function that utilises the ETA of the preceding vehicle (refer to Algorithm 1). This recursive approach allows us to adaptively determine the ETA in scenarios where the queue impacts the stopping behaviour of approaching DAs.

Algorithm 1 Function $getETA(a_i)$. This function calculates the estimated time of arrival of agent a_i while recursively taking its preceding vehicles into consideration.

Require: $|A^t| > 0,\ a_i \in A^t$

1: $mingap =$ minimum vehicle gap
2: **if** Reservation$_i$ $\neq$ None **then**
3: $\quad t_i^{EA} = t_i^R$
4: **else if** $i \in keys(allETA)$ **then**
5: $\quad t_i^{EA} = allETA[i]$
6: **else if** $a_{i-1} \neq$ None **then**
7: $\quad t_i^{EA} = cruise\left(d_i^{line} - d_i^{brake}, a_i\right) + t_i^{brake}$
8: $\quad t_{i-1}^{EE} = getETA\left(a_{i-1}\right) + cruise\left(l_{i-1} + mingap, a_{i-1}\right)$
9: $\quad$ **if** $t_i^{EA} < t_{i-1}^{EE}$ **then**
10: $\quad\quad t_{i-1}^{EA} = getETA\left(a_{i-1}\right)$
11: $\quad\quad v^{max} = min(v_{i-1}^{max}, v_i^{max})$
12: $\quad\quad t_i^{follow} = cruise\left(l_{i-1} + mingap - d_i^{brake}, v^{max}, a_i\right) + t_i^{brake}$
13: $\quad\quad t_i^{wait} = conflict\_resolution(p_i, a_i)$
14: $\quad\quad t_i^{EA} = t_{i-1}^{EA} + t_i^{follow} + t_i^{wait}$
15: $\quad$ **end if**
16: **else**
17: $\quad t_i^{EA} = cruise\left(d_i^{line} - d^{brake}, a_i\right) + t_i^{brake}$
18: **end if**
19:
20: **if** $i \notin keys(allETA)$ **then**
21: $\quad allETA[i] = t_i^{EA}$
22: **end if**
23:
24: **Return** t_i^{EA}

Let us describe the operational details of Algorithm 1. This algorithm centres around a function named $getETA()$, which takes an agent, here referred to as a_i, as its parameter. The core objective of this function is to calculate the ETA of the given agent. Beginning at line 1, the variable $mingap$ is introduced to represent the minimum gap required between vehicles – set at 2.5 m. Lines 2-3 check whether a reservation exists for a_i. If a reservation is present, the ETA is set to the commencement time of its reservation, denoted as t_i^R. Notably, lines 4-5 and 20-22 feature the variable named $allETA$, which stores computed ETA values for all vehicles; its usage will be elaborated upon shortly.

The core of this function is presented in lines 6-18, where the condition at line 6 examines the presence of a vehicle in front of the agent. Let us denote the preceding vehicle of a_i as a_{i-1}. Lines 7-9 examine the scenario where a_i is estimated to reach the stop bar before the preceding vehicle leaves its position. Line 7 computes the time at which a_i will reach the stop bar, while line 8 calculates the time the preceding vehicle has fully entered the junction – this is referred to as ETE, with l_{i-1} representing the length of the preceding vehicle. The ETE of a_{i-1} is determined by extrapolating from its ETA point (1^{st} term) and accounting for its progression into the junction area (2^{nd} term). If the

spot isn't occupied by the preceding vehicle while a_i approaches the stop bar (line 9), a_i's ETA will result from the calculation in line 7, similar to the equation detailed in Eq. 4.2.

However, if the preceding vehicle does obstruct the path as a_i approaches, the algorithm continues to line 10. In this scenario, a_i has to follow its preceding vehicle at a restrained pace approaching the stop bar and await for available time slots to enter the junction. Line 10 retrieves the ETA of the preceding vehicle, specifying the moment when a_i can initiate its movement. Line 11 determines the maximum following speed for a_i, ensuring it remains compliant with the speed of the preceding vehicle and its own capabilities. The subsequent calculation, executed at line 12, quantifies the time span for which a_i must trail the preceding vehicle, which also accounts for braking time. The waiting time for available time slots is computed through the conflict resolution function. Lastly, line 14 sums the calculated values, representing a_i's ETA that incorporates both the movement behind the preceding vehicle and the time spent waiting for reservations."

Notably, the recursive nature of this function can be seen through its self-calls on lines 8 and 10. These recursive calls address the situation involving multiple preceding vehicles. Theoretically, this function will recurrently call itself until it reaches the foremost vehicle on the lane, eventually returning this vehicle's ETA, either from its reservation or estimation. This recursive structure ensures that the initial call to $getETA()$ will be calculated upon every vehicle positioned ahead of a_i. Additionally, it is important to note that all $cruise()$ used in this algorithm are calculated based on the assumption that the given agent is stationary at 0 m/s, awaiting its turn at the stop bar.

Furthermore, the algorithm utilises conflict resolution at line 13 to determine the waiting time required by the approaching DA. Throughout this process, the IMA maintains a pseudo reservation map, a duplicate of the current reservation map, where reservation time slots are recorded upon each execution of conflict resolution. This secures pseudo time slots for calculating the ETA of vehicles, preventing redundant reservations from occurring due to previous calls. Importantly, the pseudo reservation map serves exclusively within the platoon formation algorithm, distinct from the reservation map in the resource reservation mechanism. Meanwhile, the variable $allETA$, featured in lines 4-5 and 20-21, logs the computed ETA values of all vehicles, effectively eliminating the need for redundant core calculations. This function can efficiently return ETAs from previous calculations accessed via $allETA$."

Furthermore, the variable $allETA$ used in lines 4-5 and 20-21 is to record all the calculated vehicles' ETA, which is to avoid repeating the core calculation of the function.

In cases where the $getETA()$ function receives a member of a platoon as an argument, as occurs when the preceding vehicle a_{i-1} belongs to a platoon, special handling is employed. This involves treating a_{i-1} as a complete platoon, considering the ETA of

the platoon leader (a_{lead}) along with cruising time adjusted by the platoon's length (l_{plt}). The calculations on lines 8 and 10-13 are adjusted as follows:

$$t_{i-1}^{EE} = getETA\left(a_{lead}\right) + cruise\left(l_{plt} + mingap, a_{lead}\right) \tag{4.4}$$

$$t_{i-1}^{EA} = getETA\left(a_{lead}\right) \tag{4.5}$$

$$v^{max} = min(v_{lead}^{max}, v_i^{max}) \tag{4.6}$$

$$t_i^{follow} = cruise\left(l_{plt} + mingap - d_{lead}^{brake}, v^{max}, a_{lead}\right) + t_{lead}^{brake} \tag{4.7}$$

$$t_i^{wait} = conflict\_resolution(p_{lead}, a_{lead}) \tag{4.8}$$

Hence, this algorithm can compute the DAs' ETA of both standalone vehicles and platoon members.

Lastly, the algorithm accounts for situations where a_i joins its preceding vehicles to extend or establish a platoon, a scenario denoted by t_{plt}^{EA} (see Eq. 4.3). Under such circumstances, t_{plt}^{EA} is computed using:

$$t_{plt}^{EA} = getETA(a_{lead}) \tag{4.9}$$

where a_{lead} here represents the leader of the extended or newly formed platoon. The forthcoming explanation will describe the calculation of ETC.

ECT: Let us define d_{line,a_i}^{trg} as the distance between the target lane and the stop line of a_i, and $d_{line,a_{lead}}^{trg}$ as the corresponding distance for a_{lead}. Additionally, let $l_{plt} = \sum_{j=1}^{N-1} h_j + l_N$ represent the total length of the platoon, where h_j denotes the headway distance between the front bumpers of the j-th and $(j+1)$-th vehicles, and l_N signifies the length of the last DA in the platoon of size N. The ECT for both individual DAs and the platoon are calculated as follows:

$$t_i^{EC} = t_i^{EA} + cruise\left(l_i + d_{line,a_i}^{trg}, a_i\right) \tag{4.10}$$

$$t_{plt}^{EC} = t_{plt}^{EA} + cruise\left(l_{plt} + d_{line,a_{lead}}^{trg}, a_{lead}\right) \tag{4.11}$$

Lastly, the platoon's reduction in waiting time, λ_{plt}, is given by:

$$\lambda_{plt} = t_i^{EC} - t_{plt}^{EC} \tag{4.12}$$

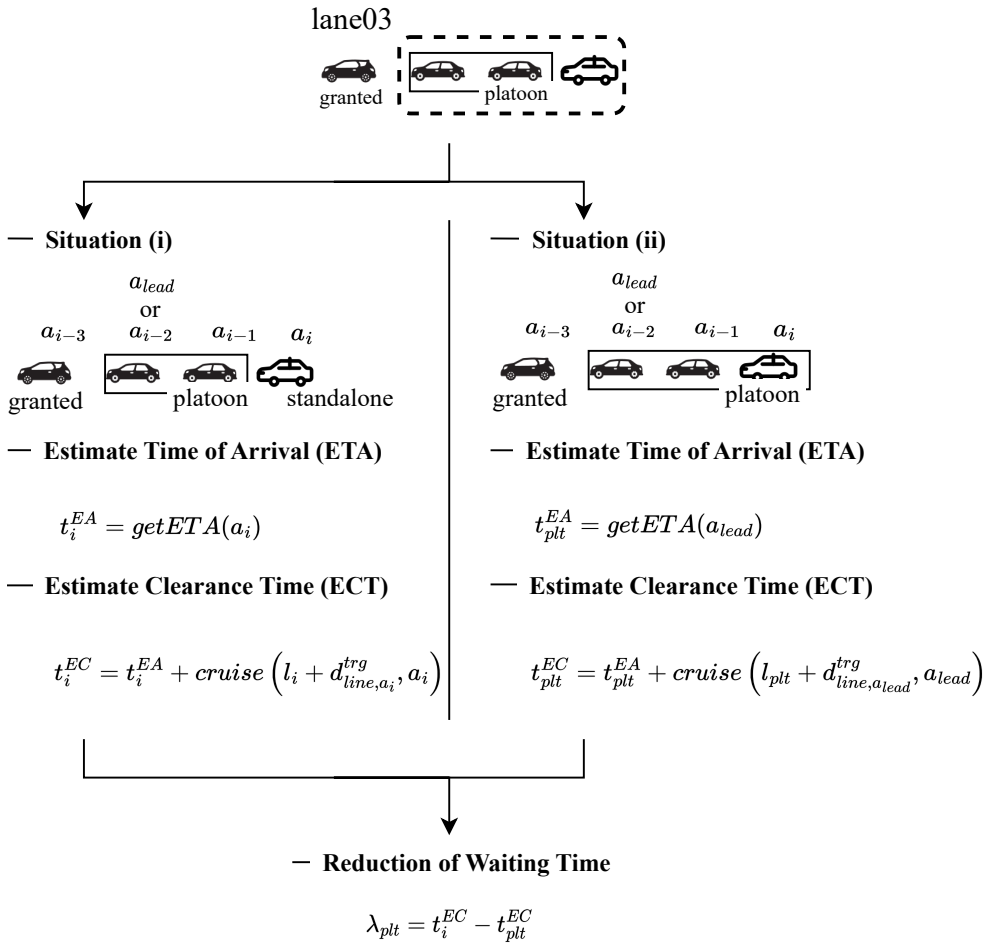

FIGURE 4.2: This illustrates the two possible behaviours for a_i, either being a standalone or a platoon member. This also shows example usages of various equations in the process of calculating the reduction of waiting time.

In conclusion, the calculation of the reduction of waiting time can be summarized through the following example scenario depicted in Figure 4.2, which corresponds to the lane03 context from Figure 4.1. Within this illustration, two potential behaviours can be suggested for a_i: either it operates as a standalone DA or it joins its preceding vehicles to form an extended platoon. Each scenario results in a distinct ETC for a_i. The difference between the ETC timings in these two situations defines the reduction in waiting time that a_i would experience.

With these calculations established, we can now proceed to assess the impact of a platoon in terms of an increase in waiting times for the rest of the junction.

4.2.2 Calculating the Increase in Waiting Time

In this section, we consider a situation in which a_i joins with its preceding vehicles as a platoon. Due to the largely reserved space and time slots of the platoon, the waiting time can gradually build up on multiple DAs more than the situation that a_i remains as a standalone. Here, we introduce a method to calculate the waiting time across all DAs.

By projecting the platoon's path and other DAs' paths on the junction, the conflict points/areas between the platoon's path and the DAs' paths can be located. Assuming the platoon is crossing, other DAs have to wait for a certain amount of time until these conflict areas become unoccupied, which means more *cost* to the platoon formation. We denote these conflict areas between the platoon and the DAs by placing virtual circles with a predefined diameter (one lane's width) on these intersecting points (see Fig. 4.3).

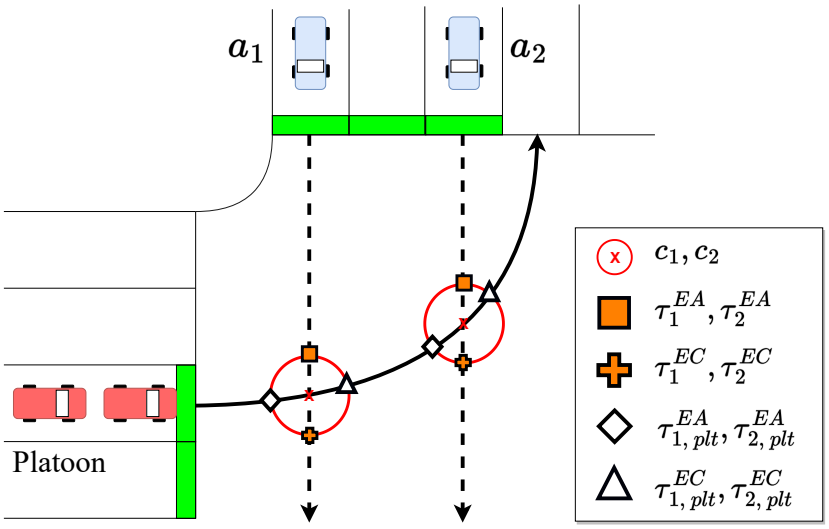

FIGURE 4.3: The example situation when the platoon's path (solid line) cuts through several DAs' paths (dashed lines); the vehicles on the left represent the platoon, while the rest are standalone DAs, and the circles represent the conflict areas between the platoon and DAs.

Given one particular circle, we can specify two occupancy time periods generated by both the platoon and a DA. The overlap between these two time periods represents the increased waiting time of this DA. However, one platoon's path usually overlaps with multiple DAs, causing a cumulative increase in waiting time across the junction. We next explain how to calculate the increase from multiple DAs using the ETA and ECT.

Let $A_{ovp} \subset A_t$ be a set of DAs having paths that overlap with the platoon's path, $A_{ovp} = \{a_1 ... a_M\}$. Each overlapping DA is denoted as $a_m \in A_{ovp}$, while c_m denotes the circle area at the point where the platoon's path and the a_m's path overlap. Additionally, d^m_{line,a_i} and $d^m_{line,a_{lead}}$ is the distance from the edge of c_m to the stop line with respect to a_i and a_{lead}. For a better understanding of terms representation, see Fig. 4.3 accordingly.

Then, the DA a_m's ETA on c_m and the platoon's ETA on c_m are calculated as follows:

$$\tau_m^{EA} = t_m^{EA} + cruise\left(d^m_{line,a_i}, a_m\right) \tag{4.13}$$

$$\tau_{m,\,plt}^{EA} = t_{plt}^{EA} + cruise\left(d^m_{line,a_{lead}}, a_{lead}\right) \tag{4.14}$$

Similarly, the DA a_m's ECT on c_m and the platoon's ECT on c_m are calculated as follows:

$$\tau_m^{EC} = \tau_m^{EA} + cruise\left(2r_m + l_m, a_m\right) \tag{4.15}$$

$$\tau_{m,\,plt}^{EC} = \tau_{m,\,plt}^{EA} + cruise\left(2r_m + l_{plt}, a_{lead}\right) \tag{4.16}$$

Here, r_m is the radius of c_m, equal to half of the lane's width. Any DA that has τ_m^{EA} (ETA on c_m's) overlapping with $\left(\tau_{m,\,plt}^{EA}, \tau_{m,\,plt}^{EC}\right)$ interval, in other words, is about to overlap with the platoon's time slot. They will receive some increase in waiting time, δ_m, that can be calculated as:

$$\delta_m = max(0, \tau_{m,\,plt}^{EC} - \tau_m^{EA}) \tag{4.17}$$

If τ_m^{EA} does not fall in this interval, δ_m will be 0 rather than a negative value. This means no extra *cost* caused by the DA a_m. Now, as a platoon's path can overlap with multiple DAs, all the DAs in A_{ovp} need to be considered and calculate the cumulative waiting time caused by the platoon, T_{plt}, which is equal to:

$$T_{plt} = \sum_{m=1}^{M} \delta_m \tag{4.18}$$

Consequently, having the reduction of waiting time and the cumulative increase in waiting time computed, we can calculate the *cost* efficiency of forming a platoon, β, as follows:

$$\beta = \lambda_{plt} - T_{plt} \tag{4.19}$$

If β is positive, the platoon will be formed.

In summary, let us consider the lane03 example from Fig.4.2; our approach quantifies the advantage gained by an individual DA, denoted as a_i, upon joining a platoon. This advantage is measured in terms of the reduction in waiting time λ_{plt}. Additionally, we assess the impact of a_i joining the platoon on other DAs within the junction, quantifying the increase in waiting time T_{plt} (see Section 4.2.2). By contrasting these two factors determining the cost efficiency β, we evaluate whether a_i's inclusion in the platoon is justified. Specifically, if a_i has the potential to decrease its travel time without imposing a significant burden on other DAs, the IMA grants approval for a_i to be labelled as a

platoon member with $status_i = follower$. In scenarios where the join of a_i forms a new platoon, similar to lane01 and lane04 in Fig. 4.1, its preceding vehicle (a_{i-1}) will be designated as a platoon leader ($status_{i-1} = leader$).

Note that β can be a negative value, representing an increase in overall waiting time. This situation usually occurs when the DA is about to join a platoon but causes more delays on the other DAs, i.e. more externalities, while reducing only a fair amount of stop-and-go movements. In this case, a_i will remain as a standalone ($status_i = null$).

Furthermore, due to the unique vehicle labelling scheme incorporating platoon formation within our system, which impacts the driving behaviour of DAs, we have enhanced the DA protocol to accommodate and support these various labelling cases.

4.3 Driver Agents Protocols

This section details the protocols and operational procedures that DAs must follow within our system. We begin with addressing the speed dynamics for DAs to support platooning to drive among standalone counterparts. Subsequently, we introduce conflict resolution that enables platoons to secure their time slots, thereby ensuring safe navigation through the junction.

4.3.1 Speed Constraint

The behaviour, especially the speed, of DAs varies based on their designated label $status_i$. The operational guidelines for *leader* DAs and standalone DAs are relatively straightforward. They aim to achieve maximum speed while adhering to safety constraints, including braking, responding to traffic signals, and maintaining a minimum vehicle gap. Typically, the leader and standalone DAs behave as described in Section 3.2.2.

On the other hand, *follower* DAs require the consideration of additional parameters during their manoeuvre. Even though followers synchronize their movements as a collective unit, their speeds cannot surpass the values dictated by the car-following model maintaining safety measures.

Specifically, the speed of any follower at a given time t can be defined as follows:

$$v_n^t = min(v_{n-1}^{t-1} + \alpha^{max}, v^{cf}) \tag{4.20}$$

In this equation, the first term represents the speed at which DAs can accelerate and follow their preceding vehicle. While the second term v^{cf} indicates the speed of the

car-following model Krauß et al. (1997) as employed in SUMO. This model constrains follower DAs from excessively accelerating, ensuring they do not violate the safety constraint of maintaining a minimum vehicle gap.

Here, $n \in \{2, \ldots, N\}$ denotes the position of a follower DA in a platoon with a size of $N|N \geq 2$, where the first position is the platoon leader with the representation of speed as v_1^t. With this speed constraint for follower vehicles, they do not need to perform any resource reservation mechanism since their primary task is to follow their preceding vehicle. Therefore, only standalone and the leader will handle the resource reservation, responsible for their own reservation and their member if any.

Furthermore, our system has gone through significant developments, with platoons now integrated among the majority of standalone DAs. This integration has posed new challenges for conflict resolution to go beyond the capabilities of the previously defined mechanisms. As a result, we have introduced substantial enhancements to our conflict resolution to effectively handle and address the complexities introduced by platoons within the system.

4.3.2 Conflict Resolution for Platoons

The ordinary conflict resolution mechanism, as outlined in Section 3.3.3, does not readily support platoon reservations due to the larger space and time slot required by platoons. The main challenge is to include the space platoons required in the conflict resolution algorithm.

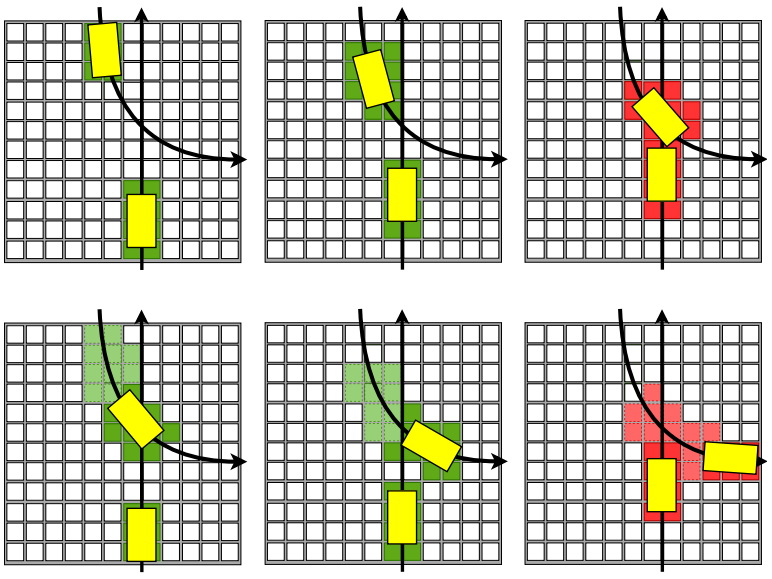

FIGURE 4.4: This illustrates a conflict between two paths, which is detectable through the overlapped desired cells. The upper part shows the situation where both DAs are standalone agents, while the lower part shows the situation where one DA is the platoon's leader that leaves its trace behind.

When it comes to platoon reservations, the number of desired cells is usually higher compared to standalone reservations. This is to allocate sufficient space for the platoon's leader and follower DAs. Therefore, we need to identify how much space each platoon needs for their reservation and interpret them into the desired cells. Note that the data structure of the desired cells remains the same as shown in Code 3.2.

However, the previous method, which projects vehicles' bodies onto junction cells to identify desired cells, faces limitations when applied to platooning scenarios. Due to the fact that multiple vehicles are driving in a uniform group while maintaining safety gaps when their bodies are projected upon cells, the existing method fails to account for these gaps, leaving them unaddressed The challenge here is to enhance the conflict resolution to accurately identify occupying platoon space and precisely capture their desired cells.

To address this challenge, we introduce a new approach by treating the platoon leader as a vehicle that leaves a trace behind, mimicking the movements of its followers who occupy additional space. It's important to note that this trace, including the leader's length, must not exceed a predefined platoon length (l_{plt}). If it does, the trace is truncated accordingly, as depicted in Fig. 4.4. In particular, we define a function named *getCurveSegments*() designed to calculate the occupied space of the platoon or the moving traces at each time step along the platoon's predicted path. This function is provided in Algorithm 2.

Algorithm 2 Function *getCruveSegments*(a_{lead}). This identifies curve segments or traces of the platoon at each timing given the leader agent a_{lead}.

1: p_{lead} = predicted path of the platoon
2: traj_curve = formCurve(p_{lead})
3: curve_segments = { } ▷ Dictionay structure
4: width = max($w_n, \ldots, w_N$)
5: **for** t in keys(p_{lead}) **do**
6: bumper_pos = $p_{lead}[t]$
7: segment = truncate(traj_curve, bumper_pos, l_{plt})
8: dilated_segment = segment.buffer(width)
9: curve_segments[t] = dilated_segment
10: **end for**
11: **Return** curve_segments

Within the algorithm, line 2 utilises a function named formCurve() creates a bezier curve based on the predicted path of the platoon, representing its trajectory. Line 3 initialises a parameter to store curve segments or traces of the platoon. The width parameter is specified by the maximum width of platoon members, determining the actual width required by the platoon. Line 5 loops to compute curve segments starting from the commencing time of path p_{lead}. The keys() function retrieves a set of timestamps from vector p_{lead}. The parameter bumper_pos indicates the front bumper position of a_{lead} at time t, acquired through $p_{lead}[t]$. Line 7 employs the truncate() function to trim

the traj_curve, starting from the bumber_pos position and extending for a distance of l_{plt}. The buffer function dilates the curve segment to the left and right, covering the size of the platoon as specified by the width parameter. This dilated segment denotes the space occupied by the platoon at time t. Lastly, this dilated segment is stored in the curve_segments variable, with t serving as the key value.

In essence, the $getCurveSegments()$ function divides the platoon trajectory into multiple curve segments, each with a defined thickness, representing the occupied space of the platoon at each time step within the junction area. Then, the curve_segments variable behaves as large vehicles' bodies are projected on the junction cells to obtain the `desired_cells`. With this enhanced development, our conflict resolution mechanism can now accurately quantify the desired cells for both standalone and leader vehicles.

Similar to Section 3.3.3, after acquiring the desired cells, the leader continues to determine the conflicts, proceeds to identify the waiting time and shifts the commencing time of its predicted path accordingly. Subsequently, the platoon leader proceeds to submit a request to the IMA in accordance with the flowchart outlined in Fig. 3.8.

4.4 Empirical Evaluation

To evaluate the performance of our proposed method, we use the open-source traffic simulator SUMO version 1.6.0 Krajzewicz et al. (2002). With the client-server-based Traffic Control Interface (TraCI) Wegener et al. (2008) available in SUMO, external applications can be used to control simulated vehicles at run time. With this feature, FCFS and our method can be implemented on the simulated junctions enabling coordination between DAs and IMA. Based on the simulation outputs, we compare our proposed method against FCFS in terms of two average values: throughput and travel time.

4.4.1 Simulation Setup

We set up the simulation with a four-approach junction replicating the junction in Manhattan similar to the one in Chapter 3 (see Fig. 3.3). v^{max} is 11.18 m/s (≈40 km/hr), and α^{max} is 2.5 m/s^2. Vehicle length is 5 m with a minimum gap of 2.5 m. Simulation time is one hour with 0.25 seconds of simulation time steps. We examined eight different traffic volumes starting from 7,000 veh/hr to 15,000 veh/hr (increasing by 1,000 veh/hr), and they were generated according to the share in 3.4. The purpose of increasing the traffic volume is to see the performance of each method under different traffic scenarios, from light to heavy traffic. Each traffic volume was simulated for 20 runs to ensure the results' accuracy.

4.4.2 Traffic metrics

In evaluating our system, we employ two essential traffic metrics: travel time (measured in seconds) and throughput (measured in *vehicles/hour* or *veh/hr*).

In contrast to the waiting time metric utilised in Chapter 3, we have chosen to focus on travel time as a more comprehensive evaluation criterion. Travel time captures the duration vehicles take to travel from one point to another, reflecting their journey from the entrance to the exit of the inbound lanes before accessing the junction. Particularly, this metric incorporates all potential delays (on average) from lane-changing, queuing, and waiting for reservations, unaffected by the vehicles' stuttering caused by FCFS (see Section 3.7). By considering these factors, this metric offers a meaningful perspective on our method's performance on a per-vehicle basis.

Furthermore, the throughput metric assesses the junction's efficiency in swiftly resolving and releasing vehicles within a defined interval (in this case one hour). This measurement provides valuable insights into the method's performance from an infrastructure point of view, indicating how effectively the junction manages the flow of vehicles in a constrained timeframe.

4.4.3 Travel Time Results

This performance assessment using this metric will be devided into two parts. The first part starts from 7,000 to 11,000 veh/hr traffic volume as it shows the climbing of the travel times as the scenarios change. The second part is from 12,000 to 15,000 veh/hr, where the travel time results remain stable. The complete results of travel time can be seen in Fig. 4.5.

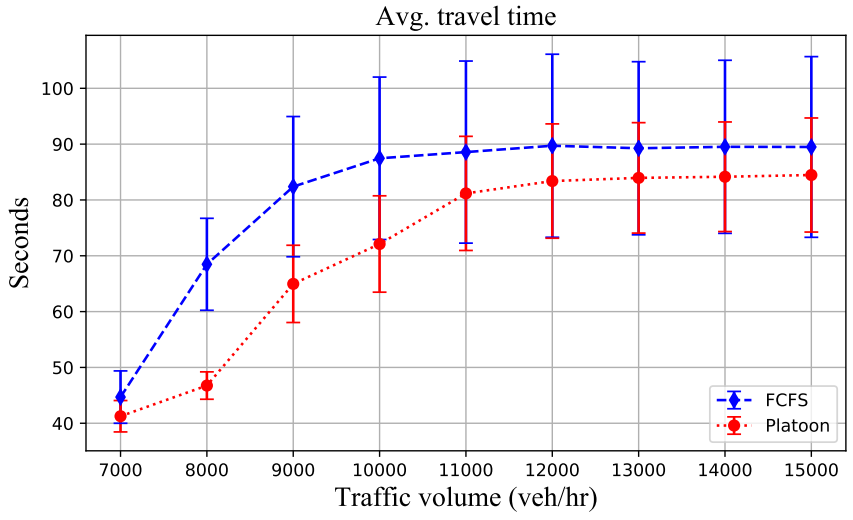

FIGURE 4.5: Average travel time results compared between FCFS and our method. Error bars show 95% confidence intervals.

In the first part of our results, DAs operating with our dynamic platoon formation algorithm exhibit a significant performance advantage over the FCFS approach. In particular, at a traffic volume of 7,000 veh/hr, our platooning strategy initially seems to display a modest performance gap of 7% compared to FCFS. This seemingly small difference might be attributed to the junction's ability to maintain a relatively smooth traffic flow, keeping queuing lengths short and reducing the opportunities for platooning.

On the other hand, when traffic volume reaches 8,000 veh/hr, the potential benefits of platooning become more evident. Longer queuing lengths create more opportunities for platoons to form. Under these conditions, our method can substantially decrease the average travel time by up to 31% compared to the FCFS baseline. However, as traffic volumes continue to rise, the gap in performance narrows. Despite the junction growing busier and queuing lengths expanding further, our platooning method fails to achieve additional reductions in travel time. This is due to our platoon formation algorithm being constrained by the externalities associated with each platoon's presence.

For the second part of these results, we observe a flattening trend in average travel time as traffic volumes reach their peak. This phenomenon suggests that the junction has approached its traffic capacity limit. The consistent travel time indicates that vehicles on the inbound lanes are moving at a constant speed due to limited available space for acceleration. This implies that a number of vehicles might be unable to enter the system at their designated times, mainly due to space availability. Despite these challenging traffic conditions, our platooning method continues to show a performance advantage, reducing the average travel time by up to 6% in comparison to the FCFS counterpart, even under the worst case of 15,000 veh/hr.

In conclusion, our performance assessment of the dynamic platoon formation algorithm reveals useful insights into its efficiency across traffic volumes. In the first half, our method presents a substantial performance gain over FCFS approach, reducing the average travel time by up to 31% under optimal conditions. While its performance advantage decreases under extreme congestion, it still maintains the reduction of travel time by up to 6% over FCFS. Overall, the dynamic platoon formation algorithm proves its worth in enhancing junction traffic performance, in this case, vehicles' travel time, especially when queuing lengths and opportunities for platoon formation are favourable.

4.4.4 Throughput Results

In this section, we proceed to evaluate the junction's performance from an infrastructure standpoint, specifically focusing on throughput results and conducting a deliberate comparison among various junction control methods.

		Traffic volume (veh/hr)								
		7000	**8000**	**9000**	**10000**	**11000**	**12000**	**13000**	**14000**	**15000**
Arrived vehicles	Platoon	74.40%	73.89%	71.00%	65.89%	60.48%	55.55%	51.36%	47.87%	44.52%
	FCFS	74.27%	71.62%	65.23%	59.28%	53.98%	49.14%	45.69%	42.65%	39.67%
Platoon vs. FCFS	Travel time (reduce)	7.68%	31.72%	21.15%	17.55%	8.35%	7.06%	5.95%	5.99%	5.62%
	Throughput (increase)	0.17%	3.17%	8.83%	11.15%	12.03%	13.03%	12.41%	12.25%	12.22%

TABLE 4.1: The top half compares the percentage of all vehicles arriving at their destination by the end of the simulation between our method and FCFS. The bottom half shows the average travel time reduction and throughput increase compared to FCFS of those vehicles that arrived at their destination. Results are averaged over 20 runs.

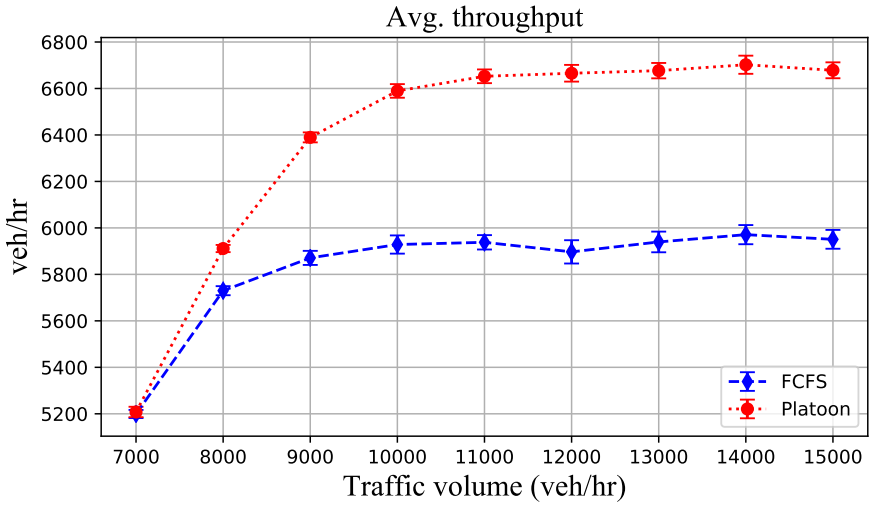

FIGURE 4.6: Average throughput results for FCFS and our method. Error bars show 95% confidential interval

The throughput results, depicted in Fig. 4.6, show a distinct trend compared to the previous travel time assessments. This traffic metric highlights the constant advantages of our method over the FCFS method. Although the performance gap between the two methods remains relatively small at 7,000 veh/hr scenarios, it becomes increasingly evident as traffic volumes rise.

Interestingly, the optimal conditions where DAs experience the fastest travel times, such as at 8,000 veh/hr, do not necessarily align with the highest performance improvements in terms of throughput. Even though travel times are significantly reduced at this traffic volume, the junction's overall performance improvement compared to FCFS is not maximised under these conditions.

Furthermore, our method demonstrates its capability to enhance throughput, achieving a peak value of approximately 6,700 veh/hr, representing a substantial increase of around 12% compared to FCFS. However, as traffic volumes continue to escalate, the trend in the results begins to flatten after reaching a traffic volume of 12,000 veh/hr. These findings are consistent with our observations in the travel time results, suggesting that the junction has reached its maximum operational capacity.

In this context, a crucial consequence becomes evident. As traffic volumes increase, not all generated vehicles can reach their destinations. This limitation arises due to the junction's limited traffic capacity, as indicated in Table 4.1. Practically, this means that some vehicles remain stranded outside the simulation environment. If adjacent junctions exist, these stranded vehicles can potentially trigger traffic overflow or spill-out issues in those areas. It's important to note that our assessment here focuses solely on the traffic elements within this individual junction. However, the real-world implications of such an issue can accumulate and have ripple effects in surrounding areas.

In summary, our dynamic platoon formation method shows substantial benefits in terms of throughput, especially at higher traffic volumes, where it outperforms FCFS. While travel time improvements are notable, they become constrained as traffic volumes increase, indicating the presence of external limitations. Nonetheless, our method exhibits a performance advantage even under challenging traffic conditions.

4.5 Discussion & Limitations

The results emphasise a consistent difference between FCFS and our method, which persists even as traffic volumes reach notably high levels, approximately 12,000 to 13,000 veh/hr. This suggests that the single junction scenarios have effectively reached their capacity limits, leaving no room to accommodate additional vehicles, especially in the context of such high traffic conditions.

From the results, it can be seen that the difference between FCFS and our method remains stable after reaching a considerably high traffic flow (approximately 12,000 or 13,000 veh/hr). It appears that the single junction scenarios have reached their limit as they have no available space to spawn more vehicles, obviously in such high traffic. However, a lighter one could certainly lead to a little out-performance gap or even similar results. For instance, in situation 7,000 veh/hr, there is a very tiny performance difference between the two methods since platoons have less opportunity to even form since the traffic is too light (see Fig. 4.6). The change in the environment setting can have a huge influence on the junction results, not only the amount of space but also the possible externalities, e.g. expanding or decreasing junction scale. In a bigger junction setting, it is still unclear whether the out-performance gap between FCFS and platoon will be larger (due to extended traffic space) or smaller (constrained by high externalities). To properly demonstrate the platooning in more detail, a dynamic and more realistic environment would be more suitable to evaluate our platooning method.

Additionally, the limitations of the junction's capacity hinder our ability to perform a deep analysis on the second half of the results, as they remain stable. Theoretically, as traffic volumes increase, delays should also increase, at least at a constant rate. However, under this phenomenon, the junction reaches its capacity limit, causing vehicles to

be stranded outside the simulated junction, affecting traffic beyond the simulated environment. At the moment, SUMO's traffic metrics fail to capture such effects. Nevertheless, it is important not to overlook these stranded vehicles, as they exist physically. This situation resembles one of the traffic flow issues called overflow traffic, where vehicles essentially spill over into nearby junctions. This issue introduces unique challenges to the evaluation process and necessitates us to expand the simulated environment, allowing SUMO to broaden its scope of observation. Also, the expansion enables us to better assess the impact of overflow traffic.

Furthermore, another limitation of this work lies in a lack of data. Despite using the real-world junction as a benchmark, a realistic modelling environment is beyond the junction geometry. It needs to be detailed down to the traffic flow aspect. However, currently, the traffic generation is rather uniform, where lanes have a distinct but constant amount of traffic. This approach results in unrealistic traffic, with some vehicles travelling straight more often than turning, and vice versa. Even though we introduce a novel platooning method for junction management, using idealised settings cannot draw out the impact of platooning in a sophisticated manner. Moreover, the study's scale is limited to a single junction with identical vehicle types, which is not an accurate representation of the practical environment. For the simplicity of our study, many elements presented in the real world have been excluded, e.g., road usage, routing, variation of vehicle types, uneven vehicle speed, and asymmetrical junction configuration. The evaluation within this work only showcases our platoon's performance to a limited extent. To gain a more in-depth understanding, we need to consider the elements mentioned above.

More importantly, the realism of the traffic environment also affects platoons in terms of driving speed. Assuming that the environment has a variety of vehicle types, such as buses, taxis, private cars, etc., these vehicles tend to drive at their individual speed due to safety concerns and vehicle dynamics. For example, buses normally drive with an average speed of 20 - 35 km/hr in the city area. Whenever a platoon is formed by having a slow-moving vehicle as a leader, e.g. buses, the follower(s) cannot drive at their full speed as they have to follow the leader's speed. This can be referred to as an interruption, preventing the platoons from providing the most benefits. This issue is not apparent in the idealised situation of the benchmark junction that we used in this study. However, in a more realistic scenario, platoons will encounter a challenge of interruptions due to the varying speeds of vehicles, which will impact their efficiency.

Therefore, to explicitly evaluate the effectiveness of the platooning method, we consider expanding our study scale to cover a larger road network. To do so, a realistic dataset is utilised to model and calibrate our simulated environment recreating a highly-realistic traffic corridor. This allows us to extensively investigate the impact of our platooning method and other junction controls.

Chapter 5

Highly Realistic Multi-junction Environments

In this chapter, we describe the realistic environment modelling to evaluate our study in a more sophisticated manner with large-scale road network, mainly addressing Research Challenge 3 & 5 To achieve this, we utilise a real-world dataset that provides vehicle movement on a corridor in Athen, Greece, containing multiple junctions. The dataset provides valuable information on practical infrastructure and environments, such as road usage, the share between vehicle types, default red-green timings, and traffic flow/routing. We leverage this information to calibrate our simulation and create scenarios closely resembling the real world. Evaluation is performed using conventional traffic lights, as well as decentralised junction management and our platooning method, which are described in Chapter 3 & 4, respectively.

The structure of this chapter is as follows. Firstly, we provide an explanation of the raw dataset, along with its associated issues. Secondly, we outline the process of calibrating the simulated environment in terms of junctions, vehicles, and traffic demand. Thirdly, we evaluate our platoon-based junction management method against other non-platoon-based methods, namely conventional traffic control and FCFS. Lastly, we discuss the limitations encountered during this study.

5.1 Raw Dataset

This section describes the characteristic of the dataset: how it is recorded, what properties it has, and also the issues that come with the dataset. To begin with, the dataset is recorded by an array of drones flying over Athens that keep tracking vehicle (including motorcycles) movements in the form of snapshots. Each snapshot provides the latitude and longitude of specific vehicles as they enter and leave the recording area, meaning

that the taken routes, departure and destination road of each vehicle's trip can be determined. Moreover, other information about each vehicle is also included in the dataset, including tracking ID, type of vehicle, distance travelled (meters), and the vehicle's average speed (km/h). Initially, the dataset provides the record of ten drones covering most of the vehicle movements in Athens. However, for the system's simplicity and to save up the simulation runtime, only two out of ten focus areas are used, covering a corridor of eight junctions (see Fig. 5.1).

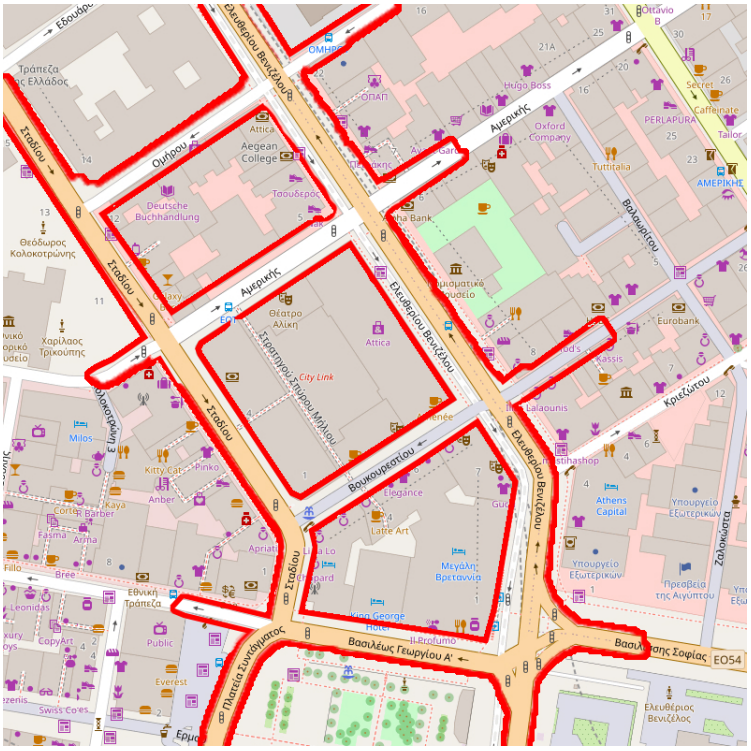

FIGURE 5.1: The corridor in Athens, Greece, used in this study, comprises eight junctions (within the solid enclosure).

Nevertheless, there are some issues with the trajectories of the vehicles in the recorded dataset, preventing us from inputting traffic demand into SUMO directly. The issues are described as follows. Firstly, static vehicles (or parking vehicles) and unconventional trajectories (e.g. motorcycles on the pedestrian path) are included as they are visible to the drones, causing several unidentified trajectories and errors on SUMO. Secondly, several tall buildings in the focus area create an issue of blind spots, causing a discontinuity in some vehicle trajectories/routings. Nevertheless, with the help of a traffic visualisation tool named Travia (Barmpounakis and Geroliminis, 2020), all of the vehicle trajectories can be mapped and visualised, allowing us to observe the actual movements of each vehicle accurately and see the disappearance of vehicles at certain spots. Note that Travia is an open-source tool written in python, and its application can be found here[1].

[1] https://github.com/tud-hri/travia

To solve these issues, the stationary vehicles are removed from the records by ensuring that they are actual parking vehicles (parked til the end of the record session). Then, for the unconventional motorcycle' trajectories, they are mapped to their closest road to ensure that all vehicles will not be missed out and drive on the road as they should. Lastly, in the case of blind spots, Travia is used to locate specific positions/roads where vehicles disappear and reappear. Any records of vehicle passing through the blind spots are filled with additional trajectory snapshots connecting the disappear and reappear positions assuming that vehicles are driving at a constant speed (from the point where they start to disappear). Thereby, the issues of unidentified trajectories and blind spots are resolved.

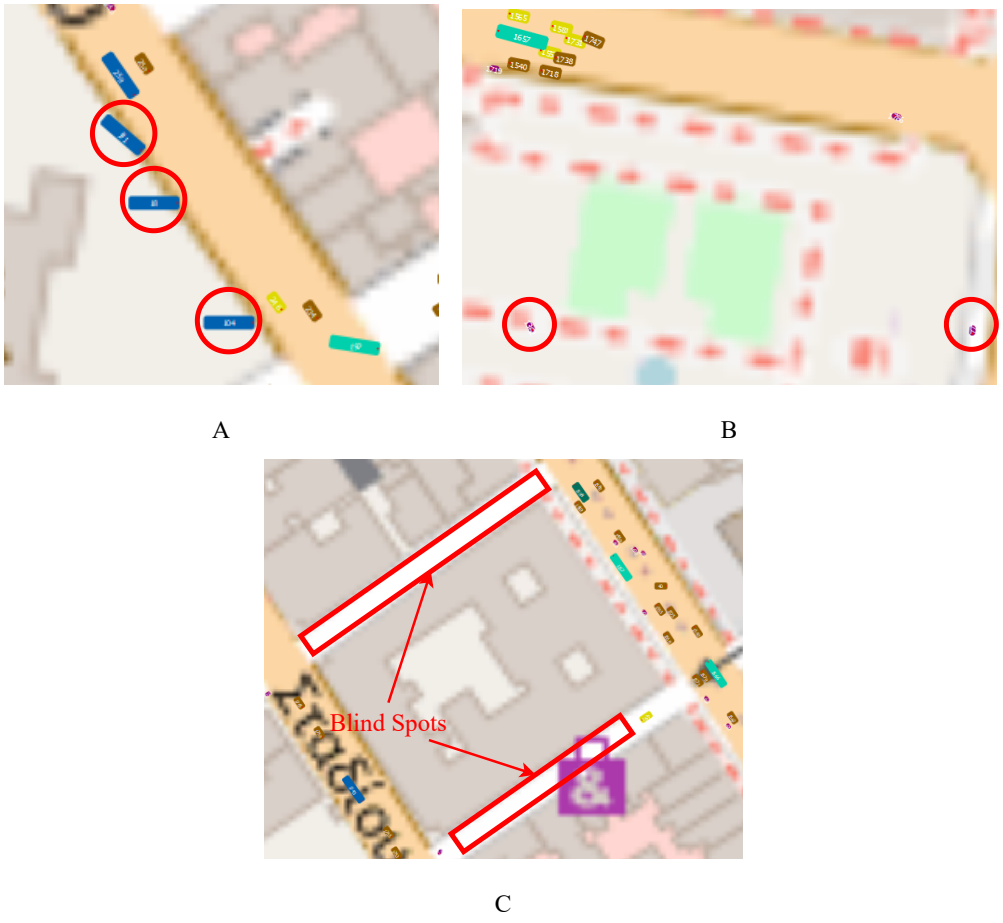

FIGURE 5.2: This figure shows the issues with the data (in the enclosure). In A, an example of static vehicles can be seen. Motorcycles having unconventional trajectories are shown in B (cutting through a park & driving on a footpath), and the locations of blind spots where vehicles can not be tracked are shown in C.

5.2 Ground Truth Traffic flow

After resolving the issues with the raw dataset, we proceed to transform this dataset into a format compatible with SUMO, enabling us to recreate the ground truth traffic

flow. In particular, the raw data provides all the individual vehicle trajectories, which we convert into a series of vehicle vectors. Each vector represents a generating vehicle, and each contains essential information including the vehicle's ID, its type, departure time and taken roads. Specifically, the vehicle vector is defined as:

$$Vehicle_i = < i,\ type,\ departure\_time,\ \{road_k, ..., road_K\} > \tag{5.1}$$

Here, i represents a unique ID, $type$ refers to its vehicle type, $departure\_time$ is a simulation time when this vehicle will be generated, and $\{road_k, ..., road_K\}$ defines the route vehicle taken. All of the vehicles that travel through this area are recorded in this form of vectors; thereby, the integrity of the flow can be maintained.

Consequently, this set of vectors can be input/transferred into the SUMO, generating and spawning the vehicles with the pre-defined route at a specific road and time. In this way, the ground truth traffic flow can be accurately recreated.

However, the question arises as to whether it is necessary to dissect the traffic data down to the individual junction level. Although focusing solely on individual junctions offers a reasonable scope, several reasons necessitate the expansion of our simulation to a macroscopic level.

5.3 Macroscopic Simulated Environment

In this section, we will provide a detailed explanation of our decision to simulate the entire corridor environment rather than simulating multiple junctions individually with connected vehicles' input and output. We have identified three primary reasons for this choice: model complexity, availability of the data and the presence of unique phenomena that manifest at the macroscopic level.

Simulating traffic at a macroscopic level reduces the complexity associated with designing a microscopic-level simulation. At the macroscopic level, we can model the traffic demands in terms of vehicle routing and flow, which provides a more comprehensive perspective. In contrast, at the microscopic or junction level, we would need to model specific details such as the number of vehicles generated on each road and the proportion of vehicles making turns versus going straight. By defining routing and flow, we can treat these numbers and proportions as representations of road users' intentions. For example, vehicles take turns at a junction because they want to follow a specific route. This simplification allows for a more manageable and efficient simulation design.

Furthermore, linking outbound and inbound vehicles between nearby junctions is not a straightforward task. Several factors must be carefully considered to establish connections and ensure vehicles travel together, forming a coherent vehicle trip. These factors

include departure time, leaving speed, lane selection, and available space. While it may be possible to implement such connections, it raises the question of whether it is more advantageous to simulate each individual junction separately and then piece together the results for analysis. The complexity of managing interconnections between vehicles and coordinating their movements raises practical challenges. Ensuring seamless transitions between junctions requires intricate synchronization and careful handling of traffic flow. The output of one simulation must be recorded in a chronological manner, logging the mentioned factors as vehicles leave so the output can be utilised as input for another junction. Attempting to implement the connections from scratch and manage them individually for every junction in the corridor would significantly increase the simulation's complexity.

Another factor influencing our decision is the availability of data. The provided data is extensive, and attempting to calibrate or interpret it in very small detail would be unwise for certain reasons. Simulating the junction individually means the ground truth data must be truncated into small pieces covering the traffic at each junction. The precise traffic distribution must be formed to maintain the ground truth demands, and it would introduce uniform traffic scenarios similar to the experiment setting in the previous chapter. Any traffic changes at the macroscopic level are constrained by the junction-level traffic settings, forcefully obeying its uniform setting.

Furthermore, the available data provides tracked movement of each vehicle from its departure to the destination, essentially representing the routing. By chunking the data into junction level it means dismantling any relationships between road usage and vehicle routing. Consider the scenario where vehicles only have knowledge of their designated trajectories at the junction level. In this case, vehicles enter the junction, continue on the outbound lane, and exit as quickly as possible without considering the optimal lane or appropriate speeds.

On the other hand, when vehicles know their complete routing, they leave a particular junction with a purpose. For instance, if a vehicle needs to turn right at the next junction, it will proactively switch to the right lane as soon as possible, as it knows this is the optimal lane for its route. Essentially, this decision to follow a lane-changing strategy is made well in advance. This routing-based approach presents advantages, particularly in scenarios involving long queues that require some give-ways to be able to change lanes. With routing information, vehicles can make smoother and more purposeful lane transitions, minimising interruptions and delays for other vehicles. However, when we simulate junctions distinctively, this advantage disappears. Particularly, vehicles only know their trajectories upon being input into the junction, and at that point, it is often too late to execute lane changes seamlessly.

Leveraging the routing information in our data ensures that transitions between junctions are realistic, purposeful, and smooth. Importantly, it prevents unnecessary navigation delays from being introduced into the simulation results.

Furthermore, the macroscopic level can reveal specific issues and phenomena that are unique to this broader scale of analysis. One of the phenomena is traffic oscillations. These oscillations refer to the propagation of congestion along a roadway, resulting in localised pockets of congestion and stop-and-go traffic patterns. In other words, it is wave pattern traffic. This wave-like traffic pattern is commonly seen in simulations employing traffic light control systems. More importantly, traffic oscillations can escalate, worsening the traffic situation and potentially resulting in gridlock scenarios.

One example is the limitation of the operational space, which we encountered while simulating a single junction (see Section4.5). In this scenario, it appears that the junction struggled to efficiently manage very high traffic inputs, leading to overflow traffic. This issue constrained the availability of inbound road space, resulting in vehicles being stranded outside the simulated area. Consequently, certain traffic metrics, such as waiting time and throughput, became flattened, hindering our ability to accurately assess the junction's true performance.

Furthermore, in the context of the single-junction setup, the outbound roads function as if they have infinite capacity, permitting vehicles to exit the junction as swiftly as possible without considering space constraints. However, in reality, the overflowed vehicles should occupy some traffic space at nearby junctions; the outbound roads behaving as infinite contradicts the actual impact of the issue. More importantly, when we simulate multiple junctions individually, we effectively introduce the potential for overflow traffic issues at every simulated junction. This creates enormous difficulties in assessing overall system performance.

A possible solution involves simulating instances of the junction and its neighbouring junctions simultaneously. With the linked vehicles' input and output, outbound vehicles from interconnected junctions cannot always exit their respective simulated environments because, at the same time, the inbound roads of others may be operating at full capacity. However, this approach is essentially comparable to the simulation of an entire corridor network.

In summary, our decision to simulate the corridor environment as a unified system is influenced by the desire to simplify model complexity, leverage available data effectively, and uncover unique phenomena that manifest at this broader scale of analysis. Even though simulating multiple junctions one by one and linking the incoming and outgoing vehicles to each other would be plausible, the inherent complexity modelling makes it a less practical approach. Instead, adopting the macroscopic-level simulation approach for the entire corridor is a more efficient and reasonable solution.

Next, we will explain how the traffic model is calibrated to produce a highly realistic corridor environment.

5.4 Calibrated Traffic Model

This section describes the traffic model used in this work, containing the junctions model, vehicles model, and traffic demand with randomness. In particular. we base our model similar to the traffic model in Chapter 4, which considers a road junction managed by an *Intersection Manager Agent* or IMA (junction manager). The IMA can grant its resource (time-slotted space in the junction) to each vehicle—*Driver Agent* or DA—to coordinate each vehicle's movement. We next describe how the junction, vehicles and traffic demands are modelled.

5.4.1 Junctions

While several studies in platooning, e.g., (Jin et al., 2013; Bashiri and Fleming, 2017), only consider ideal junction geometry, we consider more practical junctions and use the SUMO simulation tool to model these. Specifically, our study is modelled after a road network in Athens, Greece (Barmpounakis and Geroliminis (2020)), in which eight junctions are considered. The representation of the corridor can be seen in Fig. 5.3

In particular, junction geometry is another element that highly impacts the realism of the model. Even though SUMO can import a road network from OpenStreetMap[2] directly, information is partially inaccurate, e.g. the number of lanes per road, free-turn lanes, public transport lanes, and the possible driving direction of each lane. To rectify this, TraVia (Siebinga, 2021), a traffic data visualisation tool, allows us to reproduce the vehicle movements in pNEUMA and adjust the road corridor accordingly and practically. Specifically, some parts of the road are the local transport where only buses and taxis have the right to use them, see the grey roads in Fig. 5.3 & 5.4.

Moreover, as we consider the fixed-time traffic signal program, the red and green times or phases are extracted through TraVia replacing SUMO's pre-generated signal programs. The process can be done by observing when vehicles start to slow down (for red timing) and when vehicles start to speed up (for yellow and green timing). An example junction can be seen in Fig. 5.4, which is the most crowded junction in the area.

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

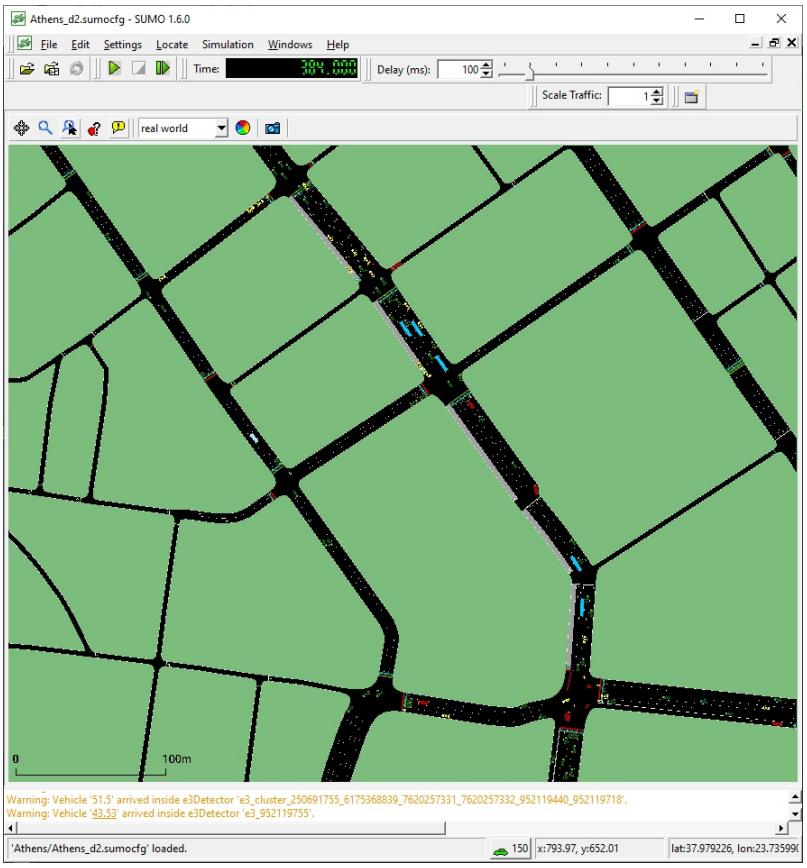

FIGURE 5.3: This image shows a SUMO-simulated corridor of eight junctions recreated after a practical setting in Athens, Greece. The roads for local transport are coloured in grey.

5.4.2 Vehicles

After the junction model is adjusted to replicate as close to the real world as possible, the vehicle model is realistically calibrated as well. The model of vehicles is similar to the model of the vehicle in 4.1.1 where an agent can be labelled with $status_i$ specifying three states: (i) leader, (ii) follower, (iii) null. However, we no longer assume that the vehicles are identical, unlike in Chapter 3 - 4, which means that they do not have similar parameters, such as length, speed, acceleration and deceleration rate. This static assumption is ideal and rarely seen in the real world.

As the real-world dataset is used, the vehicles are considered to have various types, i.e. heterogeneous. Some properties of vehicles are no longer constant values corresponding to their type. Specifically, with $A_t = \{a_1 ... a_I\}$, each $a_i \in A_t$ has its individual properties which are length l_i, $type_i$, maximum velocity v_i^{max} and accelerating rate α_i^{max}. For the width and length of the vehicles, we use the default vehicle dimension provided by SUMO[3]. For example, buses have a dimension of 12 x 2.5 m; taxis and private cars have a dimension of 5 x 1.8 m, etc.

[3] https://sumo.dlr.de/docs/Vehicle_Type_Parameter_Defaults.html

Moreover, to recreate the practical traffic flow environment similar to the ground-truth traffic, the dataset in Barmpounakis and Geroliminis (2020) is derived to specify a population of vehicle types, acceleration, and also a deviation in maximum speed. In particular, the maximum speeds are assigned using a normal distribution given minimum and maximum values of 95% spread (z=1.96). The full details of the vehicle properties can be seen in Table 5.1. Additionally, Fig. 5.4 shows a snapshot of different vehicle types in action where each one is coloured uniquely. Whenever a vehicle is newly generated SUMO randomly assigns its maximum speed that fluctuates from the average depending on the spread value. For instance, the majority of private cars have a maximum speed (v^{max}) of 12.09 m/s (the average value); the lowest maximum speed is 8.84 m/s (12.09 - 3.25 m/s); the highest maximum speed is 15.34 m/s (12.09 + 3.25 m/s), and vice versa for the other vehicle types. The"share" value signifies the percentage of each vehicle type within one simulation run. As the total number of vehicles changes, the count of each vehicle type dynamically adjusts accordingly.

Type	Share	Length	α^{max}	v^{max} (m/s)	
		(m)	(m/s^2)	avg	SD
Bus	2.2%	12	2.90	9.98	2.33
Delivery	4.1%	6.5	3.03	10.91	3.01
Motorcycle	33.2%	2.1	4.14	13.90	3.95
Private	43.8%	5	3.32	12.09	3.25
Taxi	16%	5	3.10	11.5	3.03
Truck	0.7%	7.1	2.80	9.01	3.65

TABLE 5.1: Vehicle distinct properties of each type including share, length, acceleration, and maximum velocity

This defined model ensures that vehicles within our simulated environment behave distinctly based on their assigned properties. Moreover, the flexible distribution of vehicle types via the share value allows us to closely replicate real-world traffic scenarios, encompassing a range of vehicle types. However, it is important to note that while our vehicle model is comprehensive, the routing data extracted from the ground truth dataset remains relatively uniform and does not fully capture the dynamic nature of real-world traffic demand.

5.4.3 Traffic Demand & Randomness

The ground-truth traffic as acquired in Section 5.2 provides only one benchmark scenario to our proposed study, which is insufficient to extensively test the impact of our platooning algorithm in practical settings (expressing change in traffic input/demand). To this end, we introduce the randomness in the vehicle-generating process to acquire a broad result from our study evaluating against other junction management controls, namely, conventional traffic lights and the first-come-first-serve approach.

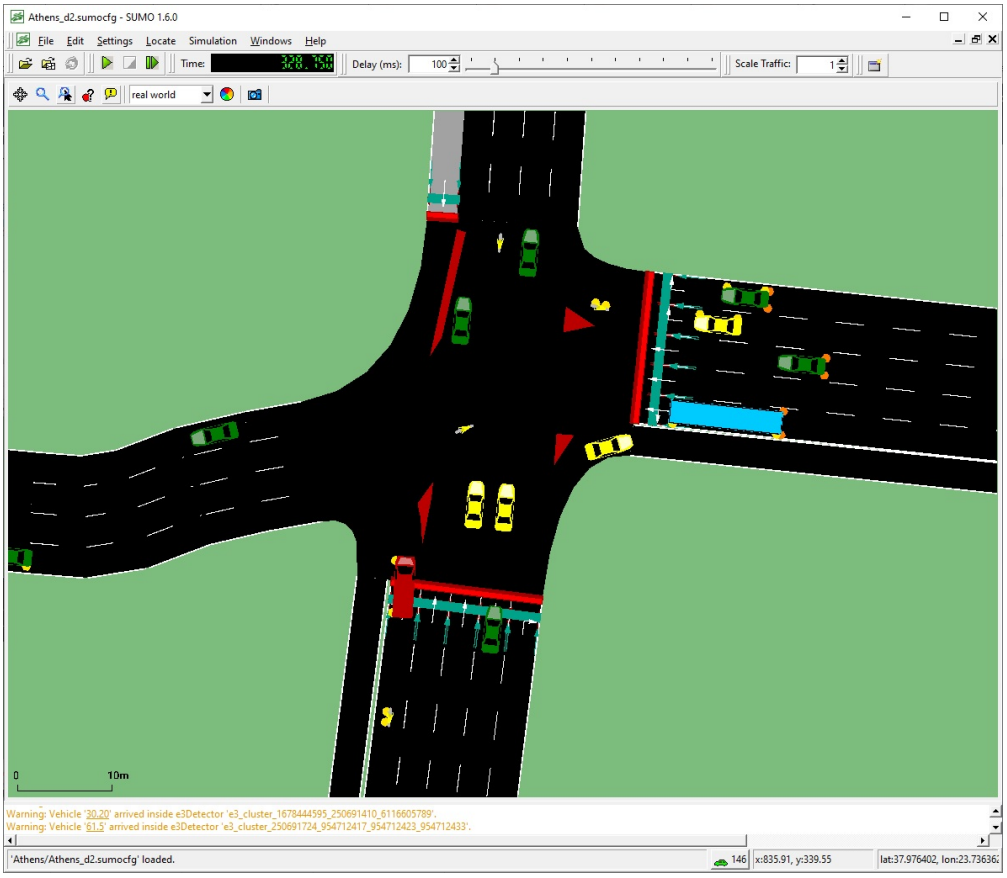

FIGURE 5.4: This figure shows a snapshot of SUMO simulating different vehicle types. The red ones represent delivery vehicles, the green ones represent private vehicles, the yellow ones represent taxis, the small yellow ones represent motorcycles, and the long blue ones represent buses. The truck is represented in blue but not included in this image. It is difficult to capture them all within one snapshot.

Besides, another advantage of randomness is to alleviate a biased wave pattern in the ground-truth traffic. The biased wave traffic pattern is a trait of the dataset since conventional traffic lights handle or release vehicles as a wave. If the vehicles are generated and released according to the timings on the ground truth, the wave-like traffic will still be seen. Without addressing this issue, the results for both FCFS and our platooning will be affected too. For example, inaccurate throughput and delayed results will be seen.

To address this, the ground-truth data are normalised into a series of routes while maintaining their demand patterns. The routes are defined as follows:

$$Route_x = < \{road_k, ..., road_K\},\ ratio > \tag{5.2}$$

denoting route $x \in X$ (total route) with a specific departure at $road_k$ travelling to $road_K$ through $road_{k+1}, ..., road_{K-1}$. This ensures that vehicles follow their designated route accordingly. The *ratio* is a constant value denoting the ratio of the number of vehicles generated per simulation to the total number of vehicles. Hence, even though the traffic

volume changes, the generated traffic can still represents the ground-truth demands. It should be noted that the share value in Table 5.1 and ratio are different. The ratio only defines how many vehicles are using particular routes. While, the share value defines how many vehicles of certain types will be generated per simulation.

The example usage of Eq. 5.2 is, in a 10,000 veh/hr scenario, when $Route_1$ has a ratio of 5%, the number of vehicles using this route will be 500 vehicles. Similarly, the number of vehicles on each route is changed accordingly, depending on the total number of vehicles per simulation. By accompanying this routing pattern and variation in vehicle types, we can generate multiple random sets of traffic demand while maintaining a similar pattern to the ground-truth traffic flow.

5.5 Empirical Evaluation

After defining all the models in this work, we proceed to describe the evaluation process. This section describes how our proposed mechanism's performance is evaluated with simulation, along with the simulation setup and results. Specifically, we use the traffic simulator SUMO version 1.6.0 (Krajzewicz et al., 2002) as in Chapter 4 and similar implementation of FCFS and platooning using TraCI (see Chapter 3 & 4 for more details of the algorithms).

5.5.1 Traffic Metrics

In the assessment process, we compare our proposed method against FCFS and fixed-time traffic signals (TFL) in terms of two key traffic metrics: junction delay and trip duration.

In particular, the junction delay refers to the amount of time captured from the point that vehicles enter the inbound roads to the point when they access the junction area. Therefore, only the delays attributed to the respective junction will be captured. On the other hand, trip duration quantifies the time vehicles need to travel from their initial departure point to their destination. This metric accumulates all delays incurred during the journey, including those experienced across multiple junctions and roads.

Our use of these two metrics serves a specific purpose, which is to demonstrate the performance of each junction control at different scale levels. Specifically, the junction delay expresses the performance at the level of individual junctions (*junction level*), whereas the trip duration reflects the performance from a higher perspective at the corridor or *network level*, where the whole trip throughout a network is considered instead. Note that to see the change in values over time and allow the simulation to warm up,

the junction delay results are captured in 15-minute intervals (simulation time) with a time step of 0.25 seconds.

5.5.2 Simulation Setup

To evaluate our method, we examined five traffic volume scenarios, from 6,000 to 14,000 veh/hr (increasing by 2,000 veh/hr). In particular, traffic volumes of 6,000 to 8,000 veh/hr represent light traffic, similar to what is usually encountered in early mornings or inter-peaks. On the other hand, traffic volumes between 10,000 and 14,000 veh/hr represent heavy traffic, similar to morning and evening peak times. Each traffic volume scenario was simulated over 20 runs.

The purpose of having multiple scenarios is not only to represent traffic at different times of the day but also to demonstrate the impact of our dynamic platoon formation under different conditions. For example, in light traffic, the resource reservation mechanism works efficiently, effectively resolving waiting vehicles and keeping the waiting queue short. However, in heavy traffic, the resource reservation mechanism alone cannot maintain a short waiting queue due to increased vehicle arrivals. As the queue grows longer, the platoon has a higher chance of forming, but the externalities' cost of the platoon increases as well. Therefore, we can evaluate how our method performs both in simple scenarios when the externalities are relatively low and in more challenging conditions, where the externalities are substantially higher.

Note that, despite the changes in total traffic volumes, a similar pattern of the ground-truth demands can be achieved as the *ratio* of each route remains static (see Eq. 5.2).

We next describe our evaluating results, which are junction delays and trip duration.

5.5.3 Junction Delay Results

We measured the junction delay through the average travel time (in seconds) on the inbound roads over eight junctions and compared our method against TFL and FCFS. It is noted that the term "travel time" refers to the amount of time that DAs need before accessing one junction within a time interval, not the amount of time DAs need from all passing junctions within their routes; thereby, it only captures the performance per junction. Additionally, the outbound roads of one junction can be the inbound roads of its nearby junctions.

In light traffic scenarios, FCFS and our mechanism can shorten the junction delay by up to 53-54% compared to TFL. Likewise, FCFS and our mechanism still outperform TFL by reducing the junction delay by up to 65%, even with heavy traffic scenarios, but the difference here is quite large. This may be attributed to unavailable space on the

road, where vehicles cannot be fed into certain roads anymore. Therefore, only part of the traffic is being recorded. We will discuss more in Section 5.5.4. Compared with the non-platoon mechanism, our platoon mechanism slightly outperforms FCFS in only a few scenarios, decreasing the delay by 2.16% at 10,000 veh/hr and 4.69% at 12,000 veh/hr. The full results of the different mechanisms can be seen in Table 5.2, and the results graph is depicted in Fig. 5.5.

However, the results here only reflect the performance at the single junction level. We next continue to evaluate our proposed mechanism from a higher point of view.

junction delay

Methods & comparison	Traffic volumes (veh/hr)				
	6,000	8,000	10,000	12,000	14,000
TFL	17.06 ±2.56	19.19 ±1.86	23.83 ±3.91	25.35 ±2.62	38.18 ±4.72
FCFS	10.04 ±0.27	10.38 ±0.32	11.13 ±0.63	12.58 ±1.77	13.08 ±0.9
Platoon	10.1 ±0.34	10.38 ±0.53	10.89 ±0.45	11.99 ±1.04	13.25 ±1.27
TFL vs FCFS	-41.15%	-45.91%	-53.29%	-50.37%	-65.74%
TFL vs Platoon	-40.80%	-45.91%	-54.30%	-52.70%	-65.30%
FCFS vs Platoon	0.60 %	0.00 %	-2.16 %	-4.69 %	1.30%

TABLE 5.2: The table shows the highest results of junction delay of different junction controls: TFL, FCFS and our platooning mechanism. The plus and minus indicate 95% confidential interval. The bottom part compares the difference in percentage, where negative and positive values indicate a decrease and increase, respectively.

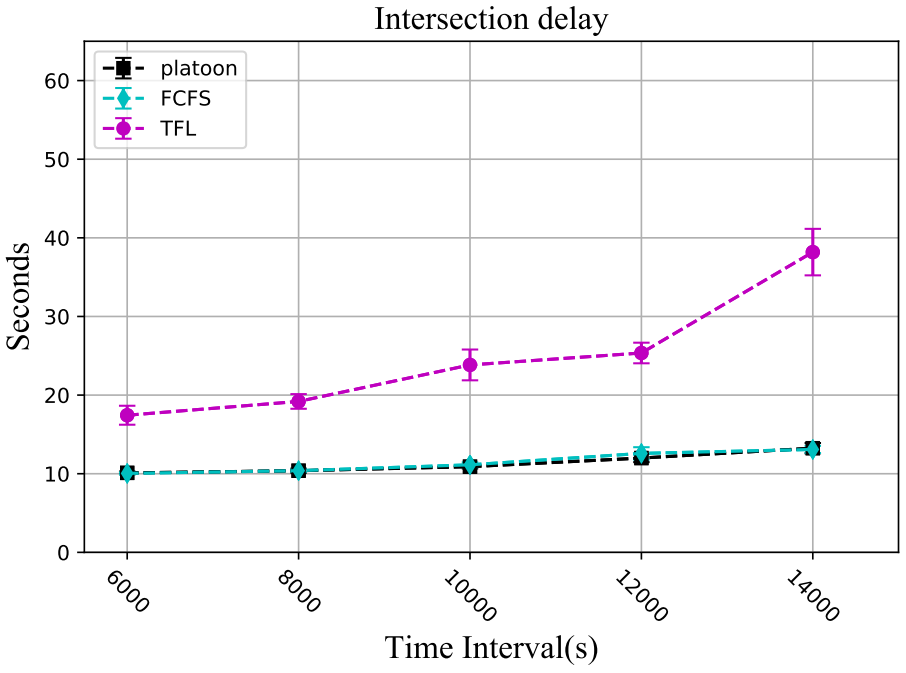

FIGURE 5.5: This graph shows the average junction delays of three different junction controls, which are TFL, FCFS, and platooning.

5.5.4 Trip Duration Results

To evaluate our method further, we also measured the average trip duration on all vehicles. The trip duration (in seconds) identify how long vehicles require to complete their designated trips. This means that the queuing delays are also accumulated as they journey through the junction(s), expressing the performance of the whole network. It should be noted that the average results here are weighted averages, as vehicles in our simulation are heterogeneous. Results are weighted using an estimation of passengers/loads according to the cost-effectiveness study in The European Commission, Standard & Poor's DRI and KULeuven (1999). To be precise, estimated values are 20.80 passengers on buses, 1.56 on deliveries, personal & taxis, 1.186 on motorcycles, and 3.07 tons on trucks. The main reason for using the weight average is the length of vehicles; some heavy-duty such as trucks, buses and delivery vehicles are slower to acquire reservations compared to taxis and private vehicles where their lengths are shorter. This is to reduce the bias in the results.

However, the trip duration results alone do not completely record the actual trip duration. During simulation, as the traffic volumes increase, some vehicles cannot depart according to their schedule due to unavailable space preventing them from entering the network. These vehicles are kept delayed outside of the simulation area, waiting to depart when the space becomes available. Fig. 5.6 shows an example situation that SUMO cannot generate some vehicles. In the figure, it can be seen that, on the south arm of the junction, the traffic is extremely crowded, and the space is fully occupied. There is no more room for vehicles to depart. In more detail, on the left window within the figure, the 2nd parameter called "insertion-backlogged vehicles" indicates the number of vehicles being kept outside waiting to be departed. With this example, only 500 seconds had past this number is already increase to 24 vehicles, and it grows constantly as the simulation runs. At the end of the simulation (3,600 seconds), this number can reach up to 100 to 200 vehicles.

To be precise, the issue is that SUMO captures the trip duration only when vehicles enter the simulation, meaning that any prior departure delays are entirely ignored. According to this, the trip duration does not indicate the actual time of each journey. To cope with this issue, we consider an additional output value named trip departure delay, which indicates "the time the vehicle had to wait before it could start his journey" according to SUMO manual[4]. In this way, the journey's departure delay is not overlooked.

Therefore, to fully cover all the usage time, the sum of trip duration and trip departure delay is used instead, which we refer to as "total trip duration". The weighted average of the total trip duration is shown in Table 5.3. It can be seen that, in light traffic scenarios, FCFS can outperform TFL by up to 20.77%, and, similarly, platooning can

[4] https://sumo.dlr.de/docs/Simulation/Output/TripInfo.html

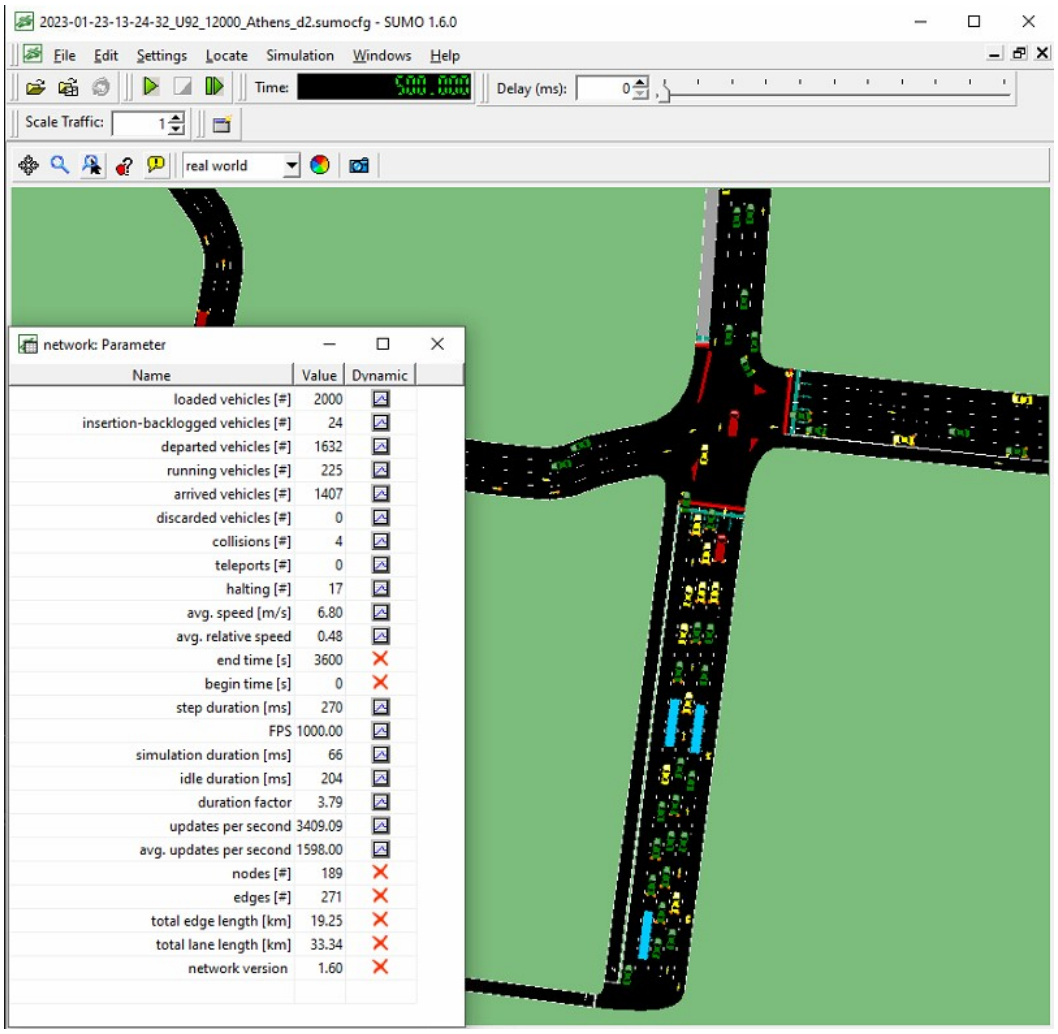

FIGURE 5.6: This figure shows an example situation where certain vehicles cannot depart on their designated schedule due to unavailable space. The setting of this 12,000 veh/hr with FCFS approach.

also outperform TFL even by up to 22.35%. On the other hand, in heavy traffic, FCFS cannot outperform TFL and even increases the total trip duration by up to 44.47%. In contrast, our platooning can reduce the total trip duration by 20.7% compared to TFL. More insights of this situation will be discussed in the next section. Moreover, our proposed mechanism also outperforms the FCFS, decreasing the total trip duration by up to 2% with light traffic and by 45% with heavy traffic.

5.5.5 Discussion

By evaluating a range of settings and comparing platooning and non-platooning-based mechanisms, we derive several useful insights. Initially, from the junction delay aspect, FCFS works well in many scenarios and significantly outperforms TFL in terms of delays. However, the trip duration results, which express the performance in a corridor

Total trip duration

Methods & comparison	Traffic volumes (veh/hr)				
	6,000	8,000	10,000	12,000	14,000
TFL	63.4 ±0.53	66.89 ±0.21	70.68 ±0.2	76.97 ±0.16	125.46 ±0.46
FCFS	51 ±0.12	52.48 ±0.12	56 ±0.16	111.2 ±1.24	154.9 ±1.08
Platoon	50.8 ±0.16	52.6 ±0.2	54.88 ±0.13	61.04 ±0.2	112.9 ±0.78
TFL vs FCFS	-19.56%	-21.54%	-20.77%	44.47%	23.47%
TFL vs Platoon	-19.87%	-21.36%	-22.35%	-20.70%	-10.01%
FCFS vs Platoon	-0.39%	0.23%	-2 %	-45.11%	-27.11%

TABLE 5.3: This table shows the weighted average total trip duration results of different junction controls: TFL, FCFS and our platooning mechanism. The plus and minus indicate 95% confidential interval. The negative values here indicate a decrease while positive values indicate an increase.

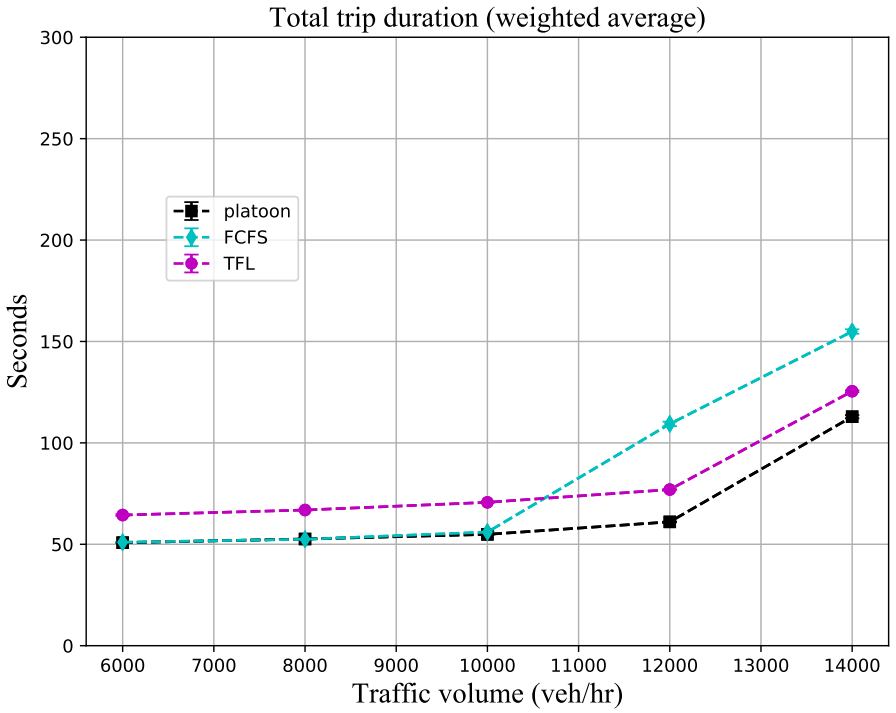

FIGURE 5.7: This graph shows the average total trip duration of three different junction controls, which are TFL, FCFS, and platooning.

aspect, show otherwise. As the traffic becomes considerably high, FCFS starts to perform relatively poorly and cannot even outperform TFL.

In more detail, the downside of FCFS is that it prioritises the sequence of releasing DAs over the impact on the junction as a whole. First come first serve is one way to resolve any conflicts, but in this case, it negatively impacts the system. The FCFS principle always grants reservations in order of arrival regardless of whether it creates significant delay costs to the other DAs (i.e., the externalities). The amount of delay may seem relatively small and negligible as FCFS can still achieve good performance over

conventional traffic lights, as shown in many studies (Dresner and Stone, 2008; Carlino et al., 2013; Liu et al., 2013), especially in light traffic or idealised predictable situations. However, under realistically-high traffic and more dynamic situations, the delays become more noticeable, and they negatively affect the traffic flow, as highlighted by our findings. The results suggest that using FCFS approach in a city that always has a high traffic demand will bring more harm to the system than using conventional traffic light control.

On the other hand, the advantage of our method lies in the core algorithm that minimises the externalities across the junction while reducing unnecessary vehicle movements in the form of platooning. As a result, our method can significantly improve the performance from the network-level aspect and outperform both TFL and FCFS, even when traffic volumes increase. To demonstrate the performance improvement results from our platoon formation, we provide the average number of platoons formed during the experiment in Table 5.4.

It can be seen from the table that under 6,000 veh/hr, which is a considerably low traffic situation, the average number of platoons is extremely small. This can result from short queuing lengths where chance of platooning is extremely low. Therefore, the previous results show similar performance between FCFS and our platooning. Despite the heavy traffic where long queuing lengths lead to higher platooning opportunities, our externalities-driven approach can adapt and constrain itself dynamically not to overly form platoons.

However, in the junction-level aspect, despite our attempt to minimise the externalities caused by platoons, they still cause a slightly negative effect on the junction (see 14,000 veh/hr Table 5.2). Apparently, small externalities are unavoidable in exchange for forming platoons. They seem significant enough to create chain delays to the end of the queue, increasing the average travel time delays, especially in such extreme traffic. Moreover, this issue also causes a performance drop in the network-level aspect.

	Traffic volumes (veh/hr)				
	6,000	8,000	10,000	12,000	14,000
Platoon	11.6	53.66	244.6	748.14	1105.5
Vehicles per platoon	2.06	2.09	2.23	2.41	2.47

TABLE 5.4: This table shows the average number of platoons formed and the average number of vehicles per platoon in different traffic volume scenarios.

5.6 Limitations

Despite considering several factors that essentially impact the realism of the simulation, there are still limits to the calibration process. The main limitation comes from the

behaviour of motorcycles, which is rather unpredictable as some already cause issues mentioned in Section 5.1. To be specific, motorcycles mostly drive alongside each other in a pack of two to four, creating a complex situation to simulate in SUMO. Frankly, SUMO allows any small-width vehicles, e.g., bicycles and motorcycles, to drive side by side with each other by enabling a sublane feature. However, enabling this feature would cause a major effect on our decentralised approach and platooning as they are not initially designed to support it. More importantly, it is unclear how alongside-driven motorcycles are going to behave as platoon members. To be precise, each motorcycle has its own parameters, e.g. acceleration rate and max speed, thereby when driving alongside they may not speed up or maintain a similar speed to each other. Hence, we disable the sublane feature, meaning that motorcycles are assumed to consume the width space of one car.

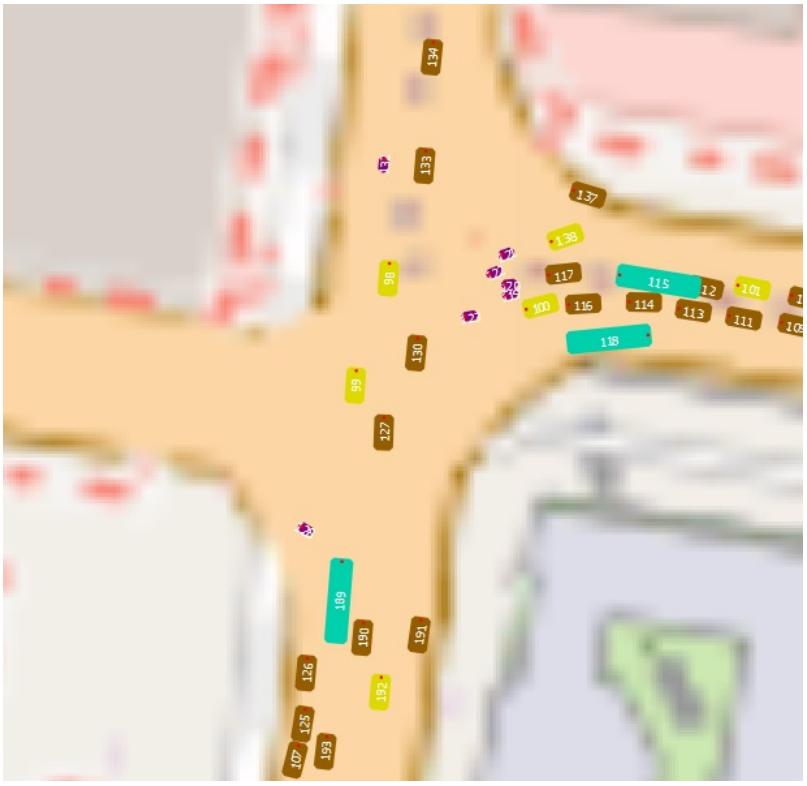

FIGURE 5.8: This figure shows an example situation where motorcycles (small rectangles) usually queue beyond the stop line (see the east side).

Furthermore, another limitation in the calibration process also comes from the motorcycle's behaviour. In particular, while waiting for the green phases at any junction, most motorcycles are likely to queue beyond the stop line (see Fig. 5.8), aiming to cross the junction as quickly as possible. Unfortunately, at the moment, SUMO cannot replicate or simulate this behaviour, which slightly affects the traffic flow, especially when comparing the ground-truth traffic against simulated traffic. Specifically, the average travel time in the simulated system is 30% higher than the ground truth and 50% on motorcycles alone. Overall, these motorcycle behaviours, especially the beyond-stop-line

queuing and alongside-driven behaviour, prevent us from replicating the ground-truth traffic flow accurately.

However, heterogeneous vehicle types and realistic traffic geometry are only parts of modelling practical traffic. To ultimately demonstrate the performance in reality, an essential matter of pedestrians must be addressed, especially in urban areas where they are prevalent.

Chapter 6

Pedestrian Consideration

This chapter focuses on how pedestrians can be considered and incorporated into our platooning method, considering highly practical situations in urban or crowded areas. Primarily, the objective of this work is to address Research Challenge 6. To do so, we utilise the highly-realistic calibrated traffic model in Chapter 5 and introduce an element of a pedestrian into the system. This traffic model provides a fundamental to this additional development. Then, we design a pedestrian model for our autonomous junction control driven by the overall waiting time on different incoming roads. Specifically, the uneven traffic volume between different incoming roads allows us to grant pedestrian right-of-way with a minimised burden on the drivers. Lastly, we evaluate our method against other junction controls: traffic lights and FCFS with pedestrians.

This chapter is structured as follows. Firstly, we introduce the pedestrian model in the case of traffic light control. Secondly, a waiting-time-driven decision tree for the autonomous junction controls (compatible with FCFS and our platooning) is described. Thirdly, we evaluate our method against different junction controls with pedestrians. In conclusion, the limitations of this work will be stated.

6.1 Pedestrian Model for Traffic Signal Control

In real-world situations, pedestrians are an essential factor that needs to be taken into consideration when managing junctions. Usually, with traffic light signal control, it is simple to include pedestrian phases in the signal programs as it was originally designed to support them. The pedestrian phases are normally used and seen extensively in many urban cities or crowded areas. However, as we study this in a simulation context, it's crucial to ensure that the simulated traffic light control accurately reflects real-world usage. To achieve this, we rely on the vehicle movement dataset, which

is discussed in Chapter 5. By analysing the traffic light programs (i.e., red-green timings), we can replicate the pedestrian phases and make our simulation as realistic as possible." The replication processes are as follows:

1. Locating all of the pedestrian crossing areas in the corridor network.
2. Virtually placing these crossings on the simulated junctions in SUMO road settings.
3. Visualising the traffic flow using the traffic visualisation tool Travia [1] (see Section 5.1).
4. Carefully recording the timings of pedestrian phases by observing the changes between traffic signals (stopping or releasing the traffic).
5. Transfering the recorded timing into SUMO and ensuring that the pedestrian phase timings match visualised traffic flow in (3).

However, replicating the timing of pedestrian phases presents a significant challenge, particularly for one of the eight junctions that feature multiple traffic islands for pedestrians to wait for their right of way. This junction is illustrated in Fig. 6.1. Since the recorded dataset only provides vehicle movements, not light programmes, identifying the timings of pedestrian phases requires careful attention. Particularly, we addressed this challenge by manually observing the red and green timings of this junction, dividing the timings into smaller chunks, and temporarily stopping free turns. This allows pedestrians to circulate around the traffic islands in a clockwise direction and safely cross to the other side of the road. Each traffic island acts as a waiting/queuing area for pedestrians as they move from one island to another. These observed pedestrian phase timings are transferred into the simulated junction, reproducing traffic light control with islands that support pedestrians.

We next describe the pedestrian model designed for our autonomous junction setting.

6.2 Pedestrian Model for Autonomous Junctions

Generally, integrating pedestrian phases into autonomous junction control is a challenging and uncommon task. Many studies in autonomous junction management are likey to avoid including pedestrians due to the increased complexity and potential negative impact on traffic performance, such as increased travel and waiting times for vehicles. Studies from Niels et al. (2020); Chen et al. (2020) have shown that having

[1] The traffic visualisation tool as described in Chapter 5 that can simulate the vehicle movements given the Athen, Greece drone dataset.

FIGURE 6.1: This figure shows the junction that has multiple islands as pedestrian waiting areas.

too many pedestrian phases can lead to significant delays on the driver's side. To address this issue, we propose a novel vehicles' waiting-time-driven pedestrian model that aims to balance the right-of-way between pedestrians and drivers while maximising traffic flow. In the following, we outline the design process for incorporating pedestrian phases into autonomous junction management.

The core design of our pedestrian model is based on a hard constraint on waiting time, limiting pedestrians to a maximum of 90 seconds of waiting time. Once this constraint is reached, the junction must explicitly give the right of way to pedestrians. However, within the 90-second time frame before granting pedestrian right-of-way, there is an opportunity to optimise the traffic flow. To do so, we categorise incoming roads on the junction into two types: "secondary" and "primary", defined by the amount of traffic volume. The road with high traffic volume is designated as *primary*, while the road with lower volume is designated as *secondary*. This categorisation allows us to consider pedestrian waiting time separately for each road type.

One advantage of this road categorisation is that granting right-of-way to pedestrians on one incoming road can benefit vehicles on the other incoming road. For instance, when pedestrians are given priority on the *primary* road, traffic on the *secondary* road can flow at full stream uninterrupted, improving traffic efficiency on one road. An example of the employment of primary and secondary pedestrian phases is depicted in Fig. 6.2 . However, constantly switching pedestrian phases between primary and secondary roads can lead to indefinite wait times for turning vehicles, as pedestrians

always block turning lanes. This is where our autonomous junction management system comes into play.

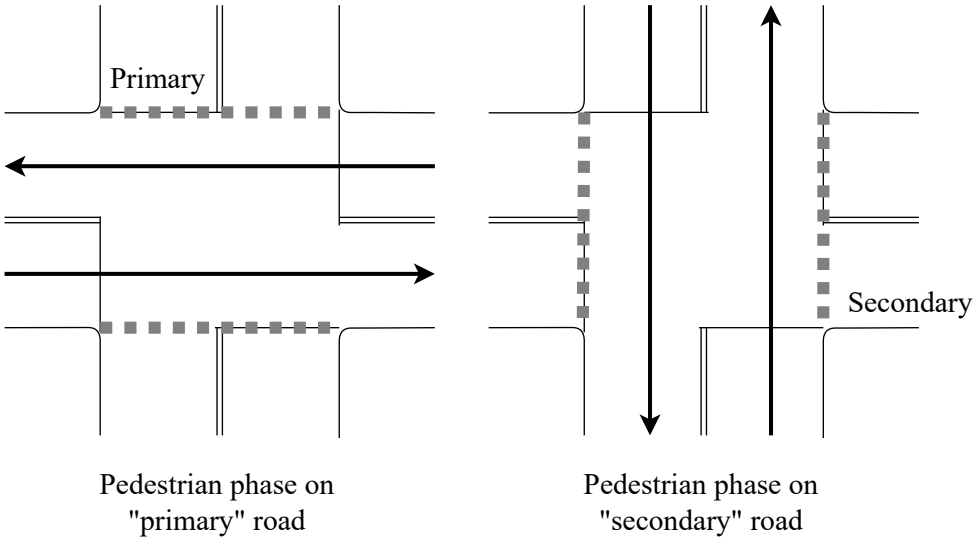

FIGURE 6.2: The rectangles represent the granted pedestrian right-of-way occupying space on the junction areas blocking some vehicles' navigations. The arrows denote the possible traffic direction given different pedestrian phases.

With the coordination and automation capability of autonomous junctions, the system can spontaneously release multiple straight-going vehicles to cross and, more importantly, the turning vehicles to turn, thereby maximising the overall throughput and lowering vehicle delays (especially the turning vehicles). Our system operates through four operational phases, which can be switched dynamically to accommodate different traffic conditions, which are:

1. Pedestrian phase on the primary road.
2. Pedestrian phase on the secondary road.
3. Free phase operating our platooning approach.
4. Pedestrian phase on both primary and secondary roads triggered by the hard constraints.

By deliberately switching to the most suitable operational phases for a specific point in time, the junction performance can be maximised while minimising overall vehicle delays.

During the simulation, several values are being monitored which are:

- W_p^{pri} – Pedestrian's waiting time on the primary road.
- W_p^{sec} – Pedestrian's waiting time on the secondary road.

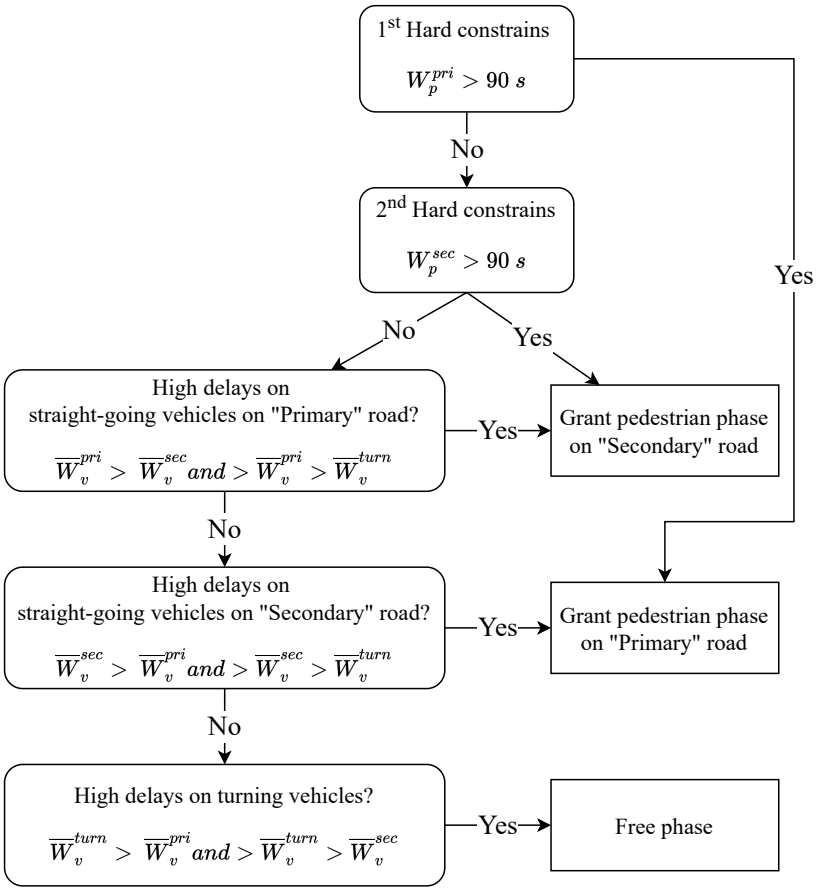

FIGURE 6.3: This figure shows a decision tree deciding the operational traffic phases.

- $\overline{W}_v^{pri}$ – Average straight-going vehicles' waiting time on the primary road.
- $\overline{W}_v^{sec}$ – Average straight-going vehicles' waiting time on the secondary road.
- $\overline{W}_v^{turn}$ – Average turning vehicle' waiting on all roads.

The W_p^{pri} is straightforward to record as it starts counting as soon as there is no granted pedestrian phase on the primary road, and it is vice versa for W_p^{sec}. The $\overline{W}_v^{pri}$ and $\overline{W}_v^{sec}$ denote the average duration that straight-going vehicles on respective roads become stationary (speed $\leq$ 0.1 m/s). Similarly, $\overline{W}_v^{turn}$ records the duration that turning vehicles is at full stop. These values are used to decide the active operation phase for the junction.

We propose a decision tree that uses these values to determine which operational phase will be active. The decision tree is detailed in Fig. 6.3. It can be seen in the figure that operational phases in (4), where both primary and secondary roads are given pedestrian right-of-way, are not included. Note that this decision tree is considered in real-time; thereby, whenever both the 1st and 2nd hard constraints condition have been met at the same time, (4) operational phase will be active. Nevertheless, the main advantage

of this approach is that every time the pedestrian phase is given, due to high delays on a specific road, the waiting time of the pedestrian will be reset (W_p^{pri} or W_p^{sec}). This reduces the chances of enabling the operational phase (4) as both hard-constrain conditions are less likely to be met.

6.2.1 Waiting Time Estimation

However, while the monitored waiting time provides information on the waiting time for vehicles in the queue, it has not accounted for moving vehicles that have not yet reached the waiting queue. Here, the waiting time only indicates the current system state and does not anticipate the impact of operational phases on the system in the near future. This lack of foresight results in suboptimal operational phases being selected. To address this issue, we propose defining a 30-second time interval for all operational phases and estimating the waiting time 30 seconds ahead. Overall, this approach will enable the pedestrian model to anticipate the impact of operational phases on the system and select optimal operational phases accordingly, ensuring efficient decision-making.

For example, $\overline{W}_v^{pri}$ is the estimated average waiting time in the next 30 seconds, considering that the primary road will be stopped, and moving vehicle(s) will be stopped having a certain amount of waiting time. This waiting time estimation considers as if the roads are about to be blocked by pedestrian phases within the next 30 seconds. It is straightforward to estimate the 30-second-ahead waiting time of the queuing or stationary vehicles by simply adding the 30 seconds more into their current waiting time. However, for moving and arriving vehicles, we utilise the average vehicle speed ($\overline{v}_{rd}^t$, in m/s) and the input rate (vehicles/second) on that road. With this, we can calculate the waiting time for the moving and arriving vehicles after they reach at the stop line.

$$\sum_{j}^{|A_{mov}^{rd}|} max\left(0,\ 30 - \frac{d_j^{line}}{\overline{v}_{rd}^t}\right) \tag{6.1}$$

$$\sum_{j}^{floor(30\times \text{input})} max\left(0,\ (30 - \frac{j}{\text{input}}) - (\frac{l_{rd}}{\overline{v}_{rd}^t})\right) \tag{6.2}$$

The summation of the estimated 30-second-ahead waiting time for the moving vehicles at time t can be calculated using Eq. 6.1. In the equation, A_{mov}^{rd} is a set of moving vehicles on the road at time t. d_j^{line} denotes the distance from a_j position to the stop line, and $\overline{v}_{rd}^t$ represents the average vehicles' speed before leaving on this road (rd) at time t. Basically, the equation aggregates waiting times of moving vehicles as soon as they arrive at the stop line. If moving vehicles cannot reach the stop line within the 30-second interval, their waiting time is constrained to zero.

Furthermore, we also include vehicles that have not even been generated into the system through the input or arrival rate variable, which is denoted by input. Eq. 6.2 computes the estimated 30-second-ahead waiting time by considering the number of vehicles arriving through $floor\,(30 \times \text{input})$, flooring the number to be an integer value. The term $(30 - \frac{j}{\text{input}})$ denotes the time remaining for vehicle j to reach the stop line. Additionally, the $(\frac{l_{road}}{\overline{v}^t_{rd}})$ estimates the average travel time of vehicles driving with the distance or length of the road (l_{rd}) to the stop line. Then, similar to Eq. 6.1, the estimated 30-second-ahead waiting time is constrained to zero if they will not arrive at the stop line in time.

Subsequently, for each road, the waiting time will be summed up, accounting for all the possible vehicles that will be on the road, which are stationary vehicles, moving vehicles, and arriving vehicles, and then average the waiting time by the total vehicles ($|A^{rd}| + |floor(30 \times \text{input})|$). It is important to note that turning vehicles are not included in these calculations, as they are treated separately. As a result, we can estimate the 30-second-ahead waiting time on average for each road, namely $\overline{W}_v^{pri}$ and $\overline{W}_v^{sec}$.

Turning vehicles, regardless of road type, are considered as a whole. The number of stationary turning vehicles includes those from both primary and secondary roads, and likewise for moving and arriving vehicles. After estimating the total number of turning vehicles, the $\overline{W}_v^{turn}$ is calculated through a similar process as $\overline{W}_v^{pri}$ and $\overline{W}_v^{sec}$.

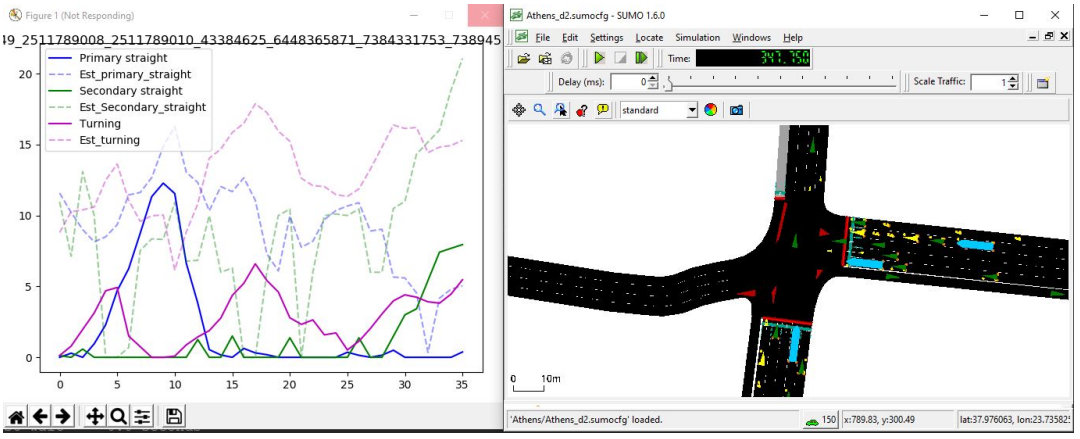

FIGURE 6.4: The graph on the left monitors the average waiting of vehicles on different roads in real-time, while the right side is the SUMO simulation window displaying the simulated traffic on the focus junction. One unit on the x-axis equals recorded three seconds

To demonstrate the use of this 30-second-ahead waiting time estimation, Fig. 6.4 shows a real-time waiting time tracker of the distinct road type. The solid lines represent the waiting time at the current traffic state, while the dashed lines represent the 30-second ahead estimation of waiting time. This estimation allows us to anticipate the situation in the near future and decide on it, which significantly improves the system by maximising the traffic flow and minimising the vehicle delay without violating the hard constraints.

6.2.2 Pedestrian Phase Timings

Nevertheless, the 30-second operation phase interval used in the previous section is a default value, and the actual pedestrian phase time interval is defined/calculated depending on the individual road width.

To specify the pedestrian phases' time interval, we base on a timing calculation from (The Department for Transport under licence from the Controller of Her Majesty's Stationery Office, 2019). In particular, the calculation considers multiple factors, including an invitation to cross, inv_to_cross, lane's width, number of lanes, and pedestrian speed, see Eq. 6.3.

$$inv\_to\_cross + round\left(\frac{lane\_width \times number\_of\_lanes}{pedestrian\_speed}\right) \tag{6.3}$$

Here, $round()$ is an integer rounding function, inv_to_cross has a default value of 12 seconds, the default lane width is 3.2 metres, and the default pedestrian speed is 1.2 m/s. Essentially, based on this calculation, the timing of the pedestrian phase can be determined dynamically depending on the number of lanes within certain roads.

After we have defined the pedestrian model utilised within our proposed junction control methods, we continue to evaluate its performance against different junction control methods.

6.3 Empirical Evaluation

In this section, we detail the evaluation of our pedestrian model. Similar to Chapter 5, we evaluate our method (platooning with pedestrians control or Platoon_ped) via SUMO simulation against traffic lights (TFL) and non-platoon-based method (First-Come-First-Serve with pedestrian control or FCFS_ped). Note that FCFS_ped is similar to normal FCFS, but with an adoption of pedestrians control in Section 6.2, in the free operational phase, FCFS is used instead of platooning. Hence, we can provide a fair comparison with the state-of-the-art method.

Similar to the simulation settings in Section 5.5, we simulate each method with five traffic volume scenarios starting from 6,000 to 14,000 veh/hr (increasing by 2,000 per step). The traffic volumes representing light traffic scenarios are 6,000 to 8,000 veh/hr, while 10,000 to 14,000 represent heavy traffic scenarios.

Furthermore, from the traffic configuration where the simulated environment is an urban city like Athens, this can be looked at as a compatibility assessment to the crowded situations. The pedestrian demand is designed as a constant as there is always at least one pedestrian waiting at every corner, and their waiting time increases every second whenever they do not have a right-of-way. Particularly, the objective of this setting is

to investigate the performance of different autonomous junction controls against the worst-case pedestrian demand. The challenge comes from the shared space between pedestrians and the CAVs: how efficiently the autonomous junction performs and benefits from the existence of CAVs under the heavy constraint of pedestrian demand.

6.3.1 Traffic Metrics

For evaluating the performance of the different methods, we employ traffic metrics similar to those used in Section 5.5: junction delays and trip duration.

To be specific, junction delays provide insights into performance at the junction level, accumulating lane changing, queuing, and reservation waiting times. Particularly, this metric captures the total time vehicles spend travelling before exiting the inbound road or crossing the junction. On the other hand, trip duration reflects performance at the corridor level, indicating how quickly vehicles complete their journeys on average while being delayed from junction(s). It offers valuable insights into the traffic flow within the corridor itself.

Next, we continue to elaborate on the junction delay results, which reflect the performance of the methods at the junction-level perspective.

6.3.2 Junction Delay Results

We measured the average travel time over eight junctions and compared our method against TFL and FCFS_ped. Note that these junction delay results are captured in 15-minute intervals, allowing the simulation and traffic to warm up and stabilise.

junction delays

Methods & comparison	Traffic volumes (veh/hr)				
	6,000	8,000	10,000	12,000	14,000
TFL	18.26 ±1.89	19.14 ±2.14	23.28 ±2.73	28.72 ±4.06	45.92 ±11.51
FCFS_ped	13.07 ±0.60	14.16 ±0.70	19.16 ±2.01	23.02 ±3.77	24.25 ±3.19
Platoon_ped	12.94 ±0.79	14.29 ±0.94	15.75 ±1.33	20.41 ±3.52	24.67 ±6.10
TFL vs FCFS_ped	-28.42%	-26.01%	-17.69%	-19.84%	-51.54%
TFL vs Platoon_ped	-29.13%	-25.33%	-32.34%	-28.93%	-46.27%
FCFS_ped vs Platoon_ped	-0.99%	0.92%	-17.80%	-11.34%	1.73%

TABLE 6.1: The table shows the highest results of junction delay of different junction controls: TFL, FCFS_ped and our platooning mechanism. The plus and minus indicate a 95% confidential interval. The bottom part compares the value difference between methods in percentage, where negative and positive values indicate a decrease and increase, respectively.

From the results in Table 6.1, it can be seen that both FCFS_ped and our platoon_ped can outperform TFL significantly. To be more precise, in light traffic, FCFS_ped can

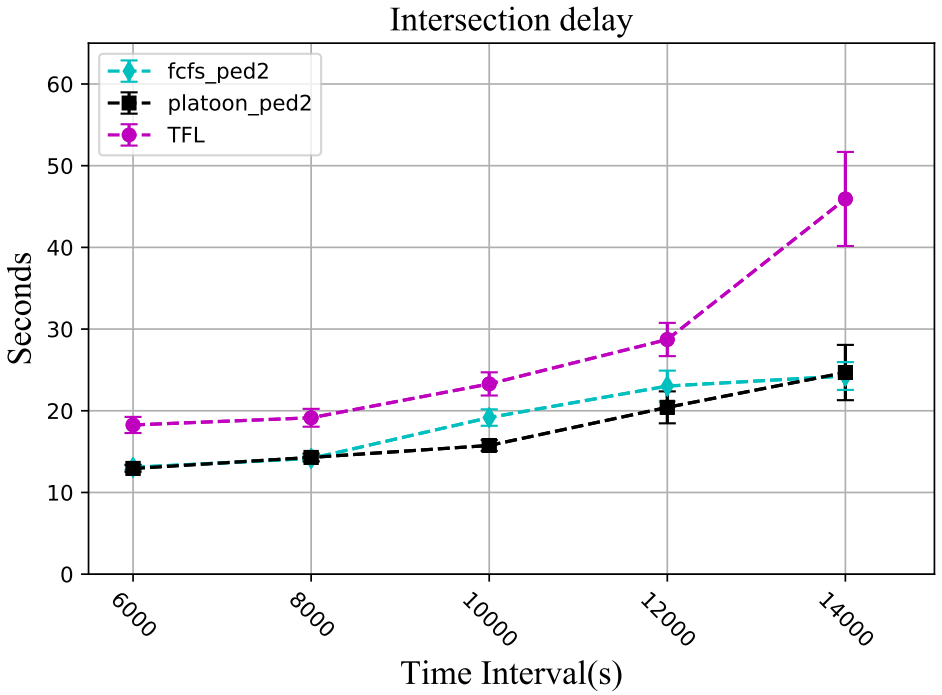

FIGURE 6.5: This graph shows the highest average junction delays of three methods: TFL, FCFS_ped and platoon_ped under different traffic volumes.

reduce the junction delays by up to 28.65%, and our platooning can reduce by up to 29.37% compared to TFL. Moreover, in heavy traffic scenarios, the junction delays can be reduced by ≈40% by both FCFS_ped and our method.

However, in comparison between autonomous junction control (see "FCFS_ped vs Platoon_ped"), our method does not always outperform the state-of-the-art one. Specifically, in 6,000, 8,000 and 14,000 veh/hr, our platooning does not entirely outperform or be overwhelmed by FCFS_ped. They present a similar performance, less than 0.50 seconds difference in results. Particularly, the difference can be seen in Fig. 6.5. Nevertheless, conditions that work well for the platooning appear to be between 10,000 to 12,000 veh/hr, in other words, medium traffic volumes.

6.3.3 Trip Duration Results

In the trip duration results, the similar weighted average and total trip duration metric (see Section 5.5.4) are used here as well. The simulation results can be seen in Table 6.2. The evaluation results appear to be different when we look at the system from the corridor-level aspect. To be precise, both FCFS_ped and our platooning no longer outperformed TFL in all traffic scenarios. In light traffic scenarios, FCFS_ped and our platooning can outperform TFL by reducing the total trip duration up to 12.35% with FCFS_ped and up to 15.38% with platooning. However, in heavy traffic scenarios,

FCFS_ped cannot outperform TFL and even increases the total trip duration by up to ≈50%, which is considerably huge. Unlike platooning, even though TFL outperforms our platooning by ≈16% in 12,000 veh/hr, a small decrease in the total trip duration, 7.91%, can be seen in 14,000 veh/hr.

Lastly, to compare between two autonomous junction controls, it appears that Platoon_ped can outperform FCFS_ped in any traffic scenarios. Notably, the most superior scenarios are at 12,000 veh/hr where platooning outperforms FCFS by up to 23.21%. By the rest of the traffic scenarios, platooning can only outperform FCFS_ped by up to ≈15%. The performance comparison in a graph can be seen in Fig. 6.6.

Total trip duration

Methods & comparison	Traffic volumes (veh/hr)				
	6,000	8,000	10,000	12,000	14,000
TFL	69.05 ±0.37	77.15 ±0.46	96.23 ±0.96	132.01 ±1.18	266.91 ±2.06
FCFS_ped	62.55 ±0.24	67.62 ±0.23	88.59 ±0.42	200.4 ±1.71	286.64 ±2.02
Platoon_ped	61.64 ±0.33	65.28 ±0.31	75.27 ±0.38	153.94 ±1.70	245.79 ±2.88
TFL vs FCFS_ped	-9.41%	-12.35%	-7.91%	51.85%	7.39%
TFL vs Platoon_ped	-10.73%	-15.38%	-21.75%	16.61%	-7.91%
FCFS_ped vs Platoon_ped	-1.45%	-3.46%	-15.04 %	-23.21%	-14.25%

TABLE 6.2: This table shows the weighted average total trip duration results of different junction controls: TFL, FCFS_ped and platoon_ped. The plus and minus indicate a 95% confidential interval. The negative values here indicate a decrease while positive values indicate an increase.

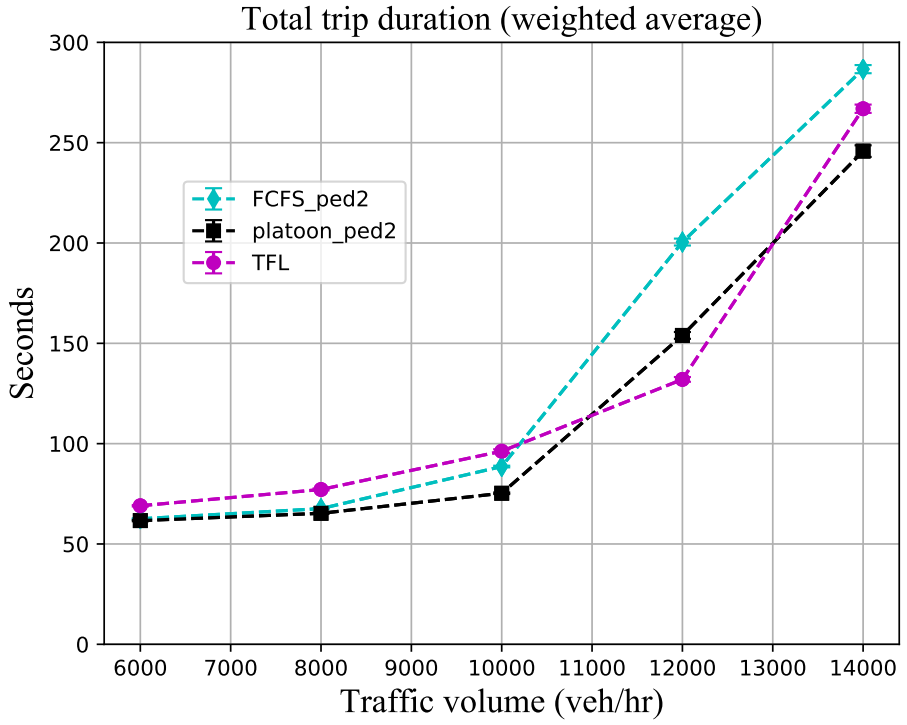

FIGURE 6.6: This graph shows the weighted average total trip duration results of three junction controls: TFL, FCFS_ped and platoon_ped under different traffic volumes.

6.4 Discussion & Limitations

Our current pedestrian model exhibits a limitation in its representation, as it lacks information on individual pedestrian performance. Instead of treating pedestrians as distinct individuals, we model them as a continuous flow. Their presence is not fully simulated in SUMO. We assume that within a given time interval, all pedestrians can completely cross certain roads. Therefore, it is challenging to obtain specific metrics such as average waiting time and timing ratios during different operating phases. This lack of detailed pedestrian information hinders a deeper understanding of our pedestrian model's performance, particularly regarding the decision tree (Fig. 6.3). To gain more meaningful insights, acquiring data on individual pedestrian behaviours would be invaluable in optimising our method and exploring alternative formulations that prioritise or strike a balance between the needs of pedestrians and drivers.

Moreover, beyond the lack of pedestrian information, our research can benefit from investigating the driver perspective in more detail. Evaluating traffic performance using different metrics, such as COx and NOx emission rates, can provide deeper insights. Currently, we have a limited understanding of the emissions impact, but early results suggest that platooning may increase the emission rate due to frequent speeding up and acceleration of vehicles. However, further evidence is needed to confirm our hypothesis. To obtain this evidence, a real-time emission tracker needs to be implemented to compare the emissions of platoon follower vehicles to those of standalone vehicles.

Regarding the junction-level results, a questionable trend emerges as our platooning method fails to outperform the state-of-the-art FCFS method consistently. An example can be seen in Table 6.1 for a scenario with 8,000 veh/hr, where despite the formation of certain platoons, we do not witness substantial improvement. This observation is further corroborated in Fig. 6.5. Currently, the root cause behind this inconsistency remains unclear.

Our hypothesis is that between the traffic flow of 8,000 to 12,000 veh/hr, platoons begin to form, effectively reducing intersection delays and shortening queuing. However, with FCFS, as the traffic flow reaches 14,000 veh/hr, the junctions approach their capacity, and available traffic space becomes constrained. Consequently, some vehicles cannot be generated and enter the road as intended, leading to an underestimation of the junction delays. These delays, calculated from the travel time of vehicles on the incoming road, do not account for the delays of non-generated vehicles, thus yielding lower delay values than expected. This unaccounted discrepancy might contribute to the limited superiority of the platooning method in this scenario.

By introducing worst-case pedestrian scenarios, which involve a continuous flow of pedestrians alongside vehicular traffic, the intersection experiences extreme congestion due to combined demands. This scenario mirrors high-density urban areas like

Shibuya in Tokyo, where pedestrian volumes are significantly high. Such a scenario has a profound impact on autonomous junction controls, especially the platooning method.

Currently, our proposed pedestrian control strategy explicitly blocks certain roads for an extended time to provide reservation time slots for pedestrians. However, this approach compromises the opportunity for platoon formation, limiting the potential of our platooning method. As a result, our platooning approach may not be compatible with junctions experiencing high pedestrian demand scenarios. Even though our platooning method can outperform TFL and FCFS in many evaluated situations, further investigation is still needed. Introducing dynamic pedestrian demands, ranging from low to high, can help us identify what are the conditions that would be the most suitable for our platooning method.

Additionally, our analysis identifies another critical factor that could significantly influence the effectiveness of our proposed method: the structure of primary and secondary roads. In Athens, the primary roads are intersected by numerous smaller roads (classified as secondary roads in our study). Given that a majority of vehicles typically travel on primary roads, there is a noticeable imbalance in traffic demands between primary and secondary routes. This inequality presents an opportunity for our pedestrian control system, which heavily relies on capitalising on these imbalanced traffic demands.

However, relying exclusively on this characteristic might be overly strict. While our proposed method demonstrates improved performance compared to traditional traffic lights, its effectiveness could disappear in diverse corridor settings where the traffic dynamics change. Our concern comes from situations where pedestrian demand exceeds that of vehicles. In such scenarios, the autonomous junction, designed to fully exploit CAVs, might be less suitable than conventional traffic lights in terms of safety, practicality, cost-effectiveness, and, notably, junction traffic performance.

Chapter 7

Conclusions and Future Work

Our research is mainly driven by the challenge of optimising future traffic management efficiency and also the element of robustness and dynamism. To be specific, we aim to anticipate and realise the potential of traffic in the future where all vehicles have a high chance of becoming connected and autonomous vehicles (CAVs), considering many future-refining developments in road transportation: automation, connectivity, decarbonisation, and ride-sharing. To this end, we present several studies that address the distinct aspects of these challenges. Next, the conclusion to the findings of the studies and directions of future work are described in more detail.

7.1 Conclusions

Following the research challenges detailed in section 1.1, we have made the following contributions to the state of the art. Firstly, we propose decentralised junction management to address Research Challenge 1. The main objective is to address the bottleneck issue attributed to the high computational load at the central unit introduced in several junction management models. To be precise, we transfer most of the computational operations to the driver units instead of putting loads on the central manager. The computational operations include path prediction and conflict resolution that ensures vehicle safety. However, the junction manager is still responsible for the confirmation operation due to the concurrency issue that may happen in the real world. Even though our algorithm is decentralised, the simulation results show that our algorithm does not introduce any negative effects and is able to achieve similar performance to the centralised state-of-the-art one. Additionally, the number of exchange messages is reduced significantly by using our decentralised approach.

Secondly, we turn our attention to Research Challenge 2, namely junction management with collision avoidance. To this end, we proposed an algorithm capable of coordinating multiple vehicles with intelligent path calculation, guaranteeing safe crossing even when obstructions exist. In more detail, we build on a decentralised cell-based resource reservation approach. However, when obstructions exist, the original path prediction is no longer applicable. We introduce a new path prediction algorithm specifically designed to support the case of obstructions. This new algorithm allows vehicles to avoid collision with obstacles and reach their target (outgoing lane) while using the least possible space. Moreover, we use SUMO to evaluate our model against traffic lights and the state-of-the-art model with naive collision avoidance, which is a lane-closing solution. Three different obstruction scenarios, which are no obstruction, obstructions at the junction entrance, and obstructions in the middle of the junction, were used as study cases. The results show that our model is more robust to obstructions by maintaining throughput or flow rate up to 94-99% of the optimal performance, while traffic lights and the state-of-the-art model can maintain up to 65-90%.

Thirdly, we focus on platooning optimisation at the junction with CAVs. We propose a novel agent-based dynamic platoon formation mechanism for road traffic management for connected autonomous vehicles. This work partially addresses Research Challenge 4. Instead of having platoons formed arbitrarily without guaranteeing benefits to the system (greedy approach), the platoons should be formed by considering the impact on the other vehicles as well. To this end, we propose a principled method for calculating the benefits and costs of forming a platoon in terms of overall waiting time, allowing the optimal platoon size to be determined dynamically, minimising overall travel time and maximising junction throughput. Additionally, we improved the cell-based conflict resolution algorithm introduced by Dresner and Stone (2008) to support platooning. Our empirical experiments using SUMO show that our proposed mechanism can increase the throughput by $\approx$12% and reduce the travel time between 6 and 31% compared to a non-platoon-based state-of-the-art mechanism, depending on the traffic volume.

Fourthly, we improve the previous work even further, addressing Research Challenge 5, expanding the study scale covering a traffic corridor. This work ensures that the corridor environment is as realistic as possible by utilising the real-world traffic dataset in Athens, Greece. The dataset provides very rich information on how vehicles travel within the corridor, providing average speed, travel distance, and type of vehicles. By utilising this valuable information, the traffic model is calibrated to reproduce a real-world traffic flow considering various elements: practical road geometry, road usage, and heterogeneous vehicle types. Therefore, our proposed platooning approach can be evaluated against a highly realistic road corridor to demonstrate the impact on such a macroscopic scale. The empirical evaluation shows that typical FCFS works considerably well in light traffic. In contrast, interestingly, FCFS performs quite poorly in heavy

traffic, even worse than conventional traffic light control. On the other hand, our platooning can reduce the trip duration by up to 20% and 45% compared to traffic light control and FCFS.

Lastly, we consider an essential element of real-world traffic: pedestrians. To properly anticipate a practical situation in the urban area, Research Challenge 6 must be addressed. Unlike conventional traffic light control, many autonomous junction management models are not initially designed to support pedestrians but solely focus on maximising traffic performance (increase throughput and reduce travel time). Therefore, we introduce a waiting-time-driven model balancing the right-of-way between vehicles and pedestrians. The core of the algorithm lies in the decision tree that regularly switches operating phases, giving ways to pedestrians and vehicles while minimising both waiting times. The simulation results show that, at the single junction level, our proposed algorithm can reduce the average travel time up to 39% and 17% compared to traffic lights and FCFS. However, at the corridor level, our algorithm can perform well in light traffic scenarios by reducing the trip duration by up to 16%. Still, the out-performance disappears when it comes to heavy traffic scenarios.

7.2 Future Work

Despite these contributions, there is still ample room left for improvement. Firstly, in the proposed decentralised junction management, even though the vehicles have less reliance on the central unit by allowing the vehicles to participate in some of the computational steps, the model still needs to be a fully decentralised cell-based model. The reason is that all vehicle request messages are still passing through the junction manager mainly to prevent the concurrency issue and synchronise reservation orders. A straightforward solution could be regularly switching or selecting a new central unit once the current centre leaves the junction to become the fully decentralised method. An advanced strategy is distributing critical tasks more broadly across the system, thereby reducing dependency levels on any specific component. Furthermore, as our settings simulated no loss in exchanging messages, the difference in exchanging message numbers between FCFS and our approach purely comes from the conflicts between reservations. To realise an actual number, a different simulation must be used, such as the Objective Modular Network Testbed (OMNeT), since it is built mainly to simulate the communication units. With this, several message-exchanging issues will be considered, e.g. concurrency, causing changes in the number of messages.

Secondly, another future work lies in multi-vehicle collision avoidance junction management (Chapter 3). In this model, vehicles can work together to cross a junction while avoiding collisions with obstacles. Our empirical study has demonstrated that

the model can maintain optimal junction throughput even when four lanes are obstructed. However, the model's limitations are its narrow focus on specific obstacle contexts. Specifically, the model assumes that obstacles are fixed and located in the middle of the junction, while in reality, obstacles are not always static. Therefore, our proposed model may not be applicable in practical situations. At the time of writing, SUMO cannot deliberately place (that provides a physical meaning) an obstacle which prevents us from further investigating such situations. To address the issue of moving obstacles, researchers may consider using a new simulation tool or a customised simulation instead and also exploring methods that predict/detect obstacle movements.

Thirdly, an area of potential improvement is the dynamic platoon formation approach presented in Chapter 4. To enhance our algorithm, various traffic-related metrics can be used instead of relying solely on waiting time as a basis for calculating platoon formation benefits. Metrics such as emission rate, number of passengers, lane occupancy, and queuing length could be included in the calculation. Furthermore, the problem can be approached as a prioritisation problem since the current goal of forming platoons is to reduce delays without considering any priorities. However, certain vehicles, particularly emergency vehicles such as fire trucks, police officers, and ambulances, should be prioritised over typical commuters. Incorporating this prioritisation into the algorithm would make it better suited for real-world scenarios.

Nevertheless, in this contribution, there is another critical aspect that can enhance the assessment of our method's performance. Specifically, during the process of estimating the ETA and ECT to calculate the reduction and increase in waiting time, evaluating the accuracy of ETA and ECT estimation is possible. While the simulation results showcase reduced travel time and improved throughput, this accuracy assessment allows for a more detailed investigation of our platoon formation's efficiency from an algorithm design perspective, enabling further improvements if needed.

However, at the moment, only half of the results are available, which are the records of the calculated cost efficiency value indicating the total waiting time benefits that the junction would gain through platoon formation or extension. For example, these records comprise vectors representing the decisions made for each platoon formation, containing information on the leader, follower(s), platoon size, time step, and cost efficiency value. Regardless, these records cannot be segregated into two distinct values, namely the reduction and increase in waiting time. This lack of separation makes it difficult to trace back to the calculation origin, particularly delay estimations.

To further investigate the estimation, we need more pieces of information. When platoon formation triggers either a new formation or an extension of its size, the records should capture the estimated ETA & ECT of the potentially-joiner DA without forming the platoon and as well as the ECT when the DA joins a platoon[1]. In the case of

[1]ETA after joining the platoon is not necessary since it will be the same as the leader's ETA

Platoon formation tracker

Variable	Value type
Unique id of platoon	integer
Type	string = ”formation” or ”extension”
Size	integer = N
Leader	ID_1
Follower(s)	none or $\{\mathrm{ID}_2, \ldots, \mathrm{ID}_{N-1}\}$
Joiner	ID_N
Leader's est_ETA	float
Platoon's est_ECT before joining	float
Joiner's est_ETA	float
Joiner's est_ECT	float
Joiner's est_ECT after joining	float
Reduction in waiting time	float
Increase in waiting time or externalities	float
Affected DAs	none or $\{(\mathrm{ID}_m, \text{float}), \ldots, (\mathrm{ID}_M, \text{float})\}$

TABLE 7.1: A vector structure recorded for each platoon formation made. The first column indicates variables, while the second column indicates their value types

extending platoons, recording the original platoons' ECT and the extended platoons' ECT[2] is essential, allowing us to track the change in estimated platoon travel time. Additionally, another crucial factor is the externalities of each platoon, determining how much waiting time the junction would incur in exchange for forming/extending certain platoons. Specifically, the structure of this vector is given in Table 7.1. While most variables in the vector are self-explanatory, the "Affected DAs" variable may require further explanation. This variable refers to the DAs on different trajectories that will be affected–meaning their waiting time increases–due to the platoon, with respect to the tracker vector, where ID_m to IDM are the IDs of agents within $Aovp$, and float value type refers to their increased waiting time.

Still, these records only show the aspect from the platoon side. To comprehensively evaluate the estimation process, we need the actual ETA and ECT for all DAs traversing through junctions, providing comparable values between the actual outcome and the estimated ones. The structure of the ETA and ECT outcome is straightforward and contains only DAs' ID as an integer, arrival junction as a string, and ETA & ECT using float value types (see Table 7.2). Consequently, this approach allows us to scrutinise the accuracy of the estimation process, which is at the core of platoon formation.

Fourthly, Research Challenge 5, specifically the challenge of multi-junction platoon control, can also be addressed in different aspects. Currently, we address the realism of the environments by considering heterogeneous vehicles and practical corridor geometry, but we can also consider platoon continuity since the study already covers multiple junctions. When platoon members share a similar route, they can continue as a platoon

[2]This is equal to the joiner's estimated ECT after joining

DAs' ETA & ECT tracker

Variable	Value type
DA ID	interger
Junction ID	string
ETA	float
ECT	float

TABLE 7.2: A vector structure ETA and ECT recorded for each DA. The first column indicates variables, while the second column indicates their value types

after leaving a certain junction. This approach could further improve overall performance by reducing emission rates due to less air drag and increasing road capacity to close vehicle gaps while they are moving to different junctions. Additionally, information exchange between nearby junctions could be developed to anticipate incoming flows and optimise platoon routing. Particularly, platoon formation benefits can be calculated at the higher level rather than the junction level. These aspects deserve further attention and exploration in future research.

Lastly, while we have made progress in addressing Research Challenge 6, there is still room for improvement in considering pedestrians in our work. Our empirical evaluation only focuses on the driver's perspective, specifically delays. Different metrics should be used to determine the performance of the method from the pedestrian perspective as well. For instance, calculating average pedestrian waiting time or throughput results can provide useful insights to improve our proposed solution even further. Furthermore, an in-depth investigation of the results and method operation is also possible to determine why the out-performance of our method against traffic light control disappears in extreme traffic scenarios at the corridor level. Understanding the actual reasons behind this phenomenon is crucial to further improving our proposed solution and ensuring that it is effective in various real-world scenarios. Therefore, future research efforts should be directed towards developing more comprehensive models that account for both pedestrian and driver perspectives and investigating pedestrian crossing operations in more detail.

However, it is important to acknowledge a major assumption made in this thesis: the presumption that all vehicles on the road are CAVs. Given the predicted timeline for CAV total adoption (expected around the 2080s, as indicated by Litman (2020)), it is realistic to anticipate a transition period creating a mixed-traffic environment between human-driven vehicles (HDVs) and CAVs. This transition is a critical aspect and a significant area of research focus. Understanding how to effectively integrate CAVs in the midst of HDVs is vital. This involves investigating the impacts of different penetration rates of CAVs (Aoki and Rajkumar (2019, 2022)) and establishing protocols (Sharon and Stone (2017)). Hence, it would be inconsiderate to overlook the potential impact of such mixed-traffic scenarios on our proposed approach.

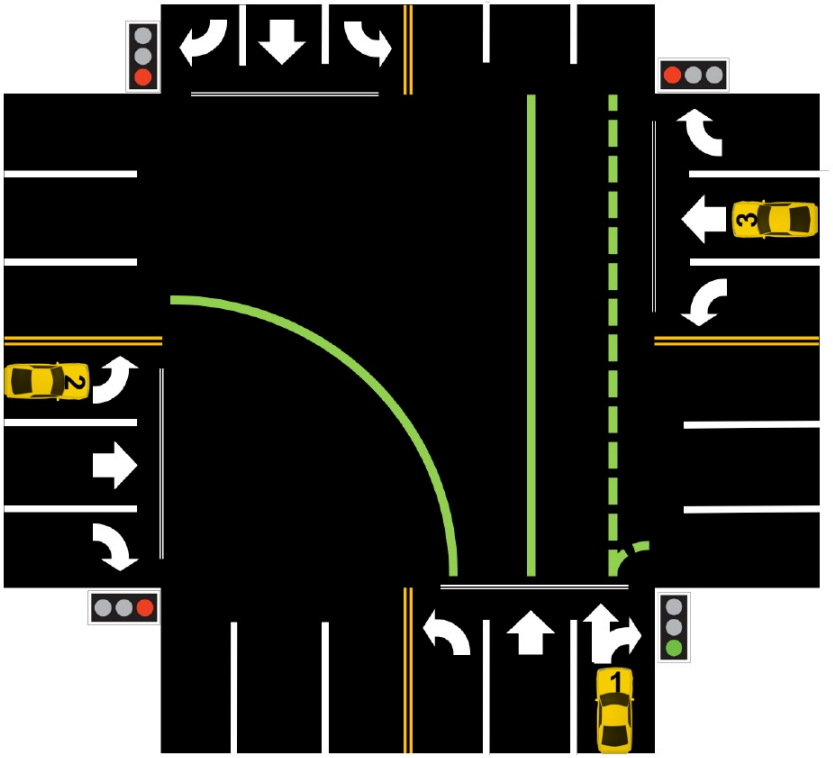

FIGURE 7.1: The example red and green traffic signals notify the HDV's right of way. Source: (Sharon and Stone, 2017)

Two key considerations arise regarding the impact of mixed-traffic scenarios on our method. The first point of impact is resource reservation. Protocols like the one proposed by Sharon and Stone (2017) and Aoki and Rajkumar (2022) suggest the need for external signals or indicators to notify HDVs of crossing timings at junctions. They utilise red-green signals to manage the right of way for HDVs; see the example in Fig. 7.1. This directly influences how DAs perform resource reservation in our proposed method. The majority of time slots will likely be reserved for HDVs, especially when the red-green signal is frequently switching phases. It is highly possible to enhance our resource reservation system to include traffic signals, allowing DAs to navigate the junction while avoiding conflicts with the green-light stream. However, this adjustment might lead to sacrifices in junction performance metrics such as throughput and travel delays. Green-light traffic streams often align with straight-going trajectories, a significant part of vehicle routing, potentially causing a reduction in throughput and increased travel delays. Higher proportions of HDVs exacerbate this drop in junction performance compared to a scenario with 100% CAVs. While there is still some traffic space for automated resource reservation, maintaining an optimal traffic flow becomes challenging.

On the positive side, having red-green signals dedicated to HDVs does offer advantages, particularly in terms of coping with pedestrians. For instance, as illustrated in

Fig. 7.1, the green phase creates an opportunity for pedestrians to have the right of way on the East side of the junction. Leveraging the blockage caused by green phases can effectively reduce overall pedestrian waiting times. Thus, the protocol for mixed traffic presents both pros and cons for our approach. Undoubtedly, it improves the junction control's flexibility, increasing its availability for various road users and enhancing its practicality for near-future traffic scenarios, anticipating a transition period rather than expecting 100% CAVs in 60 years. However, a primary concern is the resemblance between the mixed-traffic protocol and conventional signalised junction control. It remains essential to investigate the performance of the mixed-traffic protocol compared to a range of existing traffic signal approaches, spanning from simple to advanced methods, such as SCAT (Sims and Dobinson, 1980).

Moreover, another key consideration revolves around our dynamic platoon formation algorithm. Within this algorithm, the cost efficiency of platoons is calculated based on the ETA and ECT of all DAs as they approach and traverse the junction. However, a significant challenge emerges with HDVs as their information is rather limited. Specifically, the ETA and ECT of HDVs remain unknown (for a large part), hindering the algorithm's ability to estimate the impact of platoon formation on HDVs. Ensuring the benefits of platoons without this crucial information becomes challenging. Moreover, determining the optimal size of platoons becomes a complicated task due to this informational gap. Fundamentally, the lack of information regarding HDVs poses a major difficulty to our approach, given its heavy reliance on the determinism of CAV movements.

In our analysis, we propose a strategy to facilitate the functioning of our platooning method in a mixed-traffic environment. We suggest treating red-green phases as substantial reservation time slots, effectively occupying a large crossing space from the inbound road's exit to the outbound road's entrance. For HDVs, the ECT would correspond to the moment when the green phases end. By utilising this approach, our platooning method can estimate waiting time costs or externalities for HDVs, enabling effective platoon formation. However, a potential concern with this strategy is the possibility of excessive reservation when employing these extensive time slots. HDVs might not require their full slots to completely cross the junction. Consequently, our dynamic platoon formation algorithm may suffer from over-reservation for HDVs, resulting in sub-optimal platoon benefits. This strategy would strongly suit the situation with a low CAV penetration rate.

In high CAV penetration rate situations, typically exceeding 70%, the capabilities of both CAVs and platooning may be greatly constrained. This happens because the green phases provide an excessive window for HDVs that does not align with the actual demand for such vehicles. Practically, a constant value for the CAV penetration rate can only be expected in a small study area; however, in a road network, this parameter

will vary. Different areas may manifest varying penetration rates. Developing a dynamic system that reacts to HDV demands would enhance its availability, scalability, and practicality, and it can be done under the condition that the number of HDVs is accurately estimated.

This necessitates further investigation into how our dynamic platoon formation, which heavily relies on the determinism of vehicle trajectories and crossing times, can cope with the existence of HDVs in an environment where their information is rather limited. Moreover, we need to explore how to dynamically adapt to the actual HDV demand at individual junctions, especially in a scenario where CAV penetration rates vary across different areas.

Appendix A

Publications

The following is a list of papers that have arisen from this research.

1. Phuriwat Worrawichaipat, Enrico Gerding, Ioannis Kaparias, and Sarvapali Ramchurn. Resilient intersection management with multi-vehicle collision avoidance. Frontiers in Sustainable Cities, 3:670454, 2021

2. Phuriwat Worrawichaipat, Enrico Gerding, Ioannis Kaparias, and Sarvapali Ramchurn. Multi-agent signal-less intersection management with dynamic platoon formation. In Proceeding of the 22nd International Conference on Autonomous Agents and Multi-agents, AAMAS '23, 2022. *Accepted as a full paper*

3. Phuriwat Worrawichaipat, Enrico Gerding, Ioannis Kaparias, and Sarvapali Ramchurn. Multi-agent signal-less intersection management with dynamic platoon formation and consideration of pedestrians. ACM Journal on Autonomous Transportation Systems (JATS), *Work in progress*

Bibliography

Hossam Abdelgawad, Baher Abdulhai, and Mohamed Wahba. Multiobjective optimization for multimodal evacuation. *Transportation Research Record*, 2196(1):21–33, 2010.

Baher Abdulhai, Rob Pringle, and Grigoris J Karakoulas. Reinforcement learning for true adaptive traffic signal control. *Journal of Transportation Engineering*, 129(3):278–285, 2003.

Rahmi Akcelik. Estimation of green times and cycle time for vehicle-actuated signals. *Transportation research record*, 1457:63–72, 1994. ISSN 0361-1981. URL https://onlinepubs.trb.org/Onlinepubs/trr/1994/1457/1457-008.pdf.

Maria Alonso Raposo, Biagio Ciuffo, Fulvio Ardente, Jean Philippe Aurambout, Gianmarco Baldini, Robert Braun, Panayotis Christidis, Aris Christodoulou, Amandine Duboz, Sofia Felici, et al. The future of road transport. Technical report, Joint Research Centre (Seville site), 2019.

Shunsuke Aoki and Ragunathan Rajkumar. V2v-based synchronous intersection protocols for mixed traffic of human-driven and self-driving vehicles. In *2019 IEEE 25th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, pages 1–11. IEEE, 2019.

Shunsuke Aoki and Ragunathan Rajkumar. Safe intersection management with cooperative perception for mixed traffic of human-driven and autonomous vehicles. *IEEE Open Journal of Vehicular Technology*, 3:251–265, 2022. .

James Arbib and Tony Seba. Rethinking transportation 2020-2030: The disruption of transportation and the collapse of the internal-combustion vehicle and oil industries, 2017. URL http://static1.squarespace.com/static/585c3439be65942f022bbf9b/t/591a2e4be6f2e1c13df930c5/1494888038959/RethinkX+Report_051517.pdf.

HM Abdul Aziz and Satish V Ukkusuri. Unified framework for dynamic traffic assignment and signal control with cell transmission model. *Transportation Research Record*, 2311(1):73–84, 2012.

Emmanouil Barmpounakis and Nikolas Geroliminis. On the new era of urban traffic monitoring with massive drone data: The pneuma large-scale field experiment. *Transportation research part C: emerging technologies*, 111:50–71, 2020.

Masoud Bashiri and Cody H Fleming. A platoon-based intersection management system for autonomous vehicles. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 667–672. IEEE, 2017.

Masoud Bashiri, Hassan Jafarzadeh, and Cody H Fleming. Paim: Platoon-based autonomous intersection management. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 374–380. IEEE, 2018.

Christopher Beard and Athanasios Ziliaskopoulos. System optimal signal optimization formulation. *Transportation research record*, 1978(1):102–112, 2006.

Luis Conde Bento, Ricardo Parafita, and Urbano Nunes. Intelligent traffic management at intersections supported by v2v and v2i communications. In *2012 15th International IEEE Conference on Intelligent Transportation Systems*, pages 1495–1502, Sep. 2012. .

Carl Bergenhem, Qihui Huang, Ahmed Benmimoun, and Tom Robinson. Challenges of platooning on public motorways. In *17th world congress on intelligent transport systems*, pages 1–12, 2010.

Neha Bisht and Rahi Avinash Shet. Platoon-based cooperative intersection management strategies. In *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*, pages 1–6. IEEE, 2020.

Herib Blanco, Jonatan J. Gómez Vilchez, Wouter Nijs, Christian Thiel, and André Faaij. Soft-linking of a behavioral model for transport with energy system cost optimization applied to hydrogen in eu. *Renewable and Sustainable Energy Reviews*, 115:109349, 2019. ISSN 1364-0321. . URL http://www.sciencedirect.com/science/article/pii/S136403211930557X.

SC Calvert, G Klunder, JLL Steendijk, and M Snelder. The impact and potential of cooperative and automated driving for intelligent traffic signal corridors: A field-operational-test and simulation experiment. *Case studies on transport policy*, 8(3):901–919, 2020.

Zhiguang Cao, Siwei Jiang, Jie Zhang, and Hongliang Guo. A unified framework for vehicle rerouting and traffic light control to reduce traffic congestion. *IEEE transactions on intelligent transportation systems*, 18(7):1958–1973, 2016.

Dustin Carlino, Stephen D Boyles, and Peter Stone. Auction-based autonomous intersection management. In *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*, pages 529–534, Oct 2013. .

Lei Chen and Cristofer Englund. Cooperative intersection management: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 17(2):570–586, Feb 2016. ISSN 1558-0016. .

Rongsheng Chen, Jeffrey Hu, Michael W Levin, and David Rey. Stability-based analysis of autonomous intersection management with pedestrians. *Transportation research part C: emerging technologies*, 114:463–483, 2020.

Mario Collotta, Lucia Lo Bello, and Giovanni Pau. A novel approach for dynamic traffic lights management based on wireless sensor networks and multiple fuzzy logic controllers. *Expert Systems with Applications*, 42(13):5403 – 5415, 2015. ISSN 0957-4174. . URL http://www.sciencedirect.com/science/article/pii/S0957417415001104.

European Commission. Towards clean, competitive and connected mobility: The contribution of transport research and innovation to the mobility package swd/2017/0223 final, 2017.

Denise de Oliveira, Paulo Roberto Ferreira, Ana LC Bazzan, and Franziska Klügl. A swarm-based approach for selection of signal plans in urban scenarios. In *International Workshop on Ant Colony Optimization and Swarm Intelligence*, pages 416–417. Springer, 2004.

Kurt Dresner. Autonomous intersection management project, 2006. URL http://www.cs.utexas.edu/~aim/. Accessed: 2019-11-10.

Kurt Dresner and Peter Stone. Human-usable and emergency vehicle-aware control policies for autonomous intersection management. In *Fourth International Workshop on Agents in Traffic and Transportation (ATT), Hakodate, Japan*, 2006.

Kurt Dresner and Peter Stone. A multiagent approach to autonomous intersection management. *Journal of Artificial Intelligence Research 31*, 31:591–656, 2008. .

Pedro Fernandes and Urbano Nunes. Algorithms for management of a multi-platooning system of ivc-enabled autonomous vehicles, with high traffic capacity. In *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 1935–1941. IEEE, 2011.

Tarak Gandhi and Mohan M Trivedi. Pedestrian collision avoidance systems: A survey of computer vision based recent studies. In *2006 IEEE Intelligent Transportation Systems Conference*, pages 976–981. IEEE, 2006.

Tarak Gandhi and Mohan Manubhai Trivedi. Pedestrian protection systems: Issues, survey, and challenges. *IEEE Transactions on intelligent Transportation systems*, 8(3): 413–430, 2007.

Dariu M Gavrila. Sensor-based pedestrian protection. *IEEE Intelligent Systems*, 16(6): 77–81, 2001.

Jeffery B. Greenblatt and Susan Shaheen. Automated vehicles, on-demand mobility, and environmental impacts. *Current Sustainable/Renewable Energy Reports*, 2(3):74–81, Sep 2015. ISSN 2196-3010. . URL https://doi.org/10.1007/s40518-015-0038-5.

Tsin Hing Heung, Tin Kin Ho, and Yu Fai Fung. Coordinated road-junction traffic control by dynamic programming. *IEEE Transactions on Intelligent Transportation Systems*, 6(3):341–350, 2005.

PB Hunt, DI Robertson, RD Bretherton, and M Cr Royle. The scoot on-line traffic signal optimisation technique. *Traffic Engineering & Control*, 23(4), 1982.

G Improta and GE Cantarella. Control system design for an individual signalized junction. *Transportation Research Part B: Methodological*, 18(2):147–167, 1984.

Jirı Iša, Julian Kooij, Rogier Koppejan, and Lior Kuijer. Reinforcement learning of traffic light controllers adapting to accidents. *Design and Organisation of Autonomous Systems*, pages 1–14, 2006.

Qiu Jin, Guoyuan Wu, Kanok Boriboonsomsin, and Matthew Barth. Platoon-based multi-agent intersection management for connected vehicle. In *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*, pages 1462–1467. IEEE, 2013.

Khewal Bhupendra Kesur. Generating more equitable traffic signal timing plans. *Transportation Research Record*, 2192(1):108–115, 2010.

Shinya Kikuchi and Nopadon Kronprasert. Determining lengths of left-turn lanes at signalized intersections under different left-turn signal schemes. *Transportation research record*, 2195(1):70–81, 2010.

Yongkuk Kim and SangJoo Kwon. A heuristic obstacle avoidance algorithm using vanishing point and obstacle angle. *Intelligent Service Robotics*, 8(3):175–183, Jul 2015. ISSN 1861-2784. . URL https://doi.org/10.1007/s11370-015-0171-4.

Sirisha Murthy Kothuri. *Exploring Pedestrian Responsive Traffic Signal Timing Strategies in Urban Areas*. PhD thesis, Portland State University, 2014.

Daniel Krajzewicz, Georg Hertkorn, Christian Rössel, and Peter Wagner. Sumo (simulation of urban mobility)-an open-source traffic simulation. In *Proceedings of the 4th middle East Symposium on Simulation and Modelling (MESM20002)*, pages 183–187, 2002.

Stefan Krauß, Peter Wagner, and Christian Gawron. Metastable states in a microscopic model of traffic flow. *Physical Review E*, 55(5):5597, 1997.

Vijay Krishna. *Auction theory*. Academic press, 2009.

Ji-Wook Kwon and Dongkyoung Chwa. Adaptive bidirectional platoon control using a coupled sliding mode control method. *IEEE Transactions on Intelligent Transportation Systems*, 15(5):2040–2048, 2014.

Phil LeBeau. Relax, experts say it's at least a decade before you can buy a self-driving vehicle, 2019. URL https://www.cnbc.com/2019/07/29/experts-say-its-at-least-a-decade-before-you-can-buy-a-self-driving-car.html. Accessed: 2019-11-10.

Pengfei Li, Montasir M Abbas, Raghu Pasupathy, and Larry Head. Simulation-based optimization of maximum green setting under retrospective approximation framework. *Transportation Research Record*, 2192(1):1–10, 2010.

Xiu-ping Li, Zhi-yong Liu, and Jin-pei Wu. An approach of intersection traffic signal control. *Hsin hsi yu k'ung chih = Information and control. Vol. 30, no. 1*, 2001.

Kuo-Yun Liang. *Coordination and routing for fuel-efficient heavy-duty vehicle platoon formation*. PhD thesis, KTH Royal Institute of Technology, 2014.

Kuo-Yun Liang, Jonas Mårtensson, and Karl H Johansson. Heavy-duty vehicle platoon formation for fuel efficiency. *IEEE Transactions on Intelligent Transportation Systems*, 17 (4):1051–1061, 2015.

Peiqun Lin, Jiahui Liu, Peter J Jin, and Bin Ran. Autonomous vehicle-intersection coordination method in a connected vehicle environment. *IEEE Intelligent Transportation Systems Magazine*, 9(4):37–47, winter 2017. ISSN 1941-1197. .

Wei-Hua Lin and Chenghong Wang. An enhanced 0-1 mixed-integer lp formulation for traffic signal control. *IEEE Transactions on Intelligent transportation systems*, 5(4): 238–245, 2004.

Jennie Lioris, Ramtin Pedarsani, Fatma Yildiz Tascikaraoglu, and Pravin Varaiya. Doubling throughput in urban roads by platooning. *IFAC-PapersOnLine*, 49(3):49–54, 2016.

Todd Litman. Autonomous vehicle implementation predictions: Implications for transport planning. 2020.

Kai Liu, Edward Chan, Victor Lee, Krasimira Kapitanova, and Sang H. Son. Design and evaluation of token-based reservation for a roadway system. *Transportation Research Part C: Emerging Technologies*, 26:184 – 202, 2013. ISSN 0968-090X. . URL http://www.sciencedirect.com/science/article/pii/S0968090X12001118.

Hong K Lo. A novel traffic signal control formulation. *Transportation Research Part A: Policy and Practice*, 33(6):433–448, 1999.

Nic Lutsey. Modernizing vehicle regulations for electrification, 2018. URL https://theicct.org/publications/modernizing-regulations-electrification.

Mohammad Mamouei, Ioannis Kaparias, and George Halikias. A framework for user- and system-oriented optimisation of fuel efficiency and traffic flow in adaptive cruise control. *Transportation Research Part C: Emerging Technologies*, 92:27–41, 2018. ISSN 0968-090X. . URL https://www.sciencedirect.com/science/article/pii/S0968090X18301475.

Patrice Marcotte. Network optimization with continuous control parameters. *Transportation Science*, 17(2):181–197, 1983.

Stefano Masi, Philippe Xu, and Philippe Bonnifait. Adapting the virtual platooning concept to roundabout crossing. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 1366–1372. IEEE, 2018.

Alejandro Ivan Morales Medina, Nathan Van De Wouw, and Henk Nijmeijer. Cooperative intersection control based on virtual platooning. *IEEE Transactions on Intelligent Transportation Systems*, 19(6):1727–1740, 2017.

Pitu Mirchandani and Larry Head. A real-time traffic signal control system: architecture, algorithms, and analysis. *Transportation Research Part C: Emerging Technologies*, 9 (6):415–432, 2001.

Y Sazi Murat and Ergun Gedizlioglu. A fuzzy logic multi-phased signal control model for isolated junctions. *Transportation Research Part C: Emerging Technologies*, 13(1):19–36, 2005.

Rolf Naumann, Rainer Rasche, Jürgen Tacken, and Christoph Tahedl. Validation and simulation of a decentralized intersection collision avoidance algorithm. In *Proceedings of Conference on Intelligent Transportation Systems*, pages 818–823, Nov 1997. .

Rolf Naumann, Rainer Rasche, and Jürgen Tacken. Managing autonomous vehicles at intersections. *IEEE Intelligent Systems and their Applications*, 13(3):82–86, May 1998. ISSN 2374-9423. .

New York State Department of Transportation. Traffic data viewer, 2016. URL https://www.dot.ny.gov/tdv. Accessed: 2019-11-10.

Tanja Niels, Nikola Mitrovic, Nemanja Dobrota, Klaus Bogenberger, Aleksandar Stevanovic, and Robert Bertini. Simulation-based evaluation of a new integrated intersection control scheme for connected automated vehicles and pedestrians. *Transportation research record*, 2674(11):779–793, 2020.

Sinan Öncü, Jeroen Ploeg, Nathan Van de Wouw, and Henk Nijmeijer. Cooperative adaptive cruise control: Network-aware analysis of string stability. *IEEE Transactions on Intelligent Transportation Systems*, 15(4):1527–1537, 2014.

C. P. Pappis and E. H. Mamdani. A fuzzy logic controller for a trafc junction. *IEEE Transactions on Systems, Man, and Cybernetics*, 7(10):707–717, Oct 1977. ISSN 2168-2909. .

G. Pudics, M. Z. Szabó-Resch, and Z. Vámossy. Safe robot navigation using an omnidirectional camera. In *2015 16th IEEE International Symposium on Computational Intelligence and Informatics (CINTI)*, pages 227–231, Nov 2015. .

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

Amir Rasouli and John K Tsotsos. Autonomous vehicles that interact with pedestrians: A survey of theory and practice. *IEEE transactions on intelligent transportation systems*, 21(3):900–918, 2019.

Karima Rebai, A Benabderrahmane, Ouahiba Azouaoui, and Noureddine Ouadah. Moving obstacles detection and tracking with laser range finder. In *2009 International Conference on Advanced Robotics*, pages 1–6, June 2009.

Andrey V. Savkin and Hang Li. A safe area search and map building algorithm for a wheeled mobile robot in complex unknown cluttered environments. *Robotica*, 36(1): 96–118, 2018. .

Andrey V. Savkin and Chao Wang. Seeking a path through the crowd: Robot navigation in unknown dynamic environments with moving obstacles based on an integrated environment representation. *Robotics and Autonomous Systems*, 62(10):1568 – 1580, 2014. ISSN 0921-8890. . URL http://www.sciencedirect.com/science/article/pii/S0921889014000955.

Heiko Schepperle, Klemens Böhm, and Simone Forster. Traffic management based on negotiations between vehicles–a feasibility demonstration using agents. In *Agent-Mediated Electronic Commerce and Trading Agent Design and Analysis*, pages 90–104. Springer, 2007.

Benjamin Schneider. Traffic's mind-boggling economic toll. *Bloomberg Citilab*, 2018.

Michele Segata, Stefan Joerer, Bastian Bloessl, Christoph Sommer, Falko Dressler, and Renate Lo Cigno. Plexe: A platooning extension for veins. In *2014 IEEE Vehicular Networking Conference (VNC)*, pages 53–60. IEEE, 2014.

Guni Sharon and Peter Stone. A protocol for mixed autonomous and human-operated vehicles at intersections. In Gita Sukthankar and Juan A. Rodriguez-Aguilar, editors, *Autonomous Agents and Multiagent Systems*, pages 151–167, Cham, 2017. Springer International Publishing. ISBN 978-3-319-71682-4.

Steven E Shladover. Path at 20—history and major milestones. *IEEE Transactions on intelligent transportation systems*, 8(4):584–592, 2007.

Steven E Shladover, Dongyan Su, and Xiao-Yun Lu. Impacts of cooperative adaptive cruise control on freeway traffic flow. *Transportation Research Record*, 2324(1):63–70, 2012.

Olger Siebinga. Travia: a traffic data visualization and annotation tool in python. *Journal of Open Source Software*, 6(65):3607, 2021. . URL https://doi.org/10.21105/joss.03607.

Arthur G Sims and Kenneth W Dobinson. The sydney coordinated adaptive traffic (scat) system philosophy and benefits. *IEEE Transactions on Vehicular Technology*, 29 (2):130–137, May 1980. ISSN 1939-9359. .

James C. Spall and Daniel C. Chin. Traffic-responsive signal timing for system-wide traffic control. *Transportation Research Part C: Emerging Technologies*, 5(3):153 – 163, 1997. ISSN 0968-090X. . URL http://www.sciencedirect.com/science/article/pii/S0968090X97000120.

Daniel Sperling. Electric vehicles: Approaching the tipping point. In *Three Revolutions*, pages 21–54. Springer, 2018.

The Department for Transport under licence from the Controller of Her Majesty's Stationery Office. Traffic signs manual. 2019. URL https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/851465/dft-traffic-signs-manual-chapter-6.pdf.

The European Commission, Standard & Poor's DRI and KULeuven. The aopii cost-effectiveness study part iii: The transport base case, 1999.

Mohamed B Trabia, Mohamed S Kaseko, and Murali Ande. A two-stage fuzzy logic controller for traffic signals. *Transportation Research Part C: Emerging Technologies*, 7 (6):353–367, 1999.

Matteo Vasirani and Sascha Ossowski. A market-inspired approach for intersection management in urban road traffic networks. *Journal of Artificial Intelligence Research*, 43:621–659, 2012.

Huan Vu, Samir Aknine, and Sarvapali D. Ramchurn. A decentralised approach to intersection traffic management. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 527–533. International Joint Conferences on Artificial Intelligence Organization, 7 2018. . URL https://doi.org/10.24963/ijcai.2018/73.

Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

Axel Wegener, Michał Piórkowski, Maxim Raya, Horst Hellbrück, Stefan Fischer, and Jean-Pierre Hubaux. Traci: an interface for coupling road traffic and network simulators. In *Proceedings of the 11th communications and networking simulation symposium*, pages 155–163, 2008.

MA Wiering. Multi-agent reinforcement learning for traffic light control. In *Machine Learning: Proceedings of the Seventeenth International Conference (ICML'2000)*, pages 1151–1158, 2000.

Michael Wooldridge. *An Introduction to MultiAgent Systems*. Wiley Publishing, 2nd edition, 2009. ISBN 0470519460, 9780470519462.

Phuriwat Worrawichaipat, Enrico Gerding, Ioannis Kaparias, and Sarvapali Ramchurn. Resilient intersection management with multi-vehicle collision avoidance. *Frontiers in Sustainable Cities*, 3:670454, 2021.

Phuriwat Worrawichaipat, Enrico Gerding, Ioannis Kaparias, and Sarvapali Ramchurn. Multi-agent signal-less intersection management with dynamic platoon formation. In *Proceeding of the 22nd International Conference on Autonomous Agents and Multiagent, AAMAS '23*, pages 1542–1550, 2023.

Wei Wu, Yang Liu, Wei Hao, George A Giannopoulos, and Young-Ji Byon. Autonomous intersection management with pedestrians crossing. *Transportation research part C: emerging technologies*, 135:103521, 2022.

Dong-Fan Xie, Xiao-Mei Zhao, and Zhengbing He. Heterogeneous traffic mixing regular and connected vehicles: Modeling and stabilization. *IEEE Transactions on Intelligent Transportation Systems*, 20(6):2060–2071, 2019. .

Jun Zhong, Huan Zhou, Yan Lin, and Fangxiao Ren. The impact of ride-hailing services on the use of traditional taxis: Evidence from chinese urban panel data. *IET Intelligent Transport Systems*, 16(11):1611–1622, 2022.

Anye Zhou, Srinivas Peeta, Menglin Yang, and Jian Wang. Cooperative signal-free intersection control using virtual platooning and traffic flow regulation. *Transportation research part C: emerging technologies*, 138:103610, 2022.

Ismail H. Zohdy and Hesham A. Rakha. Intersection management via vehicle connectivity: The intersection cooperative adaptive cruise control system concept. *Journal of Intelligent Transportation Systems*, 20(1):17–32, 2016. . URL https://doi.org/10.1080/15472450.2014.889918.