Coordinating Autonomous Vehicles to Reduce Traffic Congestion

Oliver Shave

Abstract

This dissertation explores solving how to achieve coordination of autonomous vehicles (AVs) through using available literature and ideas. With the many types of ways to coordinate vehicles, the project will primarily focus on AVs platooning. Platooning involves having vehicles travel in groups with small distances between themselves (usually less than a metre). The outcome of the project will be an architecture that has been designed and programmed for coordinating AVs simulated in the CARLA simulator. The purpose of exploring this topic is to determine whether the coordination architecture reduces traffic congestion by a significant amount by increasing the capacity on the roads. Additionally, the architecture has been implemented with multi-lane merging to investigate the stability of including advanced platooning manoeuvres.

Contents

1. Introduction

1.1 Problem description

Traffic congestion is a normal phenomenon in transportation and can be defined as the roadway condition in which travel time or delay is over that normally incurred under light or free-flow traffic conditions [21]. It is a large problem in the urban world and is currently on the rise globally as seen in the TomTom traffic index from 2017 to 2019 [13]. Results from 2020 would not be used as the massive decrease in traffic congestion is related to the effects of the Covid-19 virus [14]. The cause of traffic congestion can be related to many reasons, such as infrastructure being created to suit the traffic conditions at the time [12] with minimum consideration for the growth of vehicles entering the roads, and the discoordination between human drivers. An example of a multi-day long gridlock caused by the overcapacity of vehicles was the 2010 China National Highway 110 traffic jam. The highway was built to withstand roughly 10,000 cars per day, but in a year the daily volume increased from 6,000 to 14,000 cars per day [16]. When all known prevention measures are in place, due to discoordination between drivers, the phenomenon of the phantom traffic jam can still frequently take place. This phenomenon is when without the effect of an accident or lane closures, a cluster of vehicles on a road can come to a complete stop and then continue smoothing driving soon after without any obvious reason. This phenomenon is more likely to occur when the road is very busy, and the traffic flow is near saturation [8].

Traffic congestion is widely perceived as one of the most frustrating problems in urban places, it is seen to be a recurring event that has negative economic, environmental, and social costs for residents. Economically, travellers experience long commuting times which reduces workforce productivity, especially for companies that rely on transporting goods, and excessive operating costs from the increase of fuel consumption [10], in the United States, congestion cost cities over $88 billion in 2019 [14]. With the increase of fuel consumption, more greenhouse gas emissions are expelled into the environment, in Mumbai, it was found that approximately 51% more travel time under congested conditions could be significantly responsible towards the 53% more CO2 emissions [11]. Socially, an increase in traffic congestion has linked to an increase in stress for drivers which consequences include drivers getting aggressive. One of the reasons traffic congestions causes stress is due to the time urgency of the drivers [15].

1.2 Motivation

With the issue stated, traffic congestion is a growing problem that can be solved by increasing the capacity of roads and coordinating vehicles. Rebuilding road infrastructure to increase road capacity would be an expensive and unpractical project, therefore this dissertation will explore the solution of implementing coordination techniques in vehicles, more specifically platooning. To achieve coordination, connected and autonomous vehicles (CAVs) will be required, these are autonomous vehicles that can communicate with others of their type. Additionally, CAVs will have to share the same knowledge and procedures of coordination techniques to allow themselves to avoid accidents and coordinate efficiently.

For communication, CAVs will need to be installed with an On-Board Unit (OBU) which is a broad term for a device that allows for two-way short-range communication between other CAVs and infrastructure. This device can be integrated into the vehicle by the original equipment manufacturer or be an aftermarket device that is added to the vehicle after its original assembly [34]. For safety reasons, the aftermarket device should be authorised by the manufacturers due to the potential crashes that could happen if the device and vehicle are not fully compatible.

For coordination techniques such as platooning, a set of manoeuvres need to be implemented to form, merge, and split platoons, as well as allow members to leave and new vehicles to join the platoons. Additionally, it needs to be decided whether the route, speed, and positioning of platoon members will be controlled by themselves or their leader. In the case each platoon has a leader, a process of deciding which platoon member becomes the captain is required.

1.3 Structure of the report

These sections outline the structure of the dissertation to lead the reader through the process of the project.

In section 2, a review of papers related to coordination between autonomous vehicles is conducted to understand the current scope of the topic and the methods available to simulate coordination to produce reliable results.

In section 3, the development process of designing and implementing a platooning architecture is shown which is influenced by the background research.

In section 4, the single-lane platooning architecture is tested against a benchmark of autonomous vehicles and the stability of multi-lane merging I tested with a discussion of the results.

In section 5, the software development lifecycle is presented and reflected on.

In section 6, the software testing methodology is shown, and examples of the tests carried out are provided.

In section 7, a critical evaluation is completed to look at what has been done, and how it was achieved, as well as lessons learnt from the process.

In section 8, a conclusion is presented with the final discussion of the results as well as the direction which can be taken to improve the future of coordination of autonomous vehicles.

2. Background

2.1 Literature review

2.1.1 State of Autonomous Vehicles

In the current autonomous vehicle climate, coordination techniques such as platooning could not be implemented due to the involvement of human drivers still operating the vehicle. To understand where autonomous vehicles are at, a classification system developed by SAE can show what level autonomous driving is on [35]. These levels range from 0 (fully manual) to 5 (fully autonomous), autonomous vehicles now are in a transitioning phase from level 2 to level 3. With level 2, autonomous vehicles provide steering and braking assistance, but the driver is still in control of the vehicle. Level 3 would just require the driver to take control when the vehicle requests. A level 3 system can be seen in Honda's new model's traffic jam pilot, which is equipped in a limited number of vehicles, but it is limited to driving in congested traffic on an expressway when travelling slower than 50 kilometres per hour [36].

2.1.2 Communication

Simulating technology

For vehicle-to-vehicle (V2V) communication systems to operate, a vehicular channel needs to perform well in different environments, preferably in an urban scenario for this project. This introduces the challenge of simulating a model of a vehicular channel that is efficient and accurate. Vehicles in the simulation are required to have receivers and antennas that imitate the physical version.

Different types of models that range in computation complexity and accuracy can be chosen to represent vehicular channels. A model using a stochastic or analytical approach where the radio propagation characteristics are modelled by the mean of statistical techniques is beneficial for large-scale simulations due to a low computational load but cannot model site-specific channel characteristics. A geometry-based channel model has a trade-off computation complexity and accuracy as it represents the propagation environment by using information from commercial and publicly available sources. Empirical models use methods such as 3D ray tracing to compute accurate propagation characteristics of a specific environment but suffers from higher computational complexity [23].

VANET

For vehicles to coordinate, communication is required as instructions and plans need to be sent between the vehicles. Vehicular ad hoc networks (VANETs) which use vehicles as mobile nodes are a sub-class of mobile ad hoc networks (MANETs) that enable vehicles to act as nodes in a large network, with each node being able to forward data to others. An example would be a vehicle sending requests to other vehicles around the area to get updated information on open parking slots. Several challenges come with VANETS such as the potentially large scale and high mobility of the nodes as vehicles often travel

at high speeds and regularly change directions. This high mobility would cause the links between nodes to connect and disconnect very often as vehicles would come in and out of each other's communication range [27]. For this reason, a routing protocol that works well with the nature of nodes in a VANET needs to be chosen. The routing protocols of VANETS can be classified into five different categories: broadcasting protocols, route-discovery protocols, position-based protocols, clustering-based protocols, and infrastructure-based protocols. Out of all the protocols, cluster-based protocols seem the most promising as they capture the mobility of VANET nodes naturally and provide relatively stable units [6]. VANET clustering algorithms are used to associate vehicles (nodes) into groups (clusters) according to a set of rules and selecting a cluster head to mediate with the group and the rest of the network. Clustering algorithms have a series of fundamental procedures for forming and maintaining clusters. This includes neighbourhood discovery, cluster head selection, affiliation, announcement, and maintenance [26].

Anatomy of MoZo architecture

A method for pure V2V communication is by using a moving zone-based routing protocol VANET. The Moving-Zone-based (MoZo) architecture consists of multiple moving zones that are formed by vehicles with similar movement patterns [26]. Vehicles are placed into one of three classes: captain, member, or roaming. Vehicles in a roaming state are not a part of the VANET, this happens when a vehicle either enters the road or splits from a cluster zone. In this state the vehicle searches for a zone to join based on its location and direction, once zones have been found, it joins one of them based on a similarity score. In the case where no zones are found, the vehicle itself will become a captain and create its zone.

Member vehicles are a part of a zone and communicate with their captain unless instructed to communicate with other vehicles.

Captains control the merging of zones, splitting of vehicles, movement of messages, and reassignment of a new captain. Once a captain, the vehicle will create a Combined Location and Velocity tree (CLV-tree) which is a hybrid of a B+ -tree and a hash table, and a Leaving Event (LE) queue. The B+ -tree contains information about each member vehicle with their index key being their distance till the next intersection. The hash table stores the location of each vehicle in the B+ -tree. The LE queue stores the estimated timestamps when their member vehicles may be out of communication range with them. The layout of these data storages can be seen in figure 2.1. The CLV pointer is a dictionary containing the ID of each vehicle and its distance till the next intersection. The LE queue is a dictionary that contains all the leave timestamps, with each leave timestamp containing a list of all the distances till the next intersection. The key in the CLV tree is the distance till the next intersection which value contains a dictionary of the ID of each vehicle and its associated attributes.

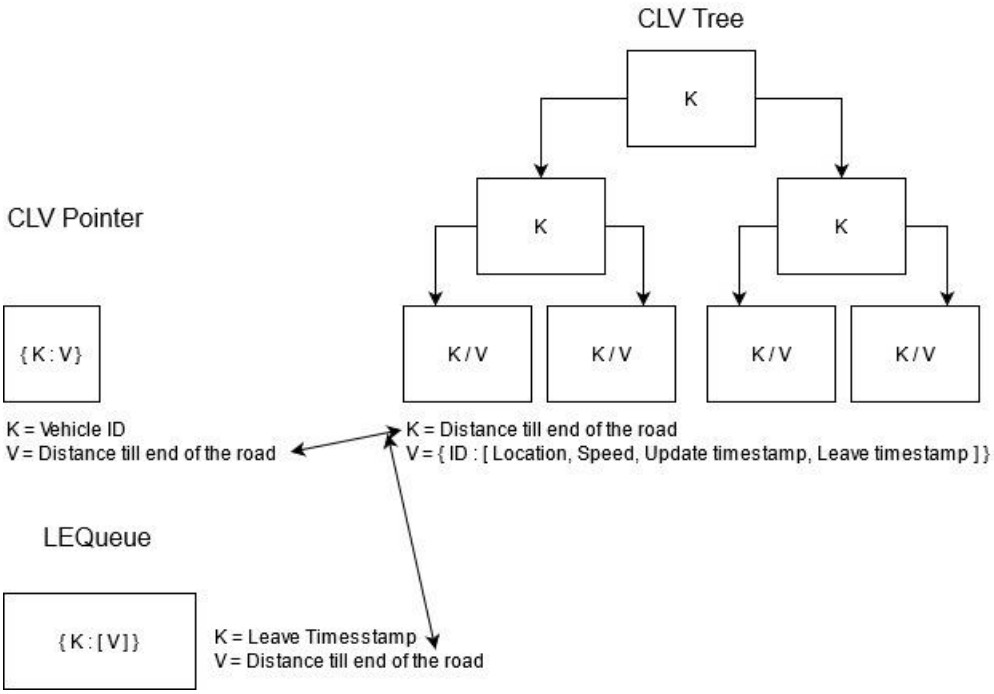


Figure 2.1: The connections between the different storage types for the MoZo architecture

Merging of zones happens when two captains are within half their communication distance of each other for a set time, this indicates that the zones have similar movement patterns therefore a single captain would be more efficient to take control. Once merging is completed and one captain has taken control, captain reassignment will take place to select the most appropriate member to stay in charge. Using the B+ -tree, the captain can take advantage of it and detect whether it is stored in the left-most or right-most part of the tree as a good captain is expected to stay relatively in the centre of the moving zone. Where this is not the case, the member which is the centre will become the new captain.

Splitting involves a member leaving a zone, this occurs when their captain checks the LE queue and notices that an expected leaving timestamp is upcoming. Seconds before that timestamp, the captain will instruct the member to leave the zone, this member state is then to roam which causes them to repeat the process of either finding a new zone or becoming a captain of their own.

2.1.3 Degree of autonomy

When autonomous vehicles coordinate, the extent to which the vehicles decide what actions to perform is the degree of autonomy in decision making. This decision making can be placed into four main classes: emergent, agreement, negotiation, and centralised.

Centralised: A individual computational entity controls the actions and decisions of all vehicles, much like how a system controls an area's traffic lights. Decisions by the entity are undebatable and no decisions are left to the vehicle. Strict control over the actions of all vehicles offers the best guarantees in terms of safety and liveness but can be

overwhelming for large-scale problems such as optimising the traffic flow of all vehicles [1].

Negotiation: An ensemble of coordinating vehicles participates in a negotiation protocol where the vehicles can propose solutions and actions, amongst a set of admissible moves dictated by the protocol. The convergence towards an equilibrium solution will be reached if properly designed where the timing and actions of each vehicle will be decided on to solve the coordination problem [1].

Agreement: Coordinating vehicles collectively define a dynamic protocol where the outcome is a set of admissible moves. Additionally, the vehicles can redetermine the goal to pursue during the coordination process [1].

Emergent: Vehicles do not explicitly engage in coordination protocol, instead act selfishly according to their goals and to maximise the utility of their actions with consideration of the information it collects about other participants to the coordination process [1]. An example of this coordination approach would be the nature-inspired algorithm particle swarm optimisation (PSO). PSO involves a population of particles representing the movements of a school of fish or flock of birds. This population roams a search space to find an optimal solution. The algorithm is emergent as particles update their velocity and position based on the utility of their actions and the action that caused the highest utility [2].

2.1.4 Coordination approaches

Platooning

Vehicles coordinate with each other to move as a single entity, in optimal scenarios, vehicles platooning would only be metres apart from each other. By keeping spaces short between vehicles an increase in roadway capacity could almost double. Also, a study showed that under heavy traffic flow the average traffic speed could increase by 10%. Other studies have shown that in a road where 20% of the vehicles are in platoons of 9, traffic speed could increase by 25% [3]. Platooning also has an economic benefit, with 3-4 metres between platooning vehicles, the average fuel consumption savings were about 11%, decreased to 8% when 8-10 metres apart [4]. If all vehicles had platooning capabilities, it could cause environmental benefits as fewer fumes from vehicles would be emitted. The SEO level of autonomy of at least 4 must be achieved for platooning to be possible in the real world, this is due to the vehicles travelling together within a metre from each other, as well as having vehicles move and stop in sync. This is unachievable with human drivers and too risky for an SAE level 3 system as it is not confident enough to complete a journey without the need for potential human intervention.

Degree of autonomy

A centralised approach would have one of the vehicles in a platoon selected as the platoon leader and would be responsible for communicating the speed profiles to other members and any other instructions when required like the MoZo architecture previously discussed. When an outside vehicle wants to join the platoon, the platoon leader decides whether the vehicle can. When a vehicle is joining/leaving, the leader is responsible for communicating the manoeuvres [1].

Although there is not much negotiation while in the platoon, this approach can be used for joining a platoon. In a no-negotiation scenario, the platoon offers a take-or-leave proposal that includes the position in which a vehicle should enter it, that vehicle can either accept or reject it. In a one-to-one negotiation scenario, a vehicle attempting to enter a platoon proposes a value for its entry into the platoon. This value takes into account the interests of the vehicle and the position in the platoon in which the vehicle wishes to enter. These interests include the vehicle's current, desired, minimum, and maximum velocity, as well as the maximum acceleration it tolerates, and a budget which is the vehicles credit for the negotiations. The platoon determines the appropriate value for the vehicle to join the platoon in the requested position [5]. Like negotiation, there are no agreements while in the platoon, also engineering an efficient agreement approach to joining is quite difficult [1].

An emergent approach has no platoon leader, vehicles engage in a platoon by adjusting their vehicles to match surrounding vehicles without any explicit agreement. Emergent approaches mimic the self-organising behaviour of many species of socialising animals that move in formation such as a flock of birds [1]. Vehicles would join into and leave from formations, when possible, which means other vehicles will constantly need to be ready to react to those events and reorganise.

Formation

When a vehicle enters the road, its first task would be to locate and attempt to join a platoon. For joining a platoon, the vehicle must have a method to evaluate candidate platoons to join and pick the most appropriate one as well as have a join manoeuvre to enter it. Otherwise, procedures must be in place for it to create a new platoon [28]. Two research areas into this subject include unilateral or bilateral formation, where a single vehicle takes action to form a platoon, or at least two vehicles are needed for formation.

Bilateral formation in literature can be seen in a study that investigated platoon formation as an optimisation problem from the perspective of cars searching to join platoons and developed a centralized and distributed approach using greedy heuristics. The optimisation problem involves a vehicle trying to find other vehicles to platoon with. It uses desired driving speed as a primary similarity metric, and position as a secondary metric to prevent vehicles from joining platoons too far away. The difference of the desired speeds of both vehicles with the distance between the vehicles is weighted separately to produce a score, with the lowest score being the best match. A vehicle will not consider platooning with another vehicle that is behind it as it wants to join from behind. In the centralized approach, the optimization problem is solved for every vehicle at the same time by a central server that has global knowledge of all vehicles. Instead of finding the lowest combined scores of possible vehicle pairs, it goes through each vehicle to find its pair that produces the lowest score and then removes those paired vehicles from the list. In the distributed approach, each vehicle executes its greedy heuristic individually, this limits vehicles to potential pairs in their communication range. Once a range of pairs is found, the vehicle will send out a request to join the vehicle that produces the lowest score with it. Overall, the study found that the distributed heuristic was worse as the centralised approach would reduce the failed platoon formation attempts and increase the available platooning options for vehicles [28].

For unilateral formation, a study proposing a distributed urban platooning protocol (DUPP) proposed that platoons have a leader. This way when a candidate node has no possible platoons to join, under certain conditions it can create a new local platoon and become its leader [30]. Another study has suggested a unilateral infrastructure-based approach could be used. Controllers would be set up at junctions of roads which would detect vehicles and have access to their kinematic information such as speed and location to determine which vehicles should be placed into a platoon [29].

discussion

Although there can be some benefit to using infrastructure such as a central server or controllers at junctions, the major flaw is that failure in that infrastructure could prevent platooning for many vehicles. With vehicles using a shared formation algorithm, this would limit the failure of one vehicle to affect just itself and not any other vehicle. Additionally, the goal of platoons is to stay together for as long as possible, the limitation to approaches [28, 29] is that puts vehicles on the same road into a platoon without taking their destinations into account. What should also be noted is there could be a difficulty for a vehicle to encounter and platoon with other vehicles whose routes to their destinations are similar, therefore a temporary platoon might be beneficial instead of waiting for the optimal one. This way vehicles can stay platooned without letting other vehicles join until they reach a junction and take different paths or one of them finds a platoon with a similar route to its destination. For DUPP, although technically a minimum of two vehicles are needed to be classified as a platoon, other vehicles around it would most likely be in the same situation and will be able to join that new leader.

Joining Manoeuvre

A joining manoeuvre is a series of processes for a candidate vehicle to join an existing platoon. For DUPP, a candidate vehicle must fulfil four join requirements:

1. There must be a platoon within the transmission range of the candidate vehicle regardless of the lane.
2. The candidate vehicle must follow the given local platoon.
3. The length of the local platoon must be smaller than a given size.
4. No obstacles (e.g., vehicles) should block the candidate vehicle from joining the local platoon.

If all criteria are satisfied, the candidate vehicle will inform the platoon leader of their decision. The platoon leader will then send the candidate information about the rear vehicle in the platoon, the candidate vehicle must switch lanes if needed, and approach the rear of the platoon within 20 metres to perform the joining manoeuvre. Simultaneously, the platoon leader exchanges information with the candidate vehicle. The leader then adds the candidate vehicle to its list of vehicles in the platoon and the candidate vehicle changes to its status to being a member. In case multiple vehicles are attempting to join the platoon at the same time, one vehicle will win in contention-based access. Afterwards, the candidate vehicles that failed to win can attempt to join again. In a multi-lane scenario, it may be unpractical for a candidate vehicle in an adjacent road to join the end of a platoon as it would require the vehicle to slow down and change lanes. Instead, a related study involving multi-dimensional platoons created a method of reshaping a multi-lane platoon

into a single-lane platoon, which has similar aspects to this problem. Its solution was to have the platoon create a gap next to the candidate vehicle for it to slip into [33].

Merging Manoeuvre

For DUPP, a merging manoeuvre is where two platoons combine and act as one. While travelling, a local platoon will slowly increase its members but can get split by traffic lights and communication failure, which in turn creates multiple smaller platoons which reduces efficiency overall. A merging manoeuvre would counter this loss of efficiency by combining two small platoons into a large one.

For a merge to proceed, four requirements must be fulfilled:

1. Two local platoons are within the same lane.
2. The leader of each platoon must be within transmission range of each other.
3. The sum of the lengths of both platoons must be smaller than a given size.
4. No obstacles (e.g., vehicles) should block the platoons from merging.

Once all criteria are satisfied, the platoon which is further down the road (following platoon) will adjust its speed to approach within 20 metres of the rear vehicle of the preceding platoon. The leader of the preceding platoon will then permit the following platoon to join it, which follows with the exchanging of information about the vehicles in each platoon. Once that is completed, the following platoon will join the proceeding platoon. With the preceding platoon's leader becoming the merged platoon leader.

As DUPP requires platoons to be within the same lane before merging, it is possible that an obstacle could prevent a platoon in an adjacent lane from firstly switching lanes, therefore a different approach is needed. One approach allows platoons that are side by side to each other on adjacent lanes to merge into one. Each platoon has different tasks, for simplicity, Platoon B merges in Platoon A. Platoon A has its vehicles make gaps in between themselves which are large enough for Platoon B's vehicles. Next, vehicles in Platoon B spread out and position themselves adjacent to an empty gap. Finally, the vehicles change lanes into the empty spaces and Platoon A and B merge into a new platoon [31].

Leaving Manoeuvre

A leaving manoeuvre is where a vehicle exits from a platoon. With DUPP, there are two cases for this, either expectedly such as when a vehicle plans to take a different road from the platoon to reach its destination, or unexpectedly such as losing connection to the platoon leader. The leaving manoeuvres are different depending on if the vehicle is a leader or member.

In the expected case, before a vehicle leaves it must inform the platoon of its plan to separate from them. The platoon leader then removes the leaving vehicles from its list of member vehicles and approves the leaving vehicle to perform the leaving manoeuvre. The leaving vehicle then precedes to change lanes. In the case it creates a large gap within the platoon, the platoon vehicle will instruct the member vehicles behind the leaving vehicles to accelerate to close that gap. If the vehicle is a leader, the member vehicle behind it inherits the leader role. This new leader constructs its list of members and

removes the previous leader. The previous leader then repeats the actions that a member vehicle would take to perform a leaving manoeuvre.

In the unexpected case, a communication failure in a vehicle would prevent them from platooning safely, therefore when this failure is detected by the platoon leader, that vehicle is removed as a member from the platoon. Additionally, any member vehicle behind that removed vehicle should also be removed as the vehicle with communication failure is now an obstacle. The removed vehicles would then have to perform creation, joining, or merging manoeuvres. The manoeuvre to perform depends on the number of removed members. If the number is less than or equal to three, then the vehicles are removed and then independently perform joining manoeuvres. If the number is greater than three, then the platoon and removed members should instead perform a splitting manoeuvre.

Splitting Manoeuvre
A platoon may be split when some of its vehicles are unable to cross the intersection point because of a traffic signal at an intersection. In DUPP, when a platoon is split into two, the vehicle which is at the front of the split platoon becomes the new leader. The new platoon leader creates a list, and the member vehicles send their information to it. The vehicles are also removed from the previous platoon's list.

Platoon size
When considering platoon size, two characteristics of traffic need to be considered: capacity and stability. Results from a study showed that a large platoon reaps the benefits of platooning as the capacity of roads increases but also showed that the stability of the platoon decreases with an increase of the platoon size. A large platoon would decrease its lateral manoeuvrability, making it difficult for vehicles to join or leave. Based on the trade-off between the two characteristics it was suggested that when the road is full of vehicles that can all platoon, a platoon size should have a maximum of 10 vehicles. This is because the increase of road capacity with platoon size starts to flat line from that point. In the real world, this number could be even smaller depending on the penetration rate of vehicles which can platoon and the platooning intensity. If both the penetration rate and intensity are low then a size limit is pointless as the maximum platoon size may not be reached, but if these values are high then a limited size is needed to improve stability [32].

2.1.5 Traffic flow optimization

Traffic flow optimisation involves directing traffic across a road network to balance the overall exploitation of the road infrastructure, minimising travel times, and avoiding traffic jams. A centralised approach would include a local control station that can receive all the information about vehicles and roads. This station would be responsible for analysing that information to send each driver information about their driving performance and recommend actions to take to avoid situations such as traffic jams [7]. The actions for vehicles to take would be the product of the station applying traffic flow/density forecasting techniques to anticipate congestions and jams, and to elaborating plans avoiding them. In an all-autonomous vehicle environment, the use of physical signals such as traffic lights and road signs as guidance would be replaced by the station. In a large-scale network, having a station guide multiple vehicles seems impractical, therefore the system can be

applied to a fraction of the vehicles by ordering them to change routes when a potential traffic jam is detected [1].

2.1.6 Adjusting communication for platooning

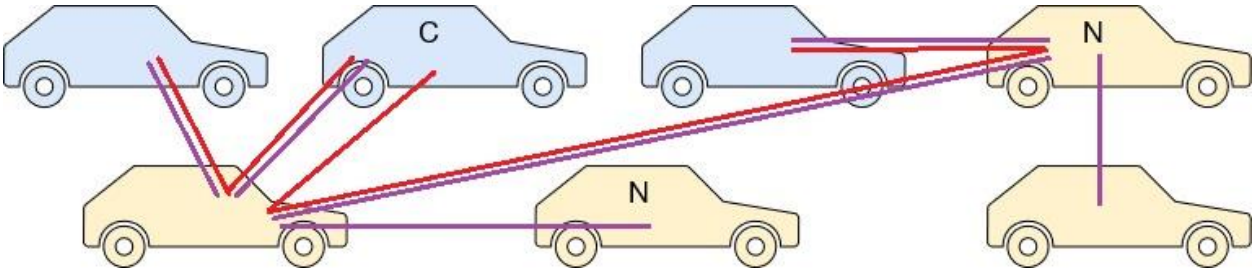


Figure 2.2: Platooning and communication architectures working separately

By having communication and coordination act as two distinct systems, network issues could cause disruptions to a platoon. Take the following scenario shown in figure 2.2, the blue vehicles are a platoon while the yellow vehicles are independent. The platoon leader is the vehicle with the letter C on it and a vehicle with the letter N on it is a zone captain in a VANET using the MOZO architecture. A purple line indicates two-way communication between vehicles. For the platoon leader to communicate with its members, a message is required to pass through one to two zone captains (as indicated with the red lines) which can cause the following issues. Firstly, there would be an unnecessary load on the network as messages could just be sent directly from a platoon leader to its members instead of through zone captains. Secondly, messages could fail to be delivered as with each zone captain it passes through, the chances of the message failing to redirect would increase. The failure in messages being sent within the platoon would in a worst-case scenario cause it to disband. When having a platoon leader in addition act as a zone captain (see figure 2.3), messages can be directly sent to each member which solves the previous problems.

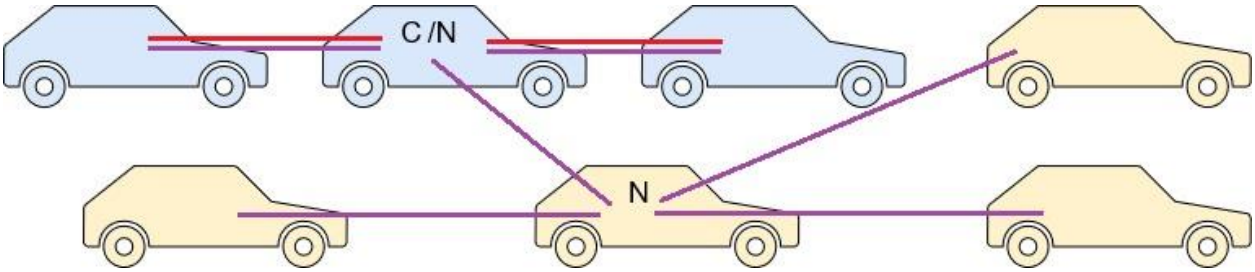


Figure 2.3: Platooning and communication architectures combined

2.1.7 Coordination techniques to focus on

There are a variety of key coordination problems such as crossing intersections, parking, ride-sharing, ramp merging, platooning, and traffic flow optimisation [1]. The large scale of coordination problems requires that only a select number be picked as with the time frame of the dissertation, the deliverable would either have simple approaches to them or

most of them would have sections uncompleted. For this reason, the project will focus mainly on platooning and potentially integrating some traffic flow optimisation.

2.2 Simulation and tools

2.2.1 Realistic Simulations

Due to the complicated nature of developing the project in a real-world environment using real vehicles, the most reasonable option would be to simulate coordination but this introduces many challenges. Most importantly, the simulator chosen must be able to replicate the real-world behaviour of the vehicles and environment to a useful degree, or the project could be useless in the real world. Secondly, to run a large-scale simulation of platooning which could potentially involve a hundred vehicles, the computational power needed for that simulator must not exceed the capabilities of the hardware available for the project. Lastly, the simulator must have extensive documentation to allow the project to be created and prevent bottlenecks from happening in the future [19]. Some of the new simulation software available is CARLA and PGdrive.

CARLA is an open-source similar for autonomous driving research [9]. CARLA allows for control of vehicles in three-dimensional environments as well as allowing for the use of different types of sensors and vehicle models that mimic their real-life counterparts. Besides realism, the simulator lacks a large community. This limits the support you can receive for issues (seen in the official Carla forum [17]) and reduces the environments available to test the project. In the scenario where an environment with specific details is required, a map can be created using the software RoadRunner [18]. CARLA additionally has a path system linked to all the roads in the simulator which would increase the ease of having vehicles follow roads to their destination.

PG drive is an open-ended and highly configurable driving simulator, it allows eight different roadblocks ranging from straight roads to intersections to be selected and assembled into an interactive environment to drive in which allows for autonomous vehicles to be tested on in multiple scenarios. It can also achieve up to 500 simulation steps per second when running a single instance on PC and can be easily paralleled to further boost efficiency compared [20]. The downfall is that due to the software being relatively new, the documentation is not in-depth and users may be presented with new unseen errors or challenges that have not been discussed online previously which has the potential to create bottlenecks in the development stage.

2.2.2 Traffic flow simulation models

Modelling of traffic flow can be done at macroscopic, mesoscopic, and microscopic levels. Macroscopic stream models show the relationship among traffic flow parameters, with the major ones including speed, density, and flow. It originated under the assumption that traffic flow can be compared to fluid streams. Microscopic models are the most in-depth level as it considers the characteristics of individual vehicles. These models can be classed into two main types, the general car-following model and the safe-distance model. The safe-distance model assumes the vehicle will keep a safe distance between itself and

the following vehicle to avoid collisions, while the general car following model assumes the characteristics of a vehicle is directly related to the influence of the leading driver. Mesoscopic models are hybrids between microscopic and macroscopic levels [24]. By looking at the Scalable Microscopic Adaptive Road Traffic Simulator (SMARTS), we can easily see the benefits of these models as SMARTS could simulate 300,000 vehicles in a 600km^2 area 2.72 times faster than real-time when the vehicles are updated 5 times per second [25]. Compared to realistic simulators such as CARLA, this allows for a greater understanding of how coordination techniques such as platooning can have an effect on a city-wide scale.

2.2.3 Selecting a simulator

After weighing the pros and cons of the researched simulators, a realistic simulator seemed to be the best option. Although the traffic flow simulators can spawn thousands of vehicles, the goal of the project is to test how a platooning architecture would affect congestion in a realistic environment. The chosen simulator was Carla, although PGdrive had the unique advantage of unlimited layouts of road networks, CARLA offered a greater understanding of its architecture as it had the largest community support as well as very in-depth documentation.

2.2.4 Measure metric for traffic congestion

To understand the nature of congestion and how coordination methods impact it, a system for measuring the severity of congestion is needed. The measured quantity can be based on the flow of the traveller's experience (e.g., travel time) or of the flow characteristics (e.g., traffic density) within areawide conditions or a certain location. Traffic congestion can be measured by a set of discrete classes or a continuous value [21].

There are two principles of traffic congestion, micro-level factors and macro-level factors. Micro-level factors relate to traffic on the road such as the number of vehicles travelling on it. Macro-level factors relate to the overall demand of the road such as vehicle ownership trends. There is no universal practice used to measure congestion, so it changes around the world. A common measurement used to declare congestion is when vehicles fall below a certain speed for a duration of time as used in California, Japan, and South Korea. For example, the California Department of Transportation defines congestion as occurring on a freeway when the average speed drops below 35mph for 15 minutes or more on a typical weekday. There are several speed measurements, one of the popular ones is the speed reduction index. It represents the ratio of the decline in speeds from free-flow conditions. A continuous scale is used to differentiate between different levels of congestion. This index can be applied to entire routes, urban areas, or segments of a freeway for off-peak and peak conditions. As a reduction of speed directly increases time spent on the road, it is appropriate to add a threshold to the metric for evaluating congestion [22].

Although speed is a popular choice, other metrics can be used to measure congestion such as travel time, delay, and level of service (LOS) and volume. With using time, the Travel Time Index can be used to compare real period travel and free-flow travel while

accounting for both recurring and incident conditions. The index allows for traffic congestion to be expressed in terms of both space and time. Delay is the additional time experienced when compared to acceptable or free-flow travel conditions. For delay estimation, a threshold value is used for the beginning of the delay such as some researchers using a threshold of congestion to begin at a volume to capacity (v/c) ratio of 0.77. The volume to capacity ratio can provide a measure of how occupied a roadway is which allows conclusions about the congested conditions. An example would be that a roadway that is 90% occupied would most likely be very congested [22].

When comparing the discussed congestion metrics, volume-to-capacity seemed like the most appropriate. One of the main points of the problem was the difficulty and cost of increasing the capacity of the roads by building new infrastructure, as platooning would potentially decrease the volume-to-capacity of the roads, this metric would suit the dissertation more than the others.

2.2.5 Collecting and analysing data

In the real world, data collection techniques would include using different types of sensors such as images or magnetic-based ones, but as the vehicles will be simulated, the exact speed, direction, and location of vehicles can be retrieved. As the purpose of this dissertation is to determine the effect platooning has on congestion, two types of scenarios will be run for each test: platooning enabled and vehicles acting independently. A test includes four aspects: run time, repetitions, environment, and population. Each test will last 2 minutes and be repeated 10 times to allow for reliable results to be collected. Tests will take place in multiple different environments with road networks varying in complexity and varying populations of vehicles to get a complete overview of how the platooning architecture would perform in different situations.

For analysing the data, the results must be represented in a way that shows its global and local effects such as how it impacts the entire map and individual roads. For local analysation, a heatmap of the road network will be used to visualise how the tests affected different parts of the road network on a map. An example of this can be seen in figure 2.4, a colour for each road will represent the average congestion experienced. The colour will be linked to a colour gradient where green indicates the lowest and red indicates the highest congestion experienced from both scenarios. For global analysation, graphs can be used to compare the average capacity of the roads in both types of scenarios over time.

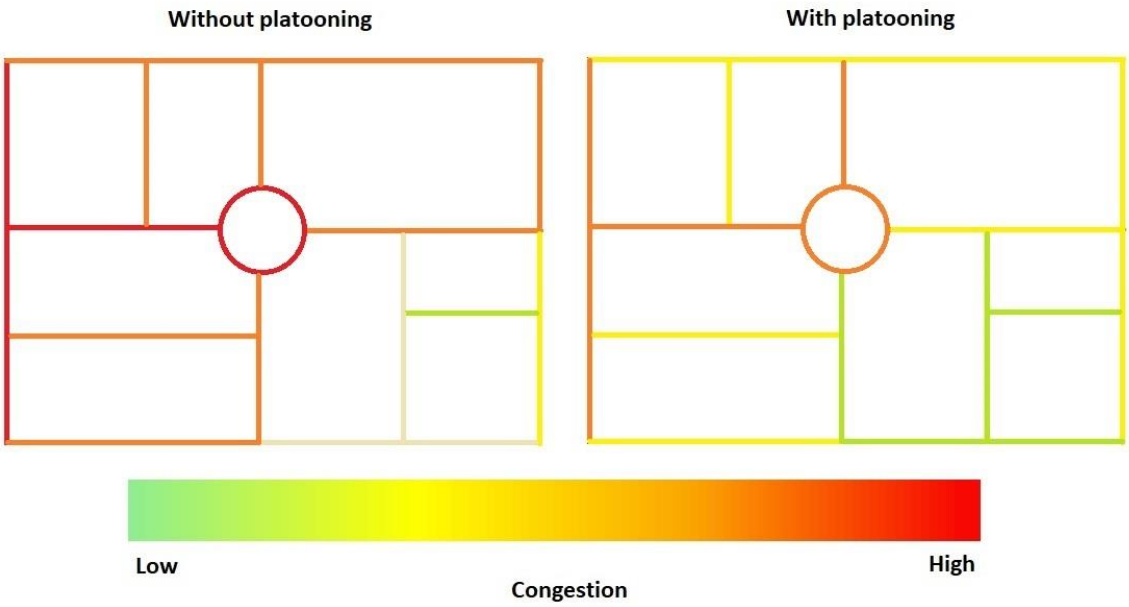


Figure 2.4: An example of the potential results for local analysation visualised.

3. Autonomous platooning development

3.1 Introduction

For the degree of autonomy, the architecture is a hybrid between a centralised and emergent approach, although the captain of a platoon has control of its members by being able to control their speed, spacing, and ability to change lanes, the member vehicles are emergent in the sense that they follow their chosen route and are just constrained by the platoon captain's decisions. The basic manoeuvres such as merging, splitting, joining, etc, are influenced by the literature of the DUPP architecture [30]. This allowed for a base design to be created and expanded upon such as implementing multi-lane merging inspired by Platoon Merging Approach Based on Hybrid Trajectory Planning and CACC Strategies paper [31]. Due to the research by Zhou and Zhu [32] on the effect of maximum platoon size, the number of members including the captain is set to 10.

3.2 Requirements

As the purpose of the architecture is to investigate the effect of platooning on congestion, the end-product has no customers so no non-functional requirements such as an easy-to-use interface are required. To prioritise the functional requirements and place a level of importance on each one, the MoSCoW method is used to classify these requirements into four different groups: must have, should have, could have, won't have. The requirements are classified as:

1. **Must have**: means that the requirements are critical for the project and need to be completed.
2. **Should have**: means that although the requirements are not vital to the project, they are expected to be implemented.
3. **Could have**: means that the requirements are unexpected in the final product but would benefit it.
4. **Won't have**: means that the requirements will not be in the final product.

In table 3.1, the functional requirements of the project are displayed. The vehicle and platooning architecture are the only two components crucial to the project as the vehicle architecture are needed to create a foundation for the platooning architecture to be built upon and this dissertation studies the effect of platooning on congestion. Although the communication architecture is not crucial, it should be implemented into the system to increase the realism of the simulation. The traffic flow optimisation may not be possible to complete but will be beneficial to understanding how different platoons can coordinate with each other to reduce congestion. A consideration to the project was to test whether the platooning architecture would work when a per cent of the vehicles are equipped with it, but due to the current size of the project, all vehicles will have the ability to platoon.

Functional requirements	Group	MoSCoW
A Vehicle architecture to control the vehicle's movements	Vehicle	Must
Vehicles must be able to join a platoon	Platoon	Must

Platoons must be allowed to merge	Platoon	Must
A platoon must be able to split into two	Platoon	Must
A vehicle must be able to leave a platoon	Platoon	Must
A vehicle must be able to form a platoon	Platoon	Must
Multi-lane merging	Platoon	Should
Simulated communication technology	Communication	Should
Cluster-based VANET	Communication	Should
Route management	Traffic flow optimisation	Could
A mixture of autonomous vehicles with and without platooning	Platoon	Won't

Table 3.1: Functional requirements for the project

3.3 Design and implementation details

3.3.1 Vehicle Movement Model

In the Carla simulator, vehicles move around road networks by following waypoints. Waypoints are nodes on a road that provide information such as the direction of the lane, the road and lane id, the previous waypoint, and the next possible waypoints. When given a destination on the map, the Carla GlobalRoutePlanner class creates an ordered list of waypoints to follow from their location to their destination. With this route, the vehicle's thrust and steering will be updated once per loop of the simulation to drive in the direction of each waypoint. When a vehicle encounters another vehicle ahead of it that is too close or a red traffic light, it will break. For road networks with multiple lanes, the navigation system did not consider collisions between vehicles when one is changing lanes. This prevented multilane platooning; therefore, a fix for that problem was created.

```
Vehicle V:
   1. If V wants to change lanes
   2.       For each V' in the world
   3.             If there is a small distance between V and V'
   4.                   If V and V' are both on the road and adjacent
                        lanes
   5.                         If the target lane of V is the lane V'
                              is on
   6.                               Have V continue driving on its
                                    current lane
   7.                               End Loop
```

Code 3.1: Collision prevention for vehicles changing lanes.

For collision detection, a list of conditions must be satisfied before the vehicle is prevented from changing lanes. If prevented, the vehicle will continue to drive forwards

until it is possible to switch lanes. As seen in the pseudo-code in code 3.1, the conditions include:

1. The vehicle must want to change lanes.
2. There must be a small distance between it and another vehicle.
3. That other vehicle and the current one must be on the same road in adjacent lanes to each other.
4. The vehicle must be planning to move onto the adjacent lane.

This can be demonstrated in figure 3.1. In scenario 1, the distance (d) between the closest two green vehicles to the blue vehicle is too small, therefore this violates one of the conditions so the blue vehicle cannot change lanes. In scenario 2, the distance between the blue vehicle and the closest green vehicle is large enough, therefore all conditions are met so the blue vehicle can change lanes.

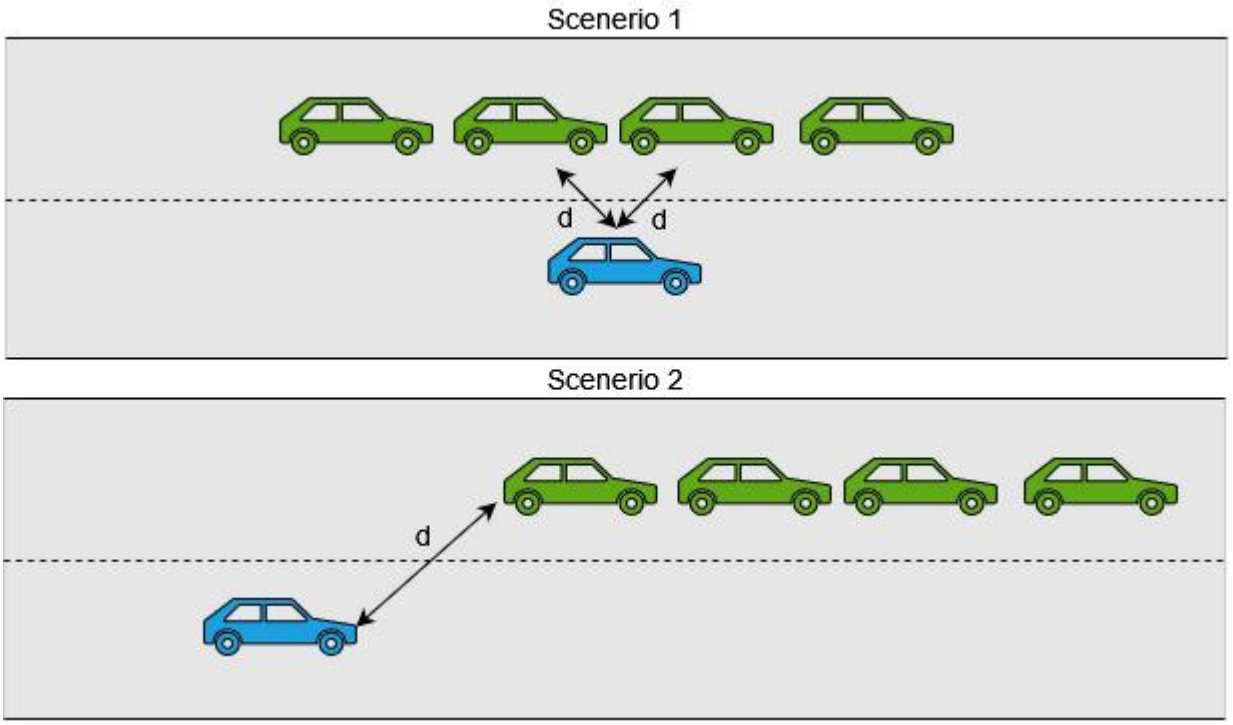


Figure 3.1: Two scenarios that use the collision prevention method.

3.3.2 Vehicle Architecture

Four parameters can be altered which affect the performance of the simulation: LPS (loops per second), check intervals, spawn amount, and default speed. LPS is the number of times the main loop of the simulation runs each second, check intervals is how often each platoon captain checks for events to happen such as a vehicle leaving or a potential merge, spawn amount is the number of vehicles to be created, and default speed is the desired speed to drive at.

Once the vehicles have been spawned in and their ability to platoon is implemented in each one, the main loop of the simulation begins. This loop goes through each vehicle

one at a time repeatedly until the simulation is stopped by the user. The first task of a vehicle is to update its attributes which include and is limited to their velocity, acceleration, location, and rotation. Next, the vehicles will perform any manoeuvres they are required to do such as joining or merging. Once that is completed, if a vehicle is a platoon leader (also known as a captain), the leader will update its members speed to maintain even spacing within the platoon, and then check if the platoon needs to split. Afterwards, if the platoon leader is not performing any actions, such as merging or waiting for a vehicle to join the platoon, it will check to see if it can merge with another platoon. If the vehicle is still the leader of the platoon whether a merge happened or not, the final task of a leader is to check for members that are potentially going to leave the platoon. When the platoon leader tasks are completed, if the vehicle has a set acceleration, it will update its next speed to achieve. Lastly, a vehicle will perform its roaming method to correct its distance and speed to the next waypoint in its route. The data flow diagram in figure 3.2 lays out this process.

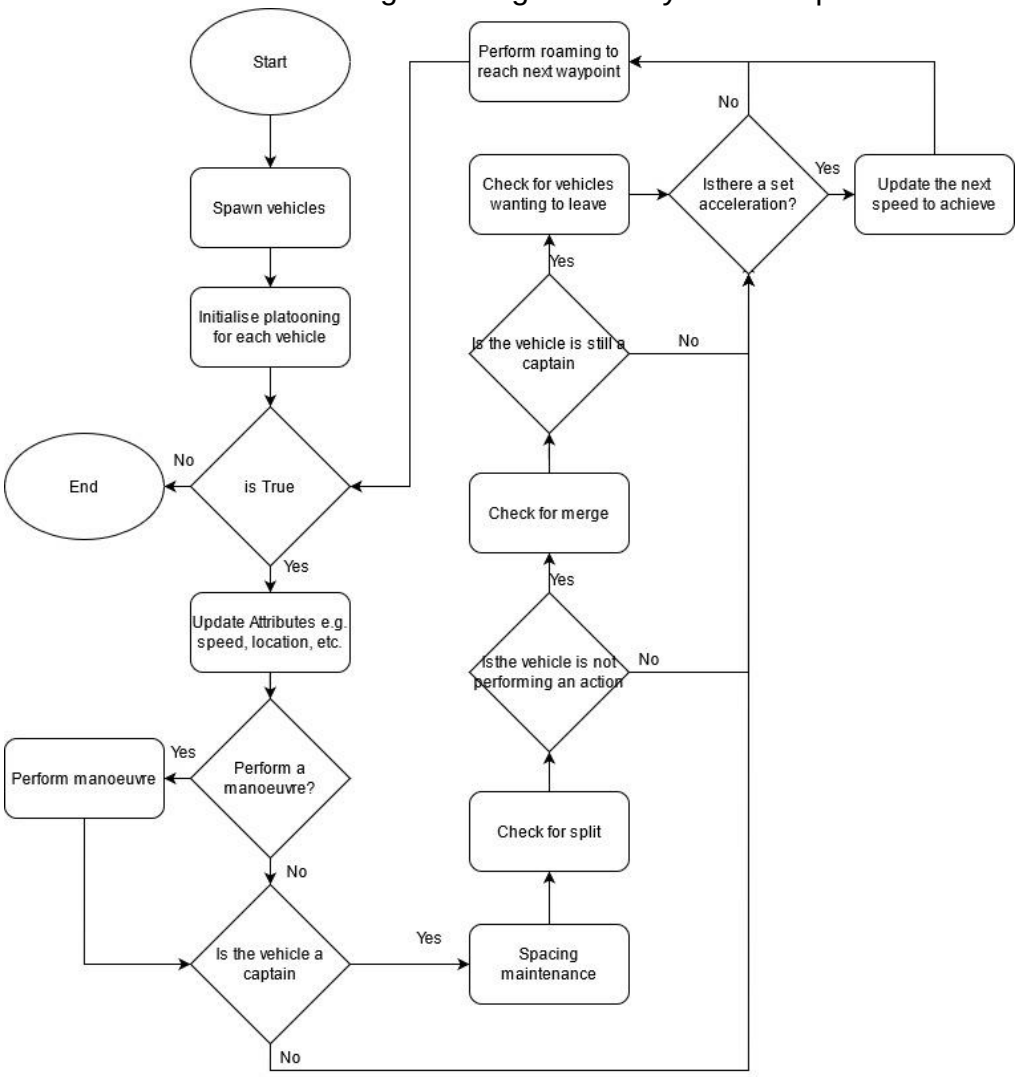


Figure 3.2: Vehicle architecture data flow diagram

3.3.3 Search for platoons

```
Vehicle Vr:
   1. For each Vc is not busy and has platoon spaces available
   2.       If Vr is behind the rear vehicle in the platoon of Vc on
            the same road and lane
   3.             If there are no obstacles between Vr and the rear of
                  Vc
   4.                   Record the distance between Vr and Vc to a
                        list of platoon options
   5. If no Vc found
   6.       Start Formation Manoeuvre and become a captain
   7. Else
   8.       Start joining Manoeuvre
```

Code 3.2: The process of either finding a platoon to join or forming a new platoon.

When the platooning object is initialised for a vehicle (V_r), the “search for platoons” method is activated. This determines if a vehicle either joins an existing platoon or forms its own. The first step is to find all the platoon captains (V_c) which would allow the vehicle to join their platoon. The conditions require the vehicle to be on the same road and lane as the rear vehicle of the platoon, and with there being no obstacles (vehicles) between them. If the conditions are fulfilled, the V_s will record the distance between itself and V_c. Once that is complete, the V_r will check if any potential platoons are available to join, in the case where none are found, the V_r will form its platoon, else it will move on to the joining manoeuvre. This process can be seen in code 3.2.

In figure 3.3, a scenario of how this process works can be seen. The blue and red vehicles are searching for a platoon. Both vehicles are at an acceptable distance from the green platoon and have no obstacles between themselves to the platoon. The difference is that the blue vehicle can join the green platoon as they are on the same lane, but the red vehicle is on a different lane, therefore it cannot join the platoon and must form its own.

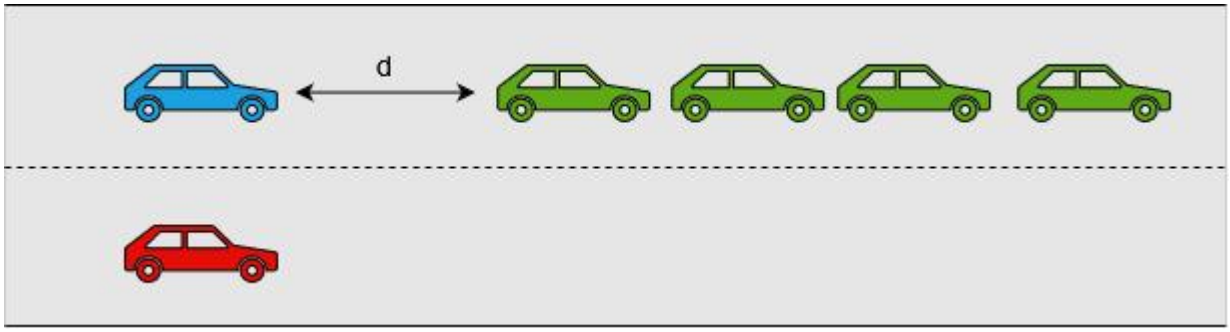


Figure 3.3: Scenario of two vehicles searching for a platoon

3.3.4 Joining Manoeuvre

```
Vehicle Vr:
   1. For each Vc in the platoon options list:
   2.       Find the Vp (Vc with the shortest distance)
   3. While True
   4.       If the rear of the platoon and Vr on the same road
   5.             If the distance between Vr and Vp is small enough
   6.                   Vr become a member of Vp platoon
   7.                   End loop
   8.             Else
   9.                   Drive faster than the platoon to catch up to
                        them
   10.            Else
   11.                  Cancel the join attempt
   12.                  Repeat the search for platoons
```

Code 3.3: The process of joining a platoon.

When a V_s activates the joining manoeuvre, it starts by picking the V_c it has the shortest distance to (referred to by V_p). The V_s will attempt to join the platoon of V_p. If V_s is not on the same road as the rear vehicle of the platoon, this means that joining the platoon is no longer possible, therefore V_r will cancel the join attempt and repeat the search for a new platoon or form its own. In the scenario where V_r and the platoon are still on the same road, the V_r will calculate the distance between itself and the rear vehicle of the platoon. If this distance is small enough, then the V_s will become a member of the platoon of V_p, otherwise, V_r will drive at a faster speed than the platoon to catch up. This process can be seen in code 3.3.

In figure 3.4, a scenario of how this process works can be seen. Once the blue vehicle has decided to join the green platoon, it drives at a speed faster than the platoon to catch up with it. Once the vehicle is within a set distance to the rear of the platoon, it will finally become a member of it and the platoon captain will take control of adjusting the speed of the blue vehicle from there on.

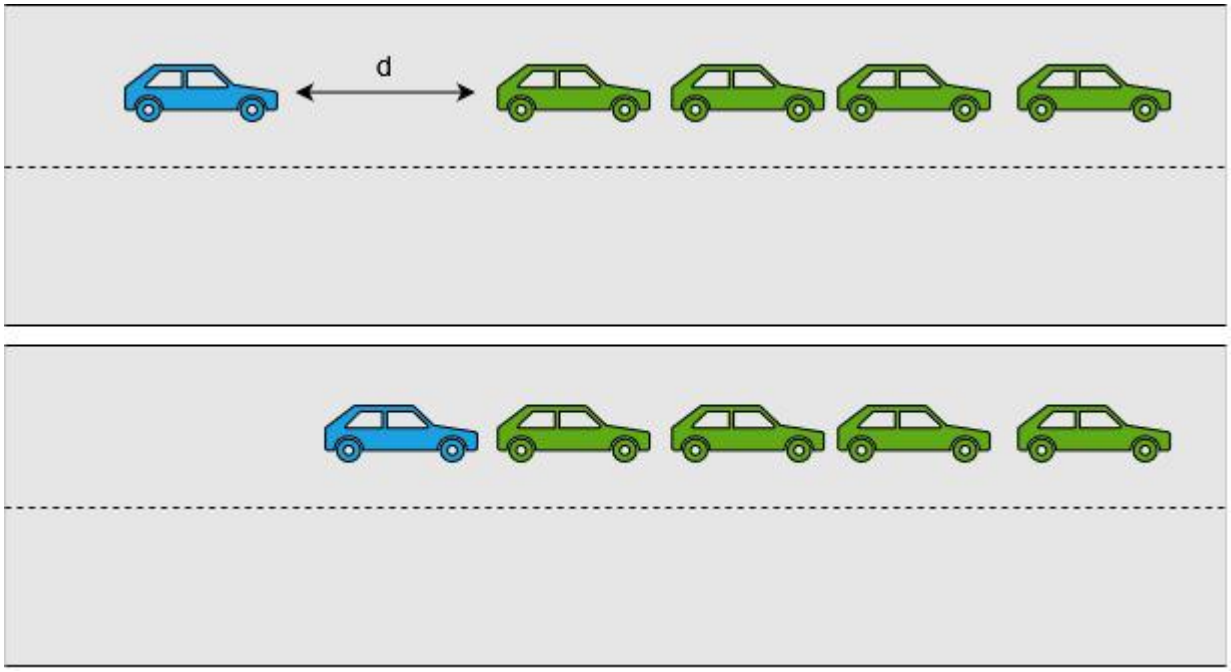


Figure 3.4: Scenario of a vehicle joining a platoon.

3.3.5 Single-lane Merging Manoeuvre

```
Vehicle Vc:
   1. For every other Vc' that is not busy and has platoon member
      spaces available
   2.       If Vc is behind Vc' on the same road and lane
   3.             If there are no obstacles between Vs and the rear of
                  Vc
   4.                   Attempt to Join Vc'
   5. While an attempt to join Vc' is active
   6.       If the rear of the platoon of Vc' and Vc are on the same
            road
   7.             If the distance between Vc and the rear vehicle of
                  the platoon of Vc' is small enough
   8.                   The platoon of Vc merges with the platoon of
                        Vc'
   9.                   End loop
   10.            Else
   11.                        The platoon of Vc drives faster than
                              the platoon of Vc' to catch up
                              to them
   12.      Else
   13.            Cancel the merge attempt
   14.            End Loop
```

Code 3.4: The process of a single-lane merge of two platoons

At a set interval, a V_c will check every other captain (V_c') to see if it can merge with them. Similar to the joining manoeuvre, the conditions for merging both platoons requires that V_c is behind the rear vehicle of the platoon of V_c', on the same road and lane, there being no obstacles between them, enough platoon spaces are available, and V_c' is not busy with other manoeuvres. Once these conditions are met, the platoon of V_c will drive faster than the platoon of V_c' until it is within a small distance of the

platoon of V_c' and then merge. V_c' will then be in control of reducing the gaps between the vehicles even further. Again, in the scenario where both platoons move onto different roads before a merge can be completed, the merge attempt will be cancelled. This process can be seen in code 3.4.

In figure 3.5, a scenario of how this process works can be seen. The captain (represented by a red vehicle on the left) has found a platoon on the same road and lane, there are no obstacles between the caption and the rear of the platoon found, therefore the platoon on the left will increase its speed to catch up and then finally merge with the platoon on the right. The vehicle in front of the platoon becomes the new captain.

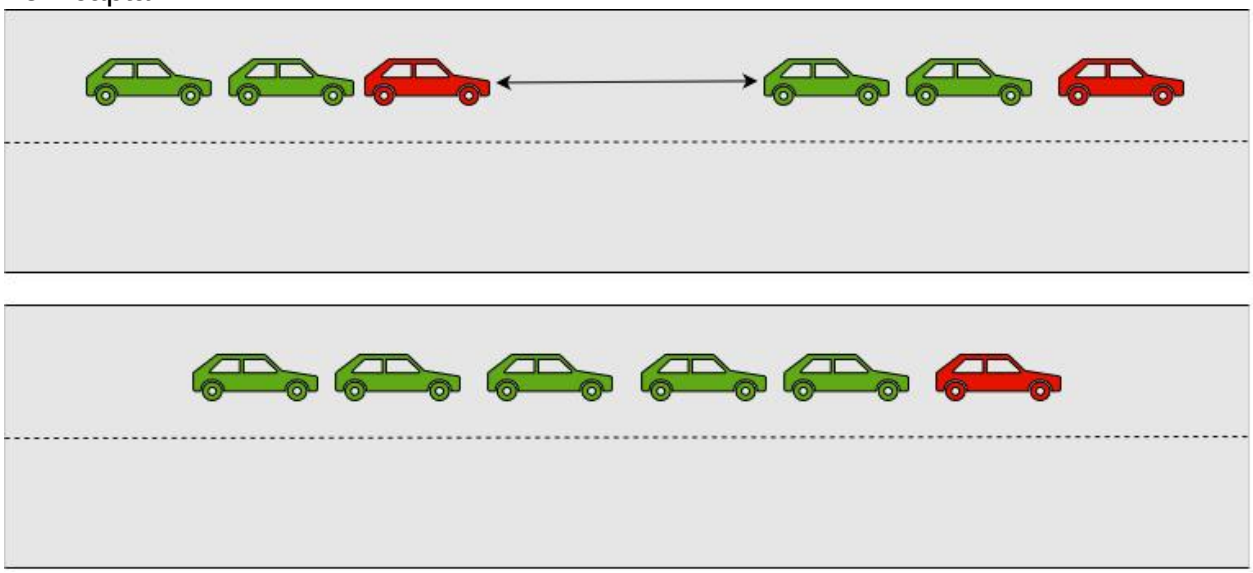


Figure 3.5: Scenario of two platoons merging.

3.3.6 Multi-lane Merging Manoeuvre

```
Vehicle Vc:
   1. For every other Vc' that is not busy and has platoon member
      spaces available
   2.       If Vc is behind Vc' on the same road and adjacent lane
   3.             If Vc wants to switch to the same lane as Vc'
   4.                   If any Vm of Vc is opposite a Vm' of Vc'
   5.                         Attempt to merge with the platoons of Vc
                              and Vc'
   6.
   7. While an attempt to merge with Vc' is active
   8.       If all vehicles aren't on the same lane:
   9.             If the merge stage is 1
   10.                        1.1.  Get the positions (behind, in
                              between, and in front of the adjacent
                              platoon) of the Vm in relation to the
                              platoon of V c'.
   11.                        1.2.  Get the Vm' which need to create
                              space for Vm which have the middle
                              position
   12.                        1.3.  Tell the Vm with middle positions
                              and the Vm' found in step 1.2. to create
                              equally large gaps.
   13.                        1.4.  Tell the Vm with the positions
                              behind and in front to also create gaps
                              half the size as in 1.3.
   14.                        1.5.  If the first Vm in the middle
                              position is not Vc, add Vm-1 to list of
                              Vm with the middle position.
   15.                        1.6.  If the first Vm' found in step
                              1.2. is not Vc', add Vm-1' to list of Vm'
                              found in step 1.2. and move on to merge
                              stage 2.
   16.                  Else if merge stage is 2
   17.                        2.1.  Check whether all Vm with the
                              middle position and the Vm' found in
                              step 1.2. have spaced out.
   18.                        2.2.  If 2.1. produces a true result,
                              change whether Vm go in front or behind
                              there opposing Vm', depending on whether
                              Vc is further up the road than Vc', then
                              move on to merge stage 3.
   19.                  Else if merge stage is 3
   20.                        3.1.  Adjust the speed of Vm so it can
                              be centre of the gap it is supposed to
                              drive into, by using the distance
                              between the two Vm' that surround the
                              gap.
   21.                        3.2   When all Vm are close enough to
                              the centre, allow them to drive into
                              the gaps.
   22.            Else
   23.                  Merge all vehicles into one platoon
```

Code 3.5: The process of multi-lane merging for 2 platoons

At a set interval, a V_c will check every V_c' to see if it can merge with them. The conditions for merging both platoons requires that V_c is on an adjacent lane of V_c' and the same road, V_c wants to switch to the same lane as V_c', enough platoon spaces are

available, V_c' is not busy with other manoeuvres, and any V_m is opposite a V_m'. Once those requirements are met, the merging attempt begins. The merge process works by having certain V_m' create gaps between themselves to allow for V_m to drive into. Any other V_m which do not get assigned a gap to drive into either change lanes to the back or front of the platoon of V_c'. The process can be seen in code 3.5 and figure 3.6, and has four stages:

1. Identifying the roles of all vehicles to perform
2. Having the chosen vehicles to space out and create gaps
3. Having V_m drive into the gaps
4. Have the vehicles merge into one platoon

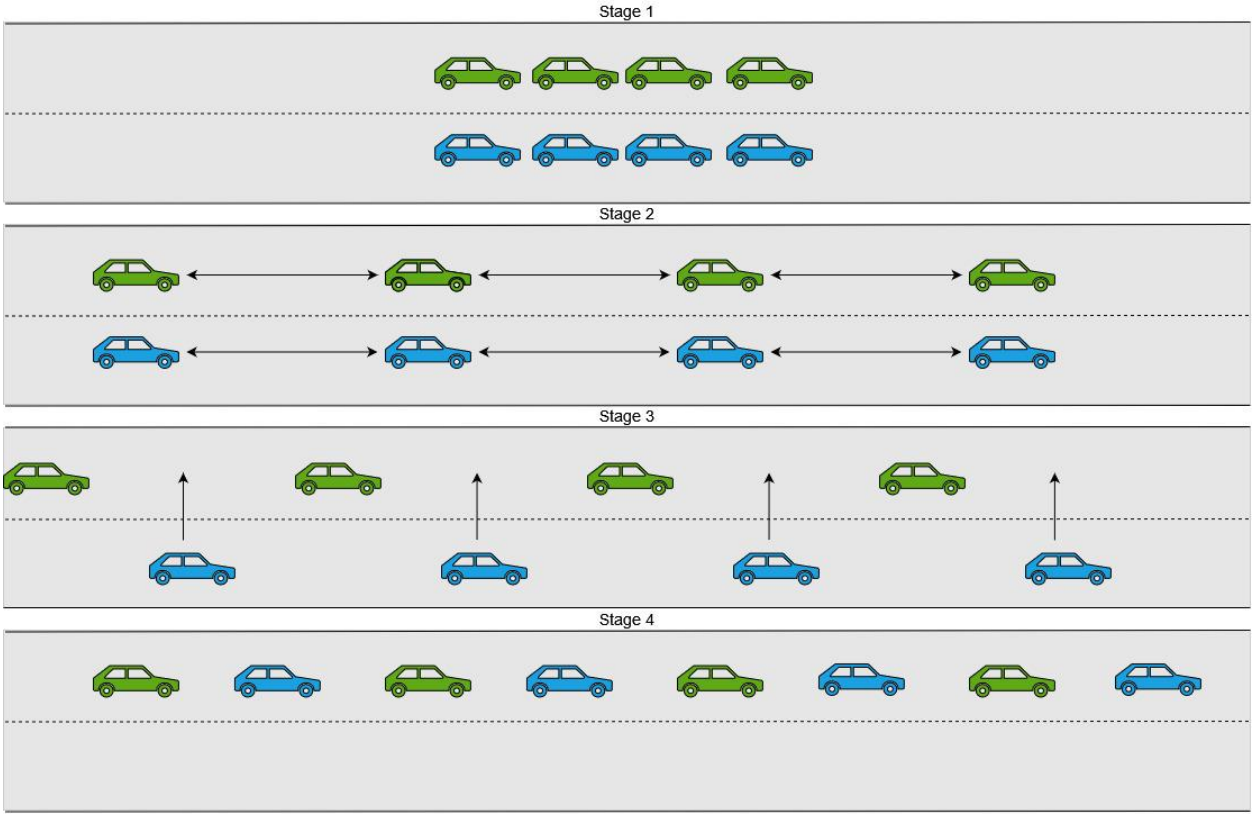


Figure 3.6: The stages of multi-lane merging.

Stage 1:

The roles given to a V_m are behind, middle, or in-front. These represent the positions of V_m in relation to the platoon of V_c'. The waypoints (A and B) at the front and end of the platoon of V_c' are selected, next, the closet waypoint (C and D) to A and B on the road containing V_c are recorded. A V_m behind waypoint C is labelled as behind the platoon of V_c', between C and D are labelled as middle, and in front of D is labelled as in-front.

In figure 3.7, a scenario of how this process works can be seen. The purple line on the left represents a segment through waypoints A and C, this is repeated for waypoints B and D on the right. Vehicles in the blue platoon between the purple lines are labelled

as middle, past the segment of B and D are labelled as in-front, and the remainders are labelled as behind.

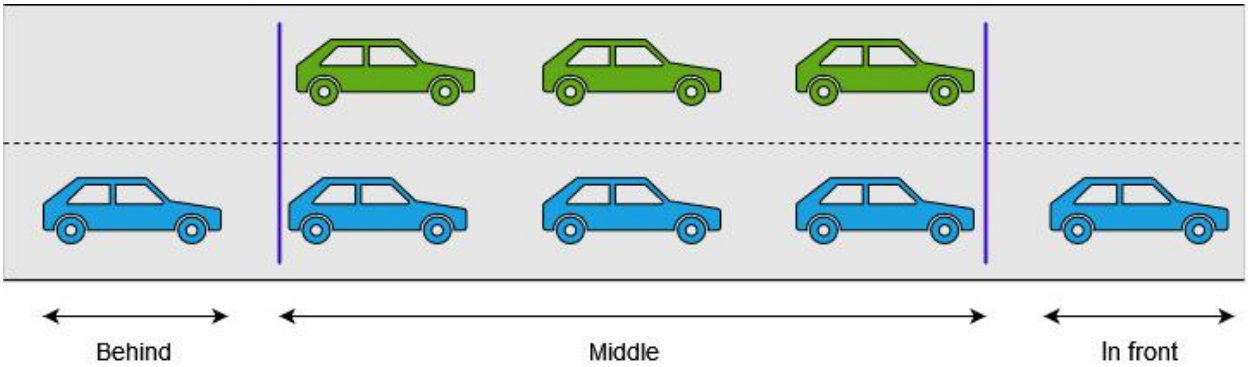


Figure 3.7: A scenario where vehicles are classified based on their position.

Having any V_m labelled as in-front means V_c is further up the road than V_c', this influences the gap each V_m labelled as middle will drive into. This is due to the time taken for each platoon to complete the creation of gaps. With each gap being created one after another, the platoon which is furthest up the road will end up having its V_m behind their opposing V_m'.

In figure 3.8, two scenarios of how this process works can be seen. In scenario 1, the blue platoon is further up the road, therefore when both platoons have finished creating gaps, the vehicles in the green platoon will be further up the road than the vehicles in the blue platoon. Taking V_{m+1} and V_m', these vehicles begin by being opposite to each other, then when creating gaps starts, V_{m+1} must slow down to create a gap between itself and V_m, but V_m' doesn't need to therefore it continues at its original speed which causes V_{m+1} to end up behind it. The same logic applies to scenario 2.

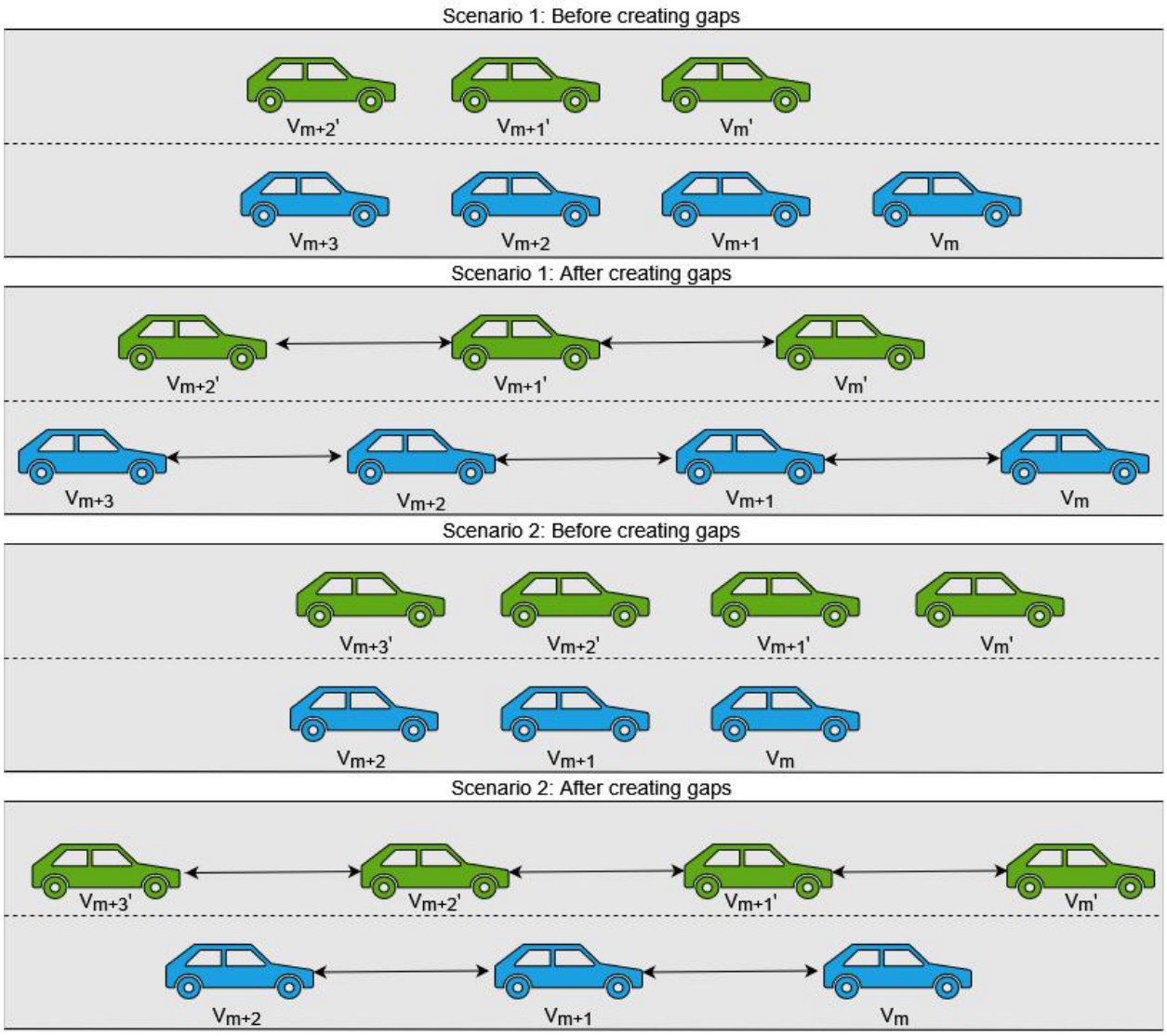


Figure 3.8: Two scenarios on how platoons create gaps depending on their position to each other.

With that consideration in mind, a group of vehicles are selected to create the gaps, these are called the vehicles to influence which includes all V_m that is labelled as middle, and a selection of V_m'. If the V_c' is furthest up the road, the selection of V_m' starts by selecting the closest V_m' to the first V_m labelled as middle (see figure 3.9). Otherwise, if V_c is further up the road, V_{m+1}' is chosen. Next, the preceding V_m' is selected to equal the number of middle V_m. The vehicles to influence chosen so far are instructed to create gaps of equal size. The V_m labelled in-front and behind are also instructed to space out to half the size of the vehicles to influence to prevent collisions when changing lanes.

Lastly, if either the first V_m or V_m' in the vehicles to influence is not a captain, their proceeding vehicle is added to the group. This is required in merge stage 2 to calculate the size of the gaps created by the first V_m and V_m' in the group.

In figure 3.9, two scenarios of how this process works can be seen. Selection of vehicles to influence depending on whether the blue platoon which wants to switch to the green platoons lane is further up the road or not.

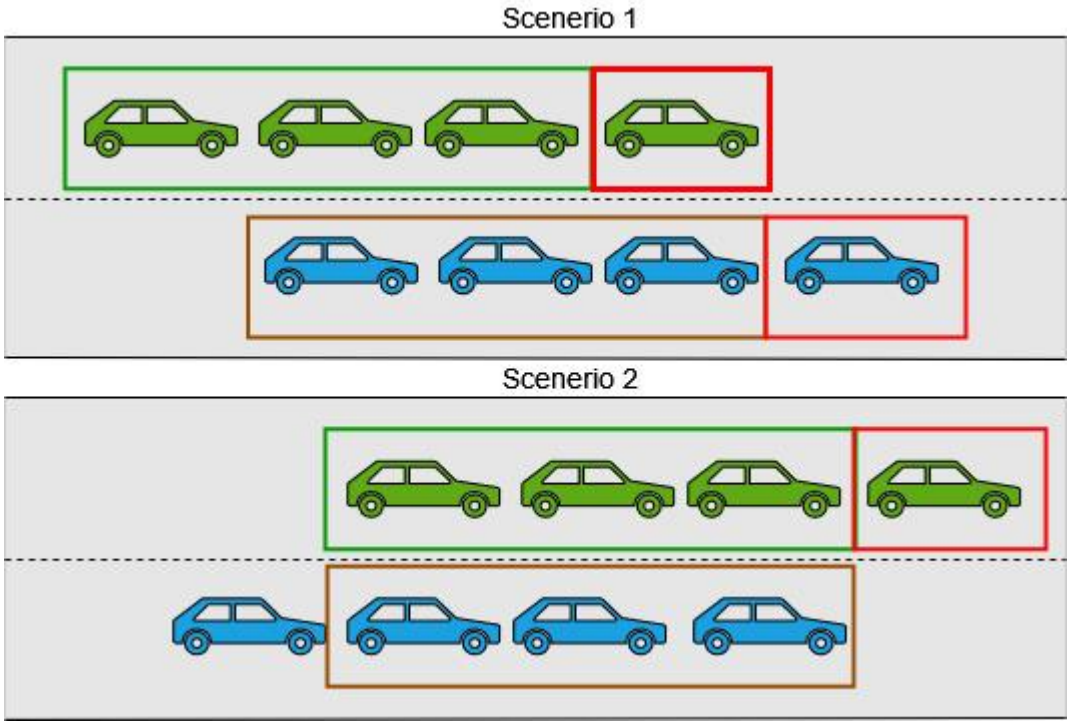


Figure 3.9: Two scenarios for selecting vehicles to influence depending on the positioning between two platoons.

Stage 2:

Each V_m and V_m' which was instructed to create gaps is constantly checked to see whether the space required to make is completed. Once all gaps are fully formed, the process for moving onto merge stage 3 is started. The process first determines the balance between vehicles to influence of both platoons. An equal number of vehicles to influence means that both V_c and V_c' are at the front of vehicles to influence, therefore, each V_m will drive into the gap in front of its opposing V_m'. A placeholder is placed in front of V_c' as V_c needs two points to position itself.

In figure 3.10, a scenario of how this process works can be seen. The vehicles have finished creating gaps, the vehicle at the front of the blue platoon will use the vehicle at the front of the green platoon and the placeholder (represented as a light green vehicle) to position itself in merge stage 3.

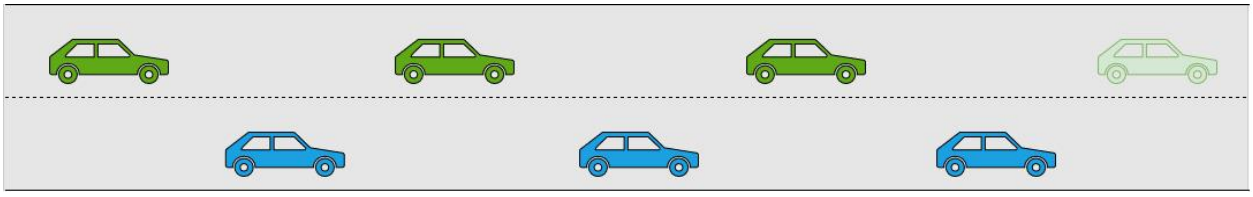


Figure 3.10: A scenario of what can happen when the gaps are created between the platoons.

Elsewise, if there are more V_m to influence than V_m', this indicates that the platoon of V_c is further up the road than the platoon of V_c', therefore as discussed previously, the first V_m in vehicles to influence is removed to cause the remaining V_m to drive into the gap behind their opposing V_m'. When there are more V_m' to influence than V_m, no additional changes are made.

Stage 3:

In this stage, speed adjustments are made to V_m to position them with the centre of the gap they are instructed to drive into. Positioning involves having a V_m calculate the distance to the two V_m' (or placeholder) surrounding its assigned gap (d_1, and d_2), when these values are equal then the V_m is perfectly between them, if d_1 is smaller than d_2, V_m is instructed to slow down, otherwise it increases its speed to reach an equal between d_1 and d_2. When all the differences between d_1 and d_2 are less than enough for all V_m, the V_m are instructed to drive into their assigned gaps. Finally, every vehicle is merged into the same platoon and the vehicle which is at the front becomes the new captain. An example can be seen in figure 3.11.

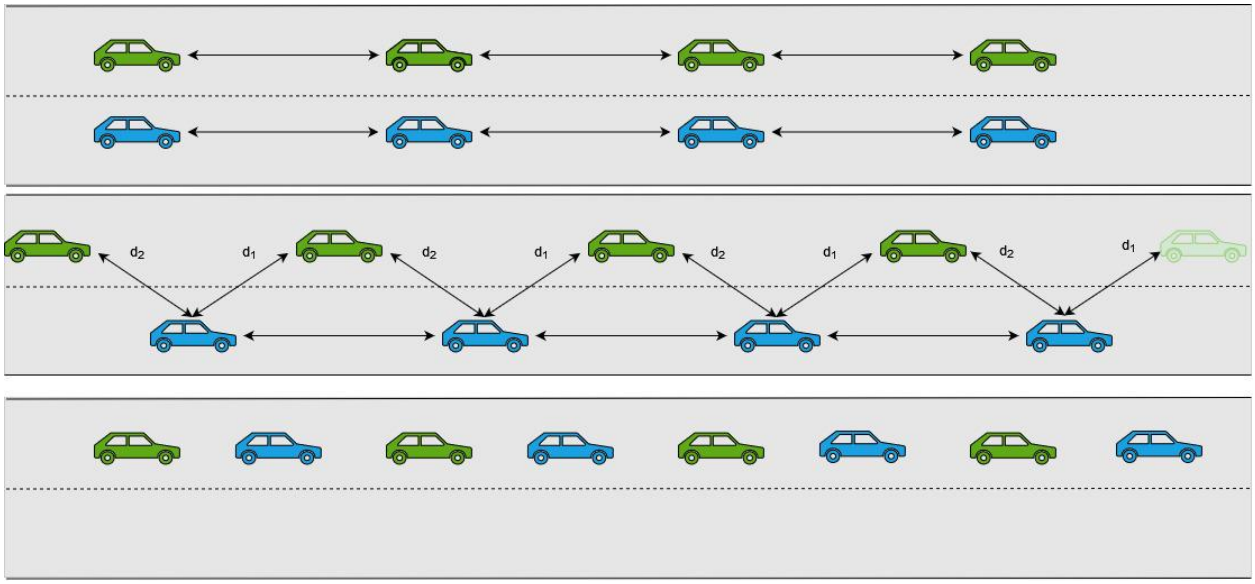


Figure 3.11: Merge stage 3 completed by two platoons of equal size and position.

3.3.7 Splitting Manoeuvre

```
Vehicle Vc:
   1. For each Vm in Vc members
   2.       If Vm has stopped at a traffic light
   3.             Split the platoon from the position of Vm in the
                  members list.
   4.             Vm will form a new platoon by taking all the member
                  vehicles behind it and become the captain.
   5.             Vc will retain the remaining members.
   6.             End Loop
```

Code 3.6: The process of splitting a platoon

At a set interval, each member vehicle (V_m) in the platoon of V_c will be checked to see if it has stopped at a traffic light. If a V_m besides the platoon captain has, then this means a portion of the platoon has been separated by a traffic light. This requires the platoon to be split into two. The V_m of the platoon which is stuck at the traffic light and is furthest to the front of the platoon will become a new platoon captain and take all the members of the platoon behind it, while the original platoon retains the remaining members. This process can be seen in code 3.6.

In figure 3.12, a scenario of how this process works can be seen. The two front vehicles of a platoon manage to cross a travelling light before it switches red which leaves two remaining members (green vehicles) of the platoon behind. The platoon captain (red vehicle) notices this and splits the platoon between the vehicle stuck at the traffic light and the last vehicle that left the traffic light. Two platoons are then created and the vehicle closest stuck at the traffic light becomes the new captain.

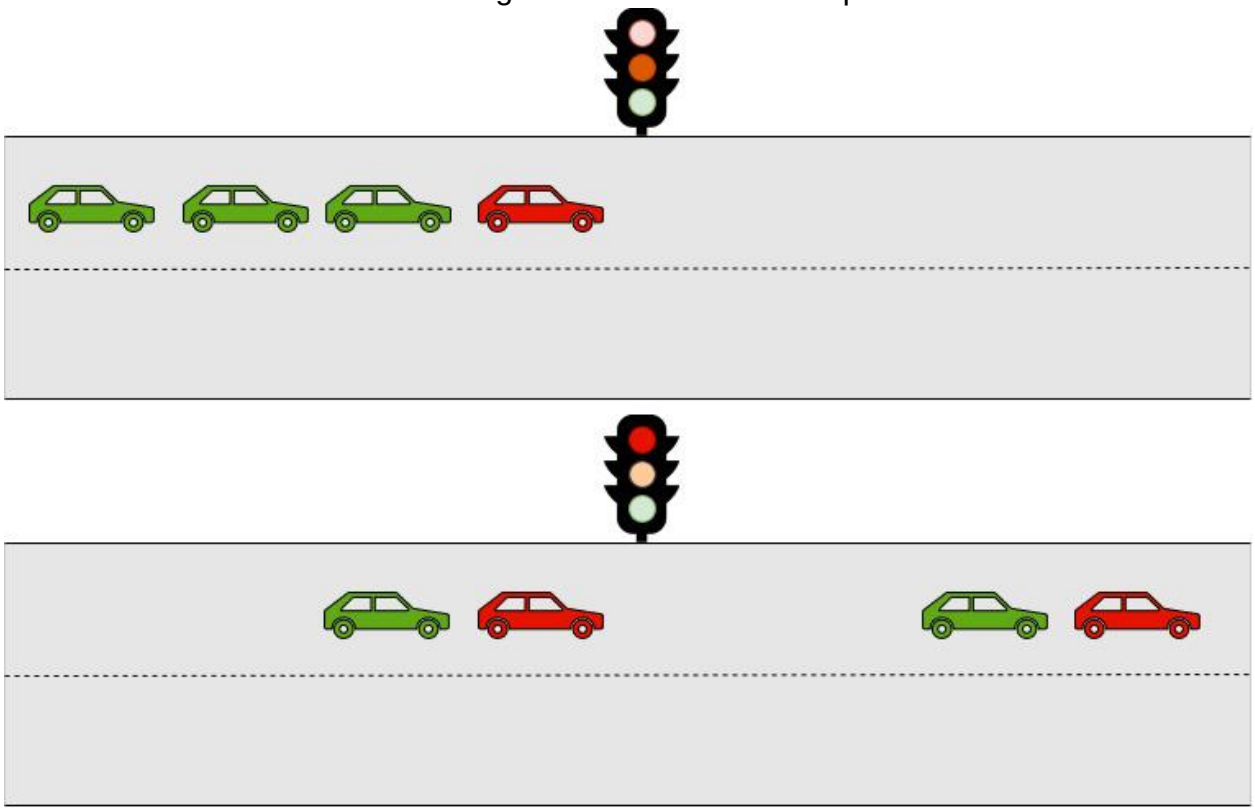


Figure 3.12: A scenario for a platoon splitting at a traffic light.

3.3.8 Leaving Manoeuvre

```
Vehicle Vc:
   1. If there are members in the platoon
   2.       Get a portion captain's route
   3.
   4.       For each Vm in Vc members
   5.             If Vm does not have the same portion of route
   6.                   Prepare Vm to leave the platoon

Vehicle Vm:
   1. While True:
   2.       Find location for when Vm will be in a different road and
            lane from the platoon
   3.       If Vm is not the rear vehicle of the platoon
   4.             If the rear vehicle and Vm are on the same road and
                  lane
   5.                   Prevent Vm from leaving
   6.             Else
   7.                   Allow Vm to leave
   8.       If Vm is at their leaving location and are allowed to
            leave
   9.             Have Vm disconnect from the platoon
   10.                  End Loop
```

Code 3.7: The process of either finding a platoon to join or forming a new platoon

At a set interval, a V_c will retrieve a portion of the next roads and lanes it will move to and compare it will each V_m. If a V_m does not have a similar portion of their route, the V_c will request that the V_m prepare to leave the platoon. Once requested, the V_m will locate the road and lane it will reach which would be different from the platoon route. If the V_m is on the rear of the platoon and reaches that exit location, it will disconnect from the platoon. In the case where it is not the rear, the V_m will wait until it is no longer between the platoon to disconnect as a premature disconnect would result in a large gap in the platoon. This process can be seen in code 3.7.

In figure 3.13, a scenario of how this process works can be seen. The green vehicle closest to the start of the platoon wants to take the next turn instead of following the rest of the vehicles, therefore the captain (red vehicle), instructs the green vehicle to prepare to leave. Once the vehicle reaches the point where it turns, it will first change roads, then disconnect from the platoon.

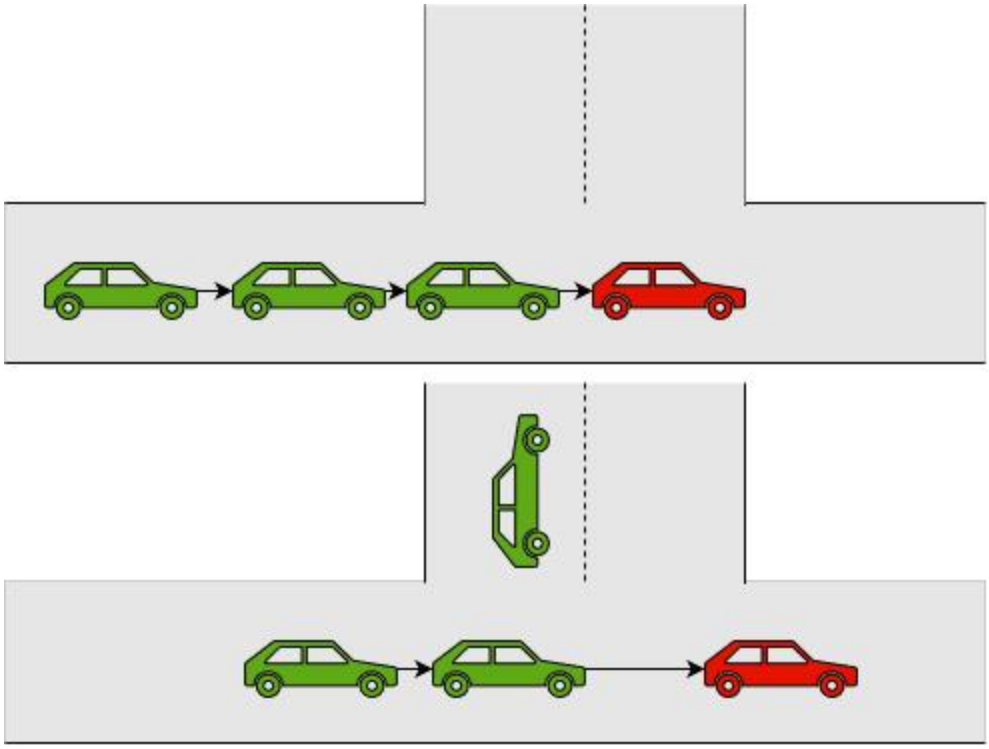


Figure 3.13: A scenario of a member vehicle leaving a platoon.

3.3.9 Captain Maintenance

```
Vehicle Vc:
   1. If there are members in the platoon
   2.       For each Vm in the platoon
   3.             If the distance between Vm and Vm+1 in front of it
                  is too large
   4.                   Vm+1 speed is increased using a multiplier
   5.                   If the multiplier is too large
   6.                         Reduce it to a reasonable value
   7.                   Set Vm+1 speed to the platoon speed with the
                        multiplier
   8.             Else
   9.                   Set Vm+1 speed to the platoon speed
```

Code 3.8: The process of either finding a platoon to join or forming a new platoon

When a V_m is in a platoon, the V_c is responsible for maintaining the distance it is from the member it is behind. This is completed by regulating the speed of each V_m. Starting from the front of the platoon, the distance between a V_m and the V_{m+1} in front is calculated, if this distance is too large, the V_c will increase the speed of the V_{m+1} by multiplying their current speed by a number so that it can reduce the distance between them. The value of the multiplier is dependant on the distance between them. The multiplier has an upper and lower limit to prevent the V_{m+1} from driving too fast or too slowly. This process can be seen in code 3.8.

3.3.10 Formula for speed and distancing

When creating an appropriate formula to regulate the speed of platoon members, some factors need to be considered. As discussed previously, limits are needed to prevent the change of speed in one vehicle from drastically interfering with other vehicles. A multiplier that causes a vehicle to drive too quickly could cause the vehicle to lose control, especially when driving around corners, additionally, a high speed will reduce the time a vehicle will be able to respond to events such as potential crashes. A multiplier that causes the vehicles to drive too slowly can cause a domino effect of vehicle breaking when the distance between two vehicles is smaller than the set platoon spacing. Most importantly, the formula should be dynamic by working with multiple set gap sizes.

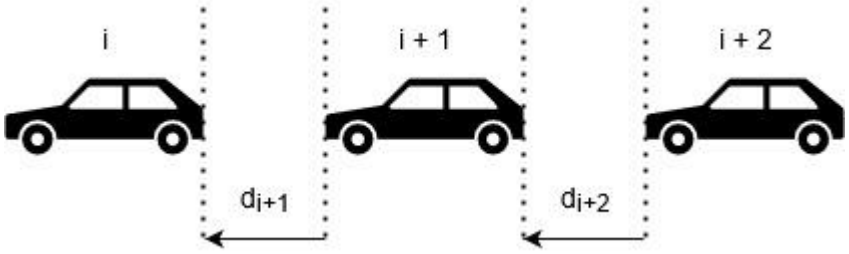


Figure 3.14: gap distance between members of a platoon.

Figure 3.14 represents a platoon, each member has its distance from the front of their vehicle to the rear of the proceeding vehicle calculated, this distance is defined as:

$$d_{i+1} = p_i - p_{i+1}, \quad 0 \leqslant i < N$$

With p_i and p_{i+1} representing the positions with i and i_{+1} respectively, and d_{i+1} being the distance between them. Each vehicle contains updated information of their location in x, y, and z coordinates, therefore the Euclidean distance between the vehicles can be calculated.

The formula uses a different set of rules to produce a multiplier depending on the spacing distance chosen by the platoon captain which is represented by s and d_{i+1}. This set of rules are defined as:

$$m_{i+1}(s) = \begin{pmatrix} (d_{i+1} + 1 - s)^{0.135}, & d_{i+1} \geqslant s \\ \dfrac{1}{(b^{(0.1 \times b)})}, \quad b = \sqrt{(d_{i+1} + 1 - s)^2}, & d_{i+1} < s \end{pmatrix}, \; 0 \leqslant i < N$$

Both sets of rules include the calculation $d_{i+1} + 1 - s$, this allows for the multiplier to not affect the speed of the vehicle when the d_{i+1} and s are equal. When the d_{i+1} is too large, the calculation is set to the power of 0.135, this is to have the value of the multiplier dramatically decrease from a large to small distance is to allow a vehicle to smoothly catch up to the platoon, instead of reaching the platoon with a large speed which creates the need to brake heavily. When d_{i+1} is too small, $d_{i+1} + 1 - s$ has a chance of producing a negative number which mathematically cannot be set to the power of 0.135, therefore another set of rules was created to continue a smooth change in speed. The boundaries of the multiplier are between and including 0.75 to 1.5.

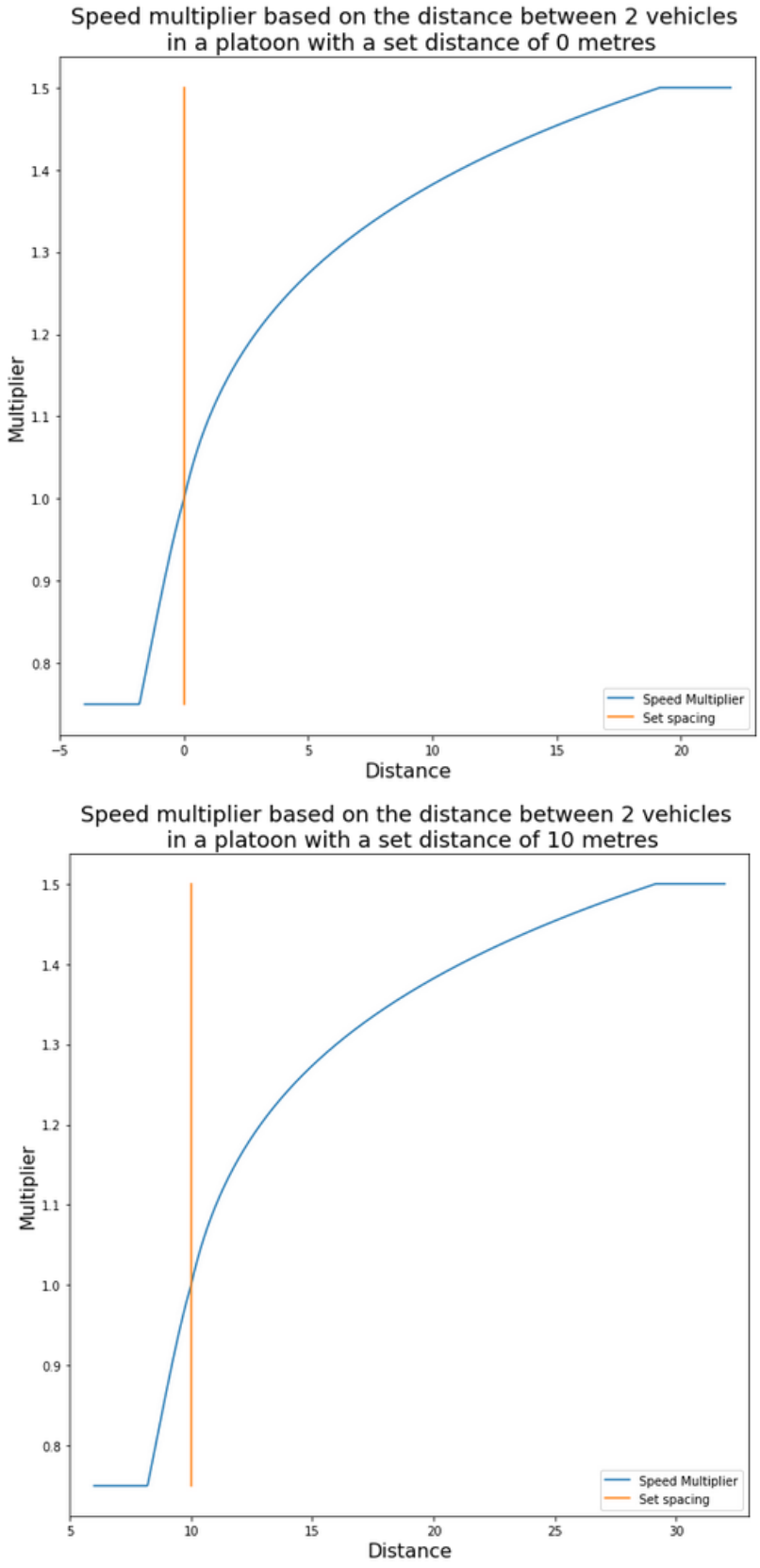


Figure 3.15: The multiplier set with a platooning distance of 0 and 10 metres.

To demonstrate the formula, it has been graphed with a platoon spacing of 0 and 10 metres as seen in figure 3.15. As discussed previously, the multiplier increases rapidly the larger d_{i+1} is from s and continues the same pattern of the rate of decrease when it is below s, as well as having a set lower and upper bound.

To validate the effectiveness of the formula, a platoon of 10 vehicles was spawned with approximately 3.5 metres between vehicles. A platoon distance of 0.5 metres was set and it took approximately 60 seconds for all the vehicles to reach the chosen distance. Once reached, the vehicles were able to maintain their distance by only having small variations of their speed, thus concluding that the formula and method used to regulate the distance between the vehicles are reliable and efficient. The results can be seen in figures 3.16 and 3.17.

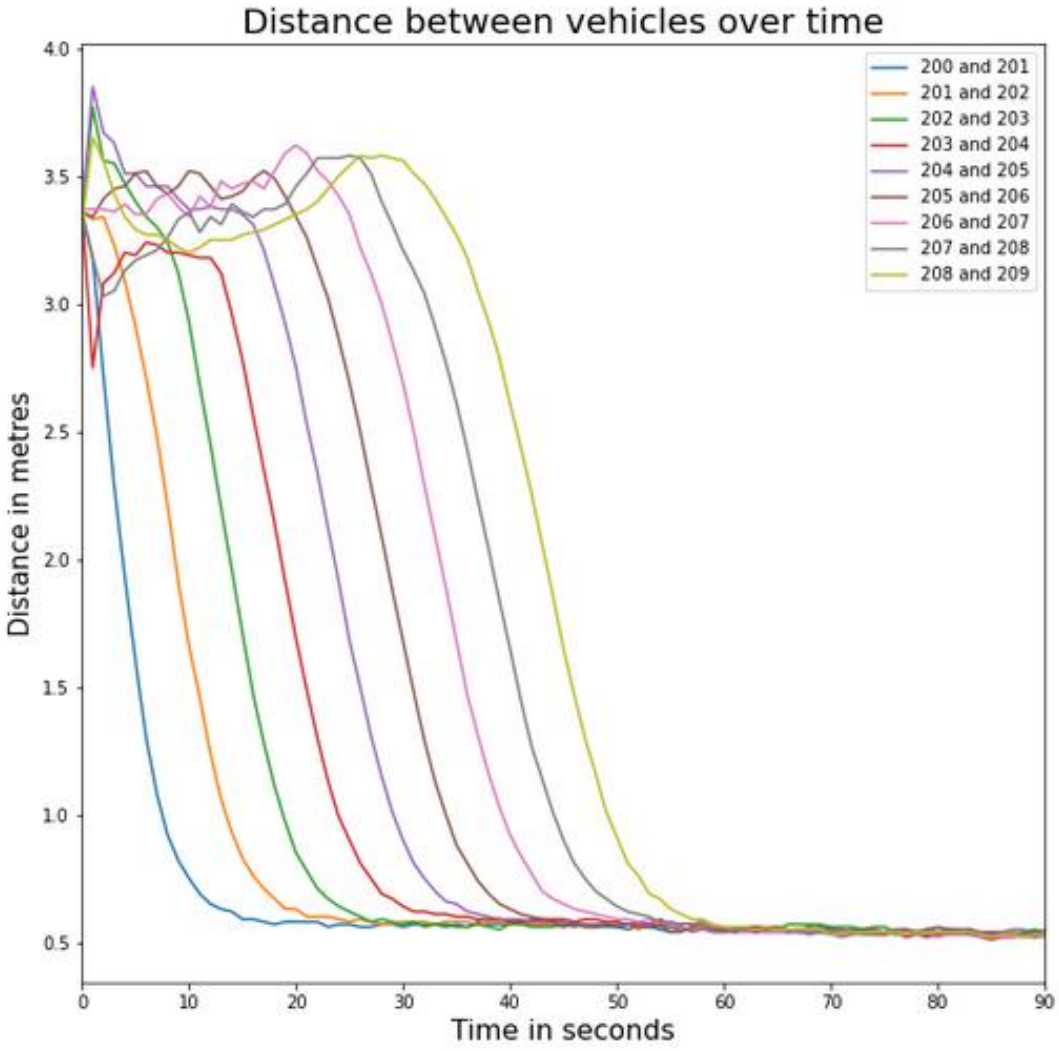


Figure 3.16: The distances between vehicles in a platoon

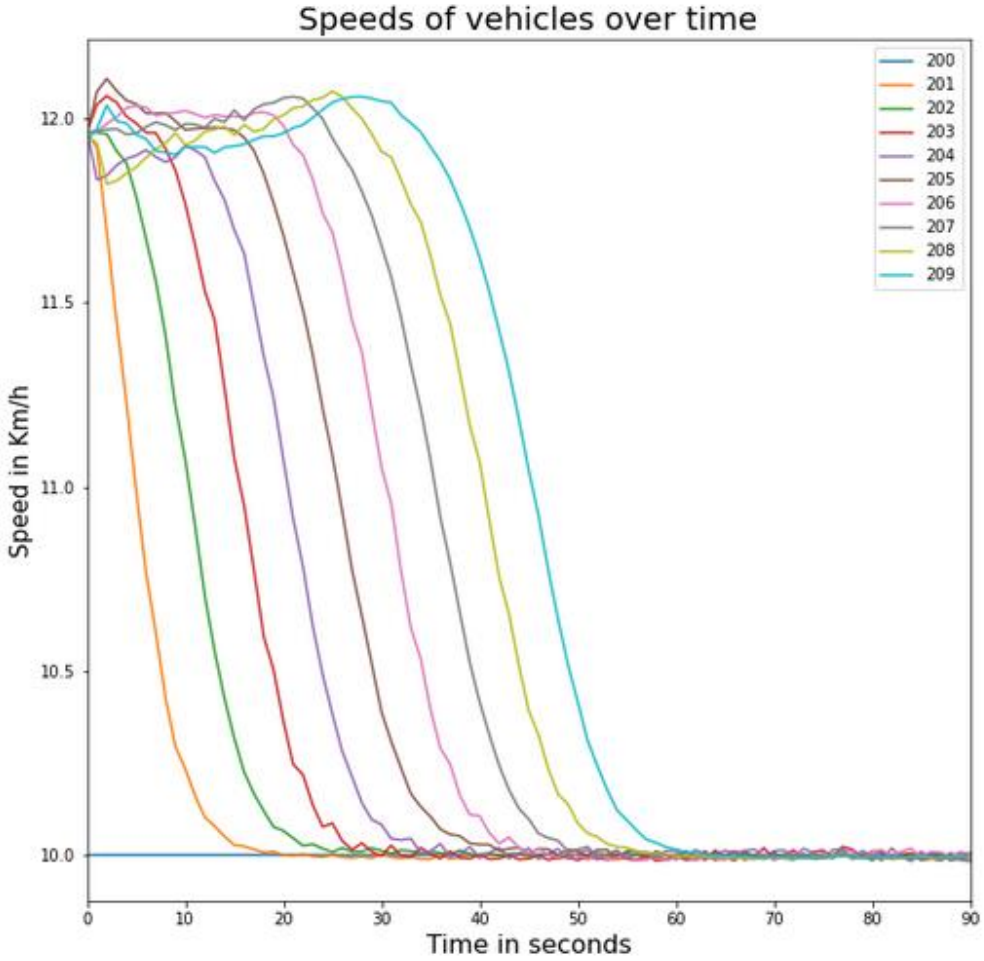


Figure 3.17: The speeds of vehicles in a platoon

3.4 Limitations

3.4.1 Performance of the system and architectures

Two different architectures were created to be combined to develop a realistic platooning simulation, a communication system based on the Mo-Zo architecture and a platooning structure based on the DUPP architecture with the addition of information from other papers. Before combining them, their performance was measured in loops per second (LPS) against vehicles spawned. In the simulation, a main loop of code is constantly run which controls the operations of the vehicles. The quicker this loop completes a cycle, the less likely vehicles would crash into obstacles or each other, and then become stuck.

Figure 3.18 includes the benchmark results from just having autonomous vehicles roam a road network, this should allow for a comparison of how each architecture has a toll on the performance of the simulation. With 40 vehicles spawned, the simulation could run around 12-13 LPS which is fast enough to control the vehicles.

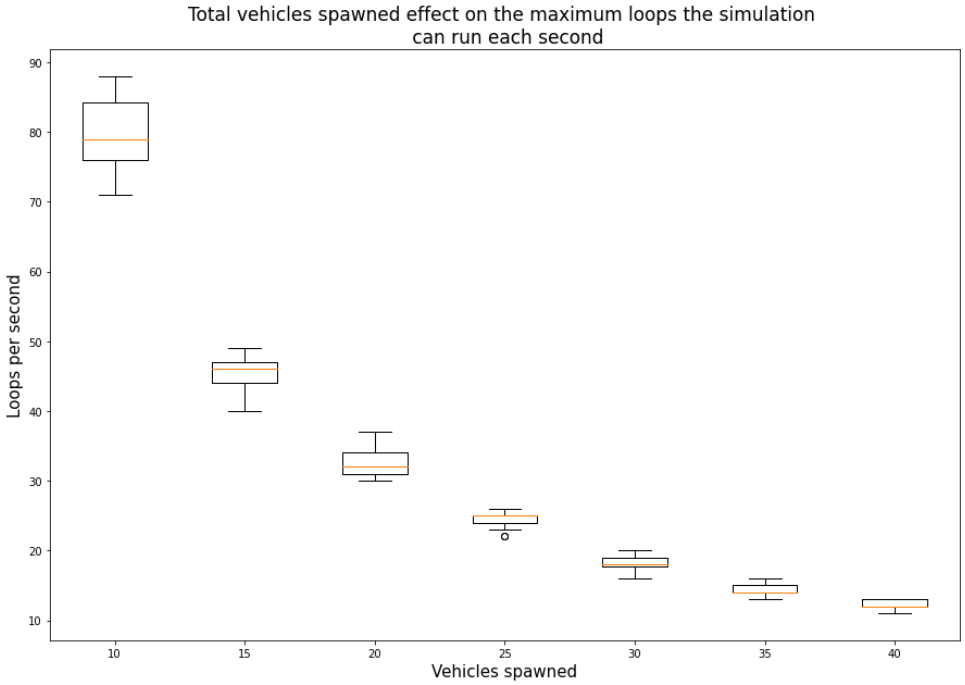


Figure 3.18: Maximum possible LPS depending on the number of autonomous vehicles spawned

In Figure 3.19, when comparing the platooning architecture to the benchmark, there is a minor difference between them. This indicates that the platooning architecture performance allows for close to the maximum number of vehicles to be spawned in and tested. When 40 vehicles are spawned in, an average of 11 LPS was recorded.

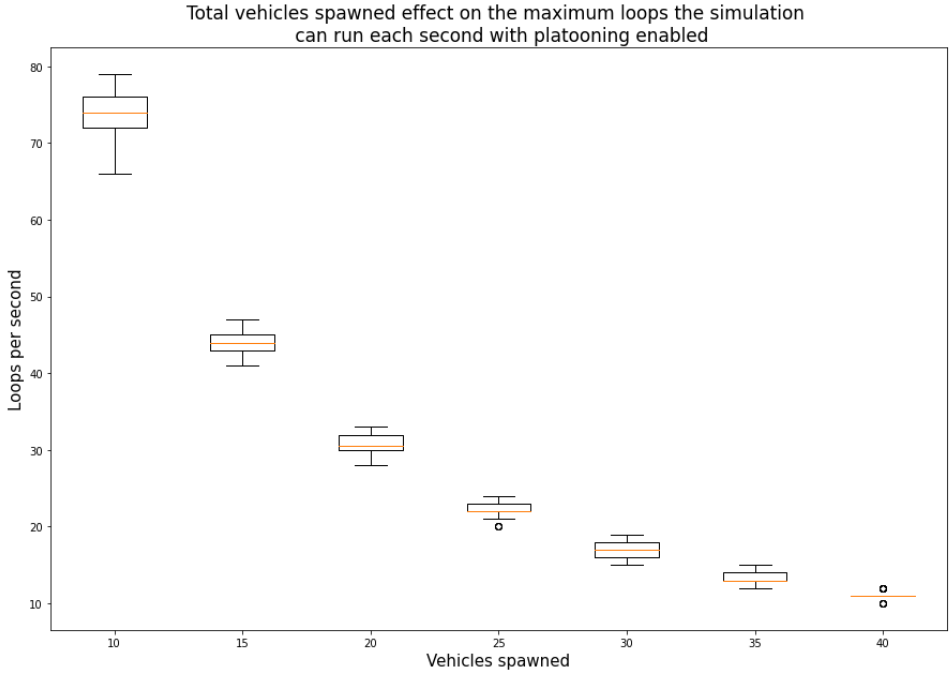


Figure 3.19: Maximum possible LPS depending on the number of autonomous vehicles spawned with platooning enabled

In Figure 3.20, there is a large drop in LPS throughout all number of vehicles spawned for the Mo-Zo architecture when compared to the benchmark and platooning architecture. With 30 vehicles spawned, the system runs as smoothly as the benchmark and platooning architecture at 40 spawned vehicles. This comes down to the features included in the V2V system such as B+-trees and messaging system, compared to the platooning system which allows vehicles to access and run each other's commands. Additionally, the next step was to create a realistic propagation channel to account for transmission attributes such as latency and messaging distance, but with the low LPS of the V2V architecture along, this would further reduce that speed. Furthermore, with the implementation of a vehicle-to-infrastructure system to provide better platoon pathing, the risk of limiting the number of vehicles that can be spawned reduces massively. Due to the limitations of the Mo-Zo architecture, and vehicles only needing to communicate with other vehicles around them in a short distance to platoon, the V2V system will be removed from the final product. Therefore, vehicles are assumed to have instant communication with each other.

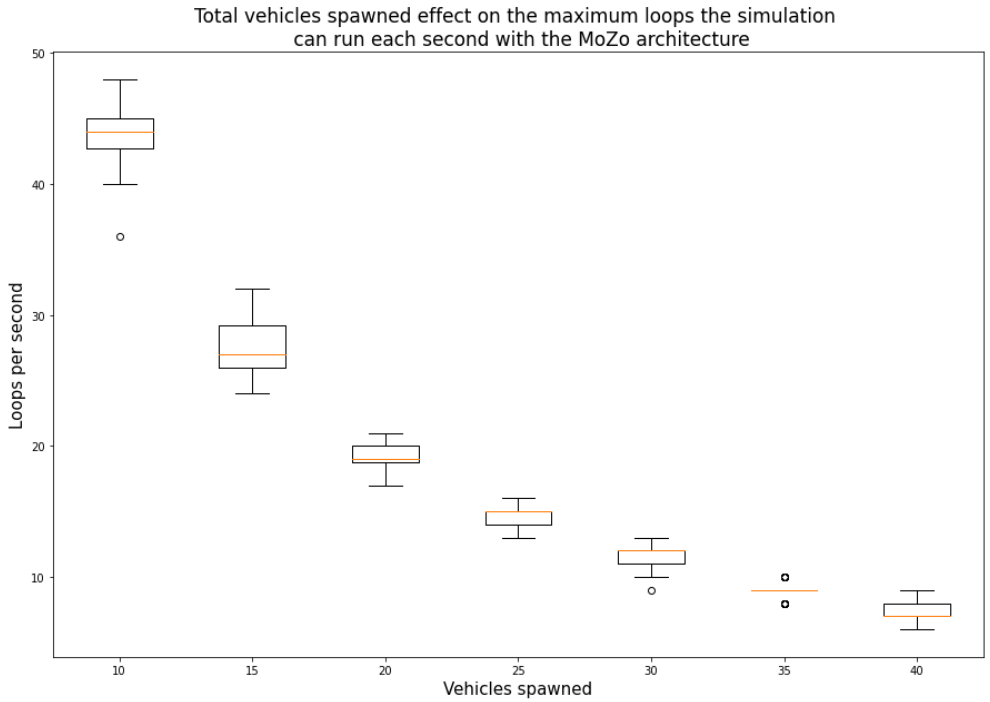


Figure 3.20: Maximum possible LPS depending on the number of autonomous vehicles spawned with the MoZo architecture.

3.4.2 Testing Environments

The Carla simulator provides 7 in-built maps for users to run vehicles in. These maps offer a range of road network types from a small layout consisting of "T junctions" to a complex road network with a 5-lane junction, a tunnel, a roundabout, and more. Although the complexity allows for testing the platooning architecture in multiple environments, the

maps besides one prevent testing of a very compact road network due to the limitations on the number of vehicles that can be spawned. There are two options to create new maps but are not viable for different reasons.

Option 1: OpenStreetMap

OpenStreetMap (OSM) is a collaborative project that holds a geographic database of the world which allows for road networks to be selected and then imported into Carla. The issue with this approach is that it is unstable with the simulator which can cause Carla to crash. It also lacks simple road infrastructure such as traffic lights. The documentation does produce a method to generate traffic lights at junctions if the OSM file is converted to OpenDrive format but currently the method described is not functioning. An example of a map generated can be seen in figure 3.21.

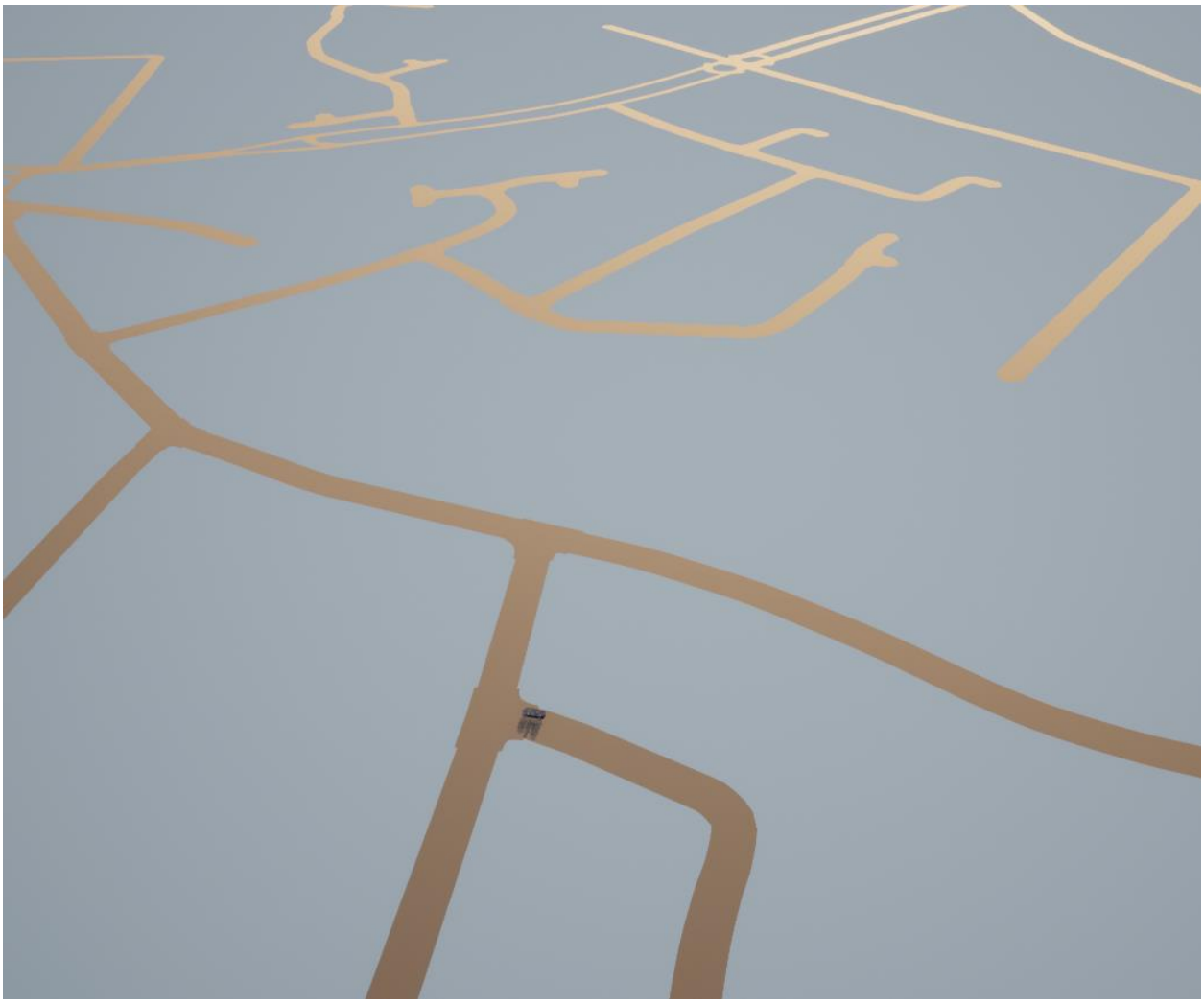


Figure 3.21: Map generated with OSM and imported into Carla

Option 2: RoadRunner and Unity

RoadRunner is software from Vector Zero to create 3D scenes. The software allows the creation of maps to be directly exported for Carla, unfortunately, the software is not publicly available. Students are allowed access to a trial but are required to fill out a form

and wait to be contacted. As of sending a form on the 8th of August 2021, and repeating the process on the 11th of August 2021, the software provider is yet to get in contact to provide a trial. With Unity, the Carla documentation offers a tutorial to create maps for Carla, but binaries in Filmbox format contain all the meshes to build the map such as roads and sidewalks, and road network information for cars to circulate the map in OpenDrive format are prerequisites and are not available. Additionally, some folders such as the BaseMap which the tutorial references to are not available in the Carla package.

4. Platooning results

4.1 Single-lane platooning

Single-lane platooning refers to an environment that includes roads that only have two lanes for travelling in opposite directions. Simulations are performed to evaluate the platooning architecture against a benchmark that includes just autonomous vehicles travelling. In all simulations, the vehicles select a random destination to drive to, once they have reached their destination, the process repeats. Although it is preferable to run the simulations for a long time, the roaming system of the vehicles can cause them to take sudden illegal U-turns on the roads once they have selected a new destination, causing them to either collide with another vehicle, traffic light, or traffic sign and get stuck. Due to limitations of the communication architecture, it is assumed that vehicles have an unlimited communication range with no delays between themselves. This would not affect the results as vehicles would only need to communicate with other vehicles on the same road.

The desired speed of vehicles was set to 10 Kmph with a maximum of 15 Kmph to allow for a low number of loops per second which was set at 9 and for 40 vehicles to be spawned. Vehicles in a platoon are set to create a gap of 1 metre between themselves which is controlled by the captain. Platoon members themselves are responsible for braking when they get too close, their proximity is set to 6 which leaves around a 0.5-metre distance between the vehicles when have come to a rest. Platoon leaders' proximity is set to 10 which leaves around a 4-metre gap when they have braked. Each vehicle is equipped with a timer to use when they are a platoon leader to perform checks. Every 0.25 seconds, platoon leaders will adjust the speed of their members to maintain a short distance between themselves, to check whether a split or merge can take place and whether a vehicle needs to leave. To prevent the potential roaming error from happening, simulations are set to 2 minutes. These parameters are displayed in table 4.1.

Parameters		
Attributes of vehicles	Platooning	Benchmark
Desired speed	10 km/h	10 km/h
Maximum speed	15 km/h	-
Platooning distance	1 metre	1 metre
Platooning Spacing	6 metres	-
Platoon Proximity	10 metres	-
Vehicle Proximity	-	10 metres
Distance till join	20 metres	-
Check intervals	0.25 seconds	-
Loops per second	9	9
Spawn amount	40	40
Run time	2 minutes	2 minutes

Table 4.1: Parameters for single-lane platooning.

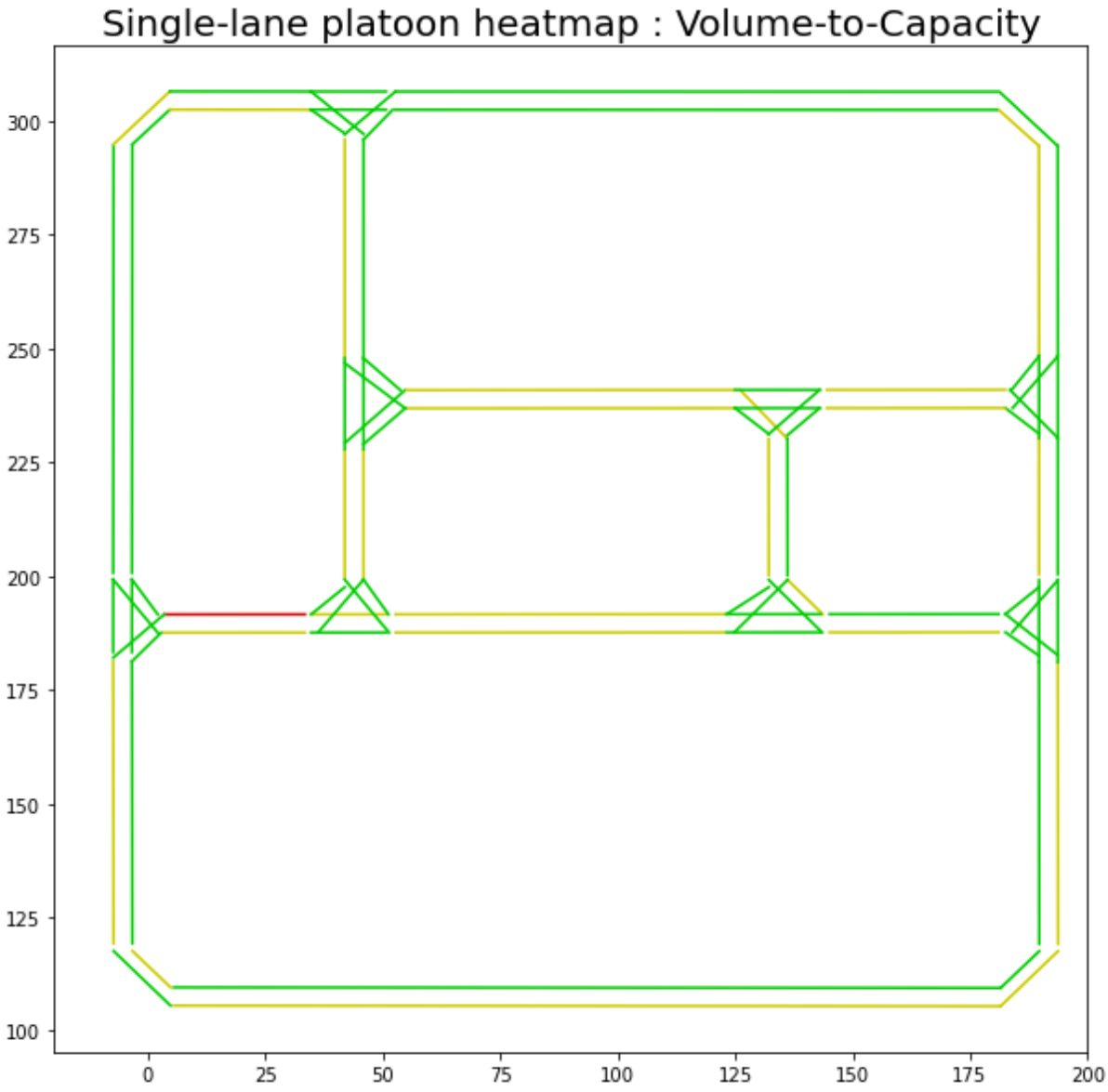


Figure 4.1: Heatmap of lanes for the volume-to-capacity of single-lane platoons roaming for all 10 simulations

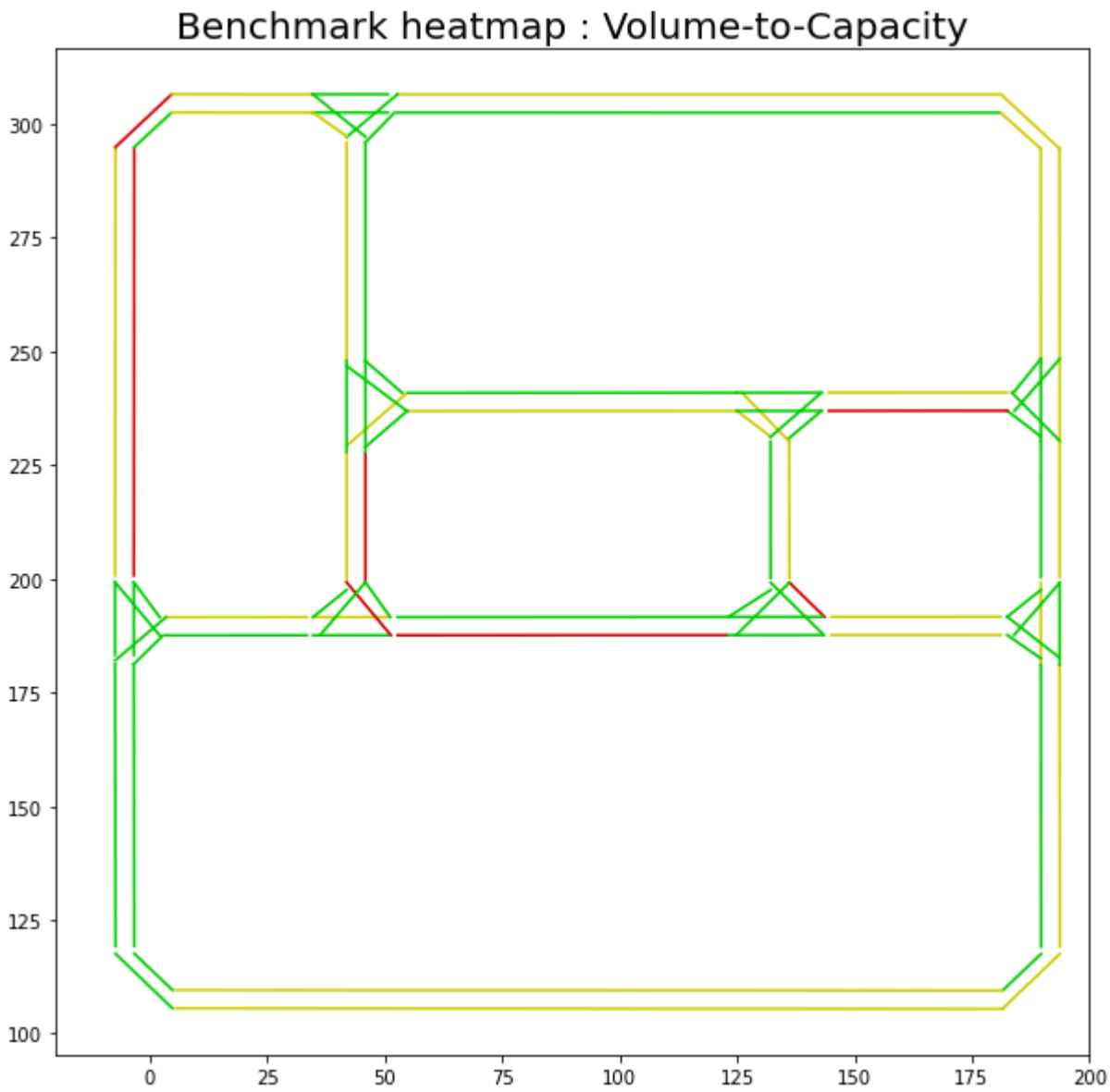


Figure 4.2: Heatmap of lanes for volume-to-capacity of autonomous vehicle roaming for all 10 simulations

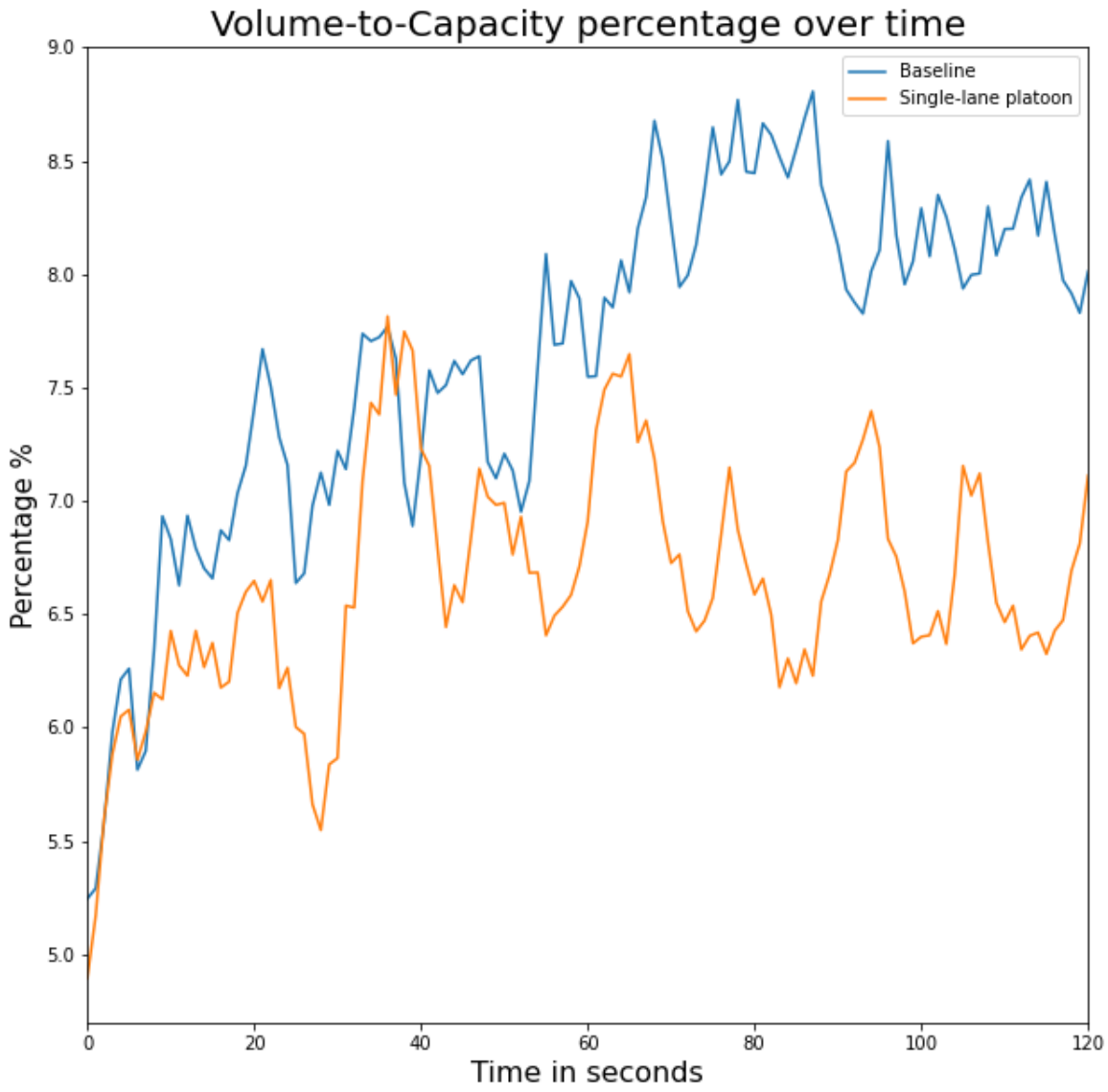


Figure 4.3: Average volume-to-capacity of all lanes for each second for all 10 simulations for both the autonomous vehicles roaming with and without platooning

4.1.1 Discussion

Figures 4.1 and 4.2 are the heatmaps of the road network to show the effect of the volume-to-capacity for the entire simulation for both platooning and autonomous vehicles roaming respectively. The purpose of the heatmaps is to provide a visual aid of what areas of the road network were affected the most. Three colours indicate the level of volume-to-capacity a lane has experienced throughout every simulation compared against the maximum volume-to-capacity recorded. Green indicates that the lane experienced 30% or under, yellow indicates that the lane has experienced between 30% to 70%, and red indicates that the lane has experienced 70% or over.

Comparing single-lane platooning in figure 4.1 to the benchmark in figure 4.2, there is a clear difference observed. Single-lane platooning only achieved at least 70% of the maximum volume-to-capacity once where the benchmark achieved that level on 7

different lanes. Both single-lane platooning, and the benchmark has a similar number of lanes between 30% to 70% of the maximum volume-to-capacity, but single-lane platooning had a higher volume of lanes under 30% of the maximum volume-to-capacity.

Moving on to Figure 4.3, when comparing the average volume-to-capacity per second of the benchmark and platoon, there is a clear difference between the results. Viewing table 4.2, on average each second, with platooning enabled, there was a 15% increase in road capacity when compared to the benchmark. Additionally, platooning also had a lower maximum and minimum road capacity when compared to the benchmark.

	Benchmark	Platoon
Mean	7.64	6.63
Maximum	8.81	7.81
Minimum	5.25	4.90
Standard deviation	0.76	0.52

Table 4.2: Numerical results from single-lane platooning

4.2 Multi-lane platooning

As single-lane platooning has shown that platooning is beneficial, the architecture was expanded upon and multi-lane merging was included. With single-lane platooning, platoon captains had to only consider merging in one dimension (behind and in-front of them), adding a second dimension increases the complexity and amount of decision making that needs to be completed. The purpose of the test and simulation would be to determine if multi-lane merging is stable. Figure 4.4 showcases the five scenarios which will be tested. The purpose of different positions verifies that the architecture for multi-lane platooning will be stable no matter the positioning of both platoons. Each scenario was run 10 times until a single platoon was created and the vehicles reached the platooning distance set. The parameters for multi-lane platooning can be seen in table 4.3.

Parameters	
Desired speed	10 km/h
Maximum speed	15 km/h
Platooning distance	0.5 metres
Platooning Spacing	0 metres
Platoon Proximity	0 metres
Vehicle Proximity	-
Distance till join	20 metres
Check intervals	0.25 seconds
Loops per second	9
Spawn amount	10

Table 4.3: The parameters used for multi-lane platooning.

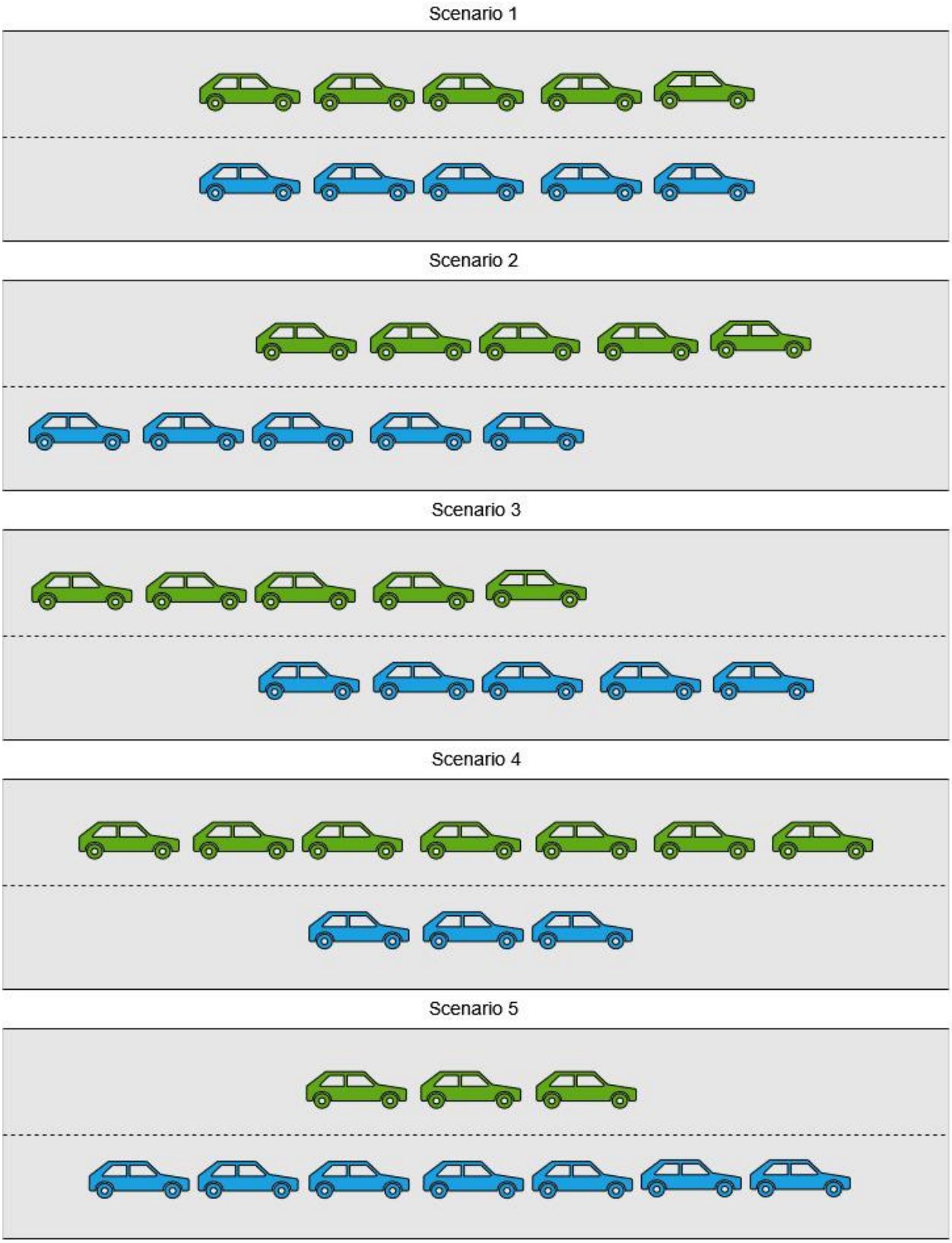


Figure 4.4: The scenarios use to evaluate Multi-lane platooning.

4.2.1 Scenario 1

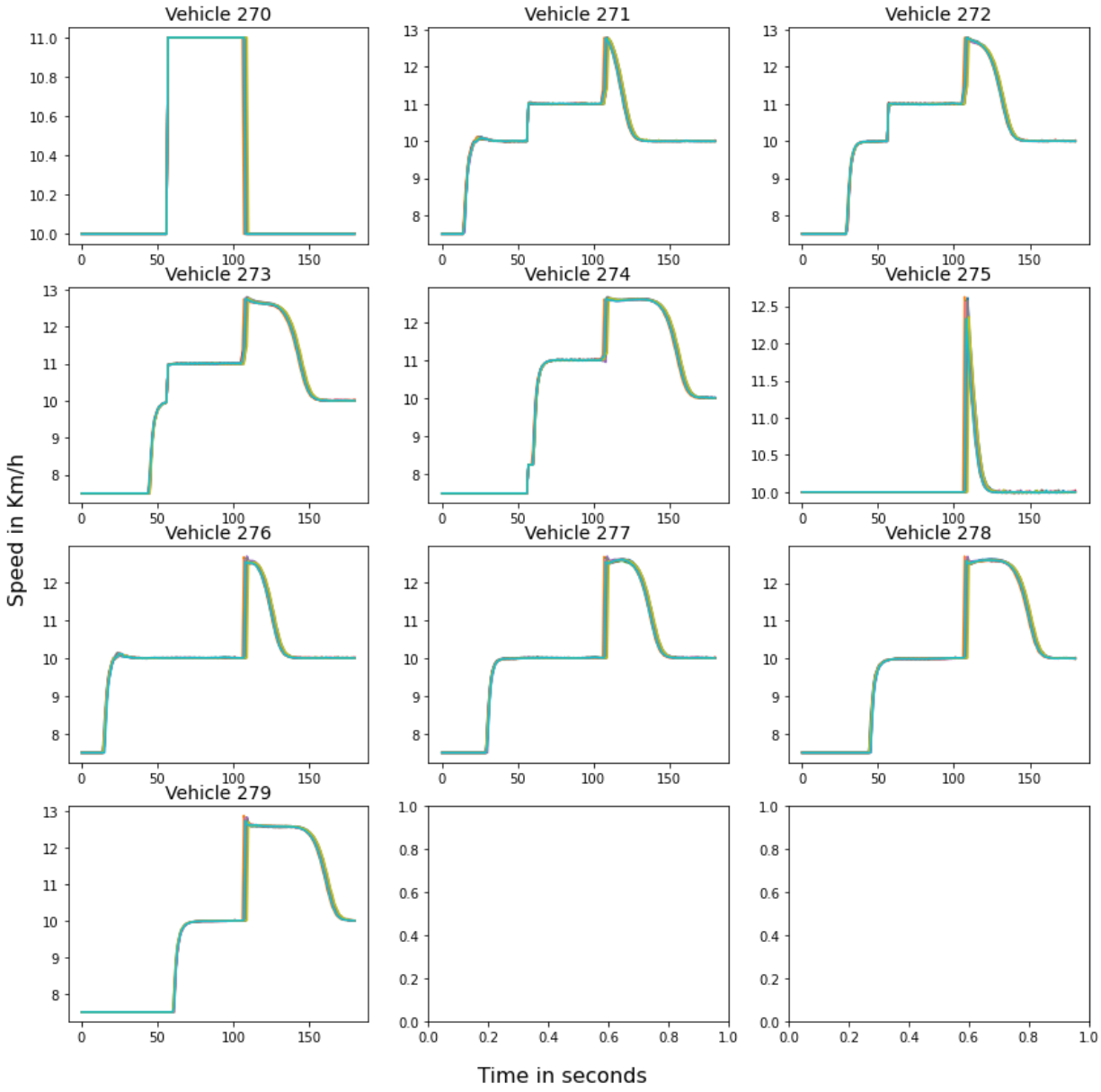


Figure 4.5: Speeds of vehicles in scenario 1

4.2.2 Scenario 2

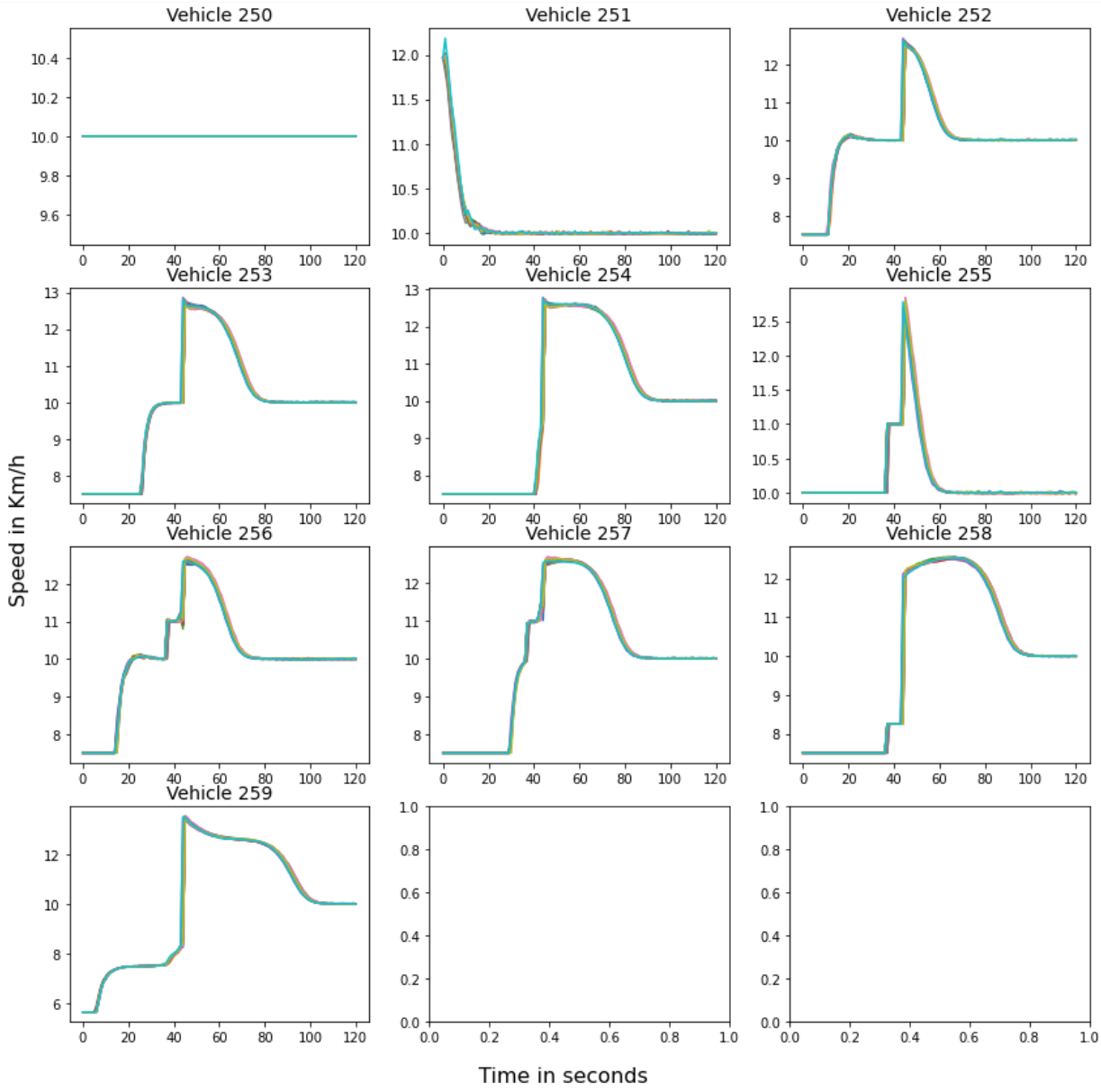


Figure 4.6: Speeds of vehicles in scenario 2

4.2.3 Scenario 3

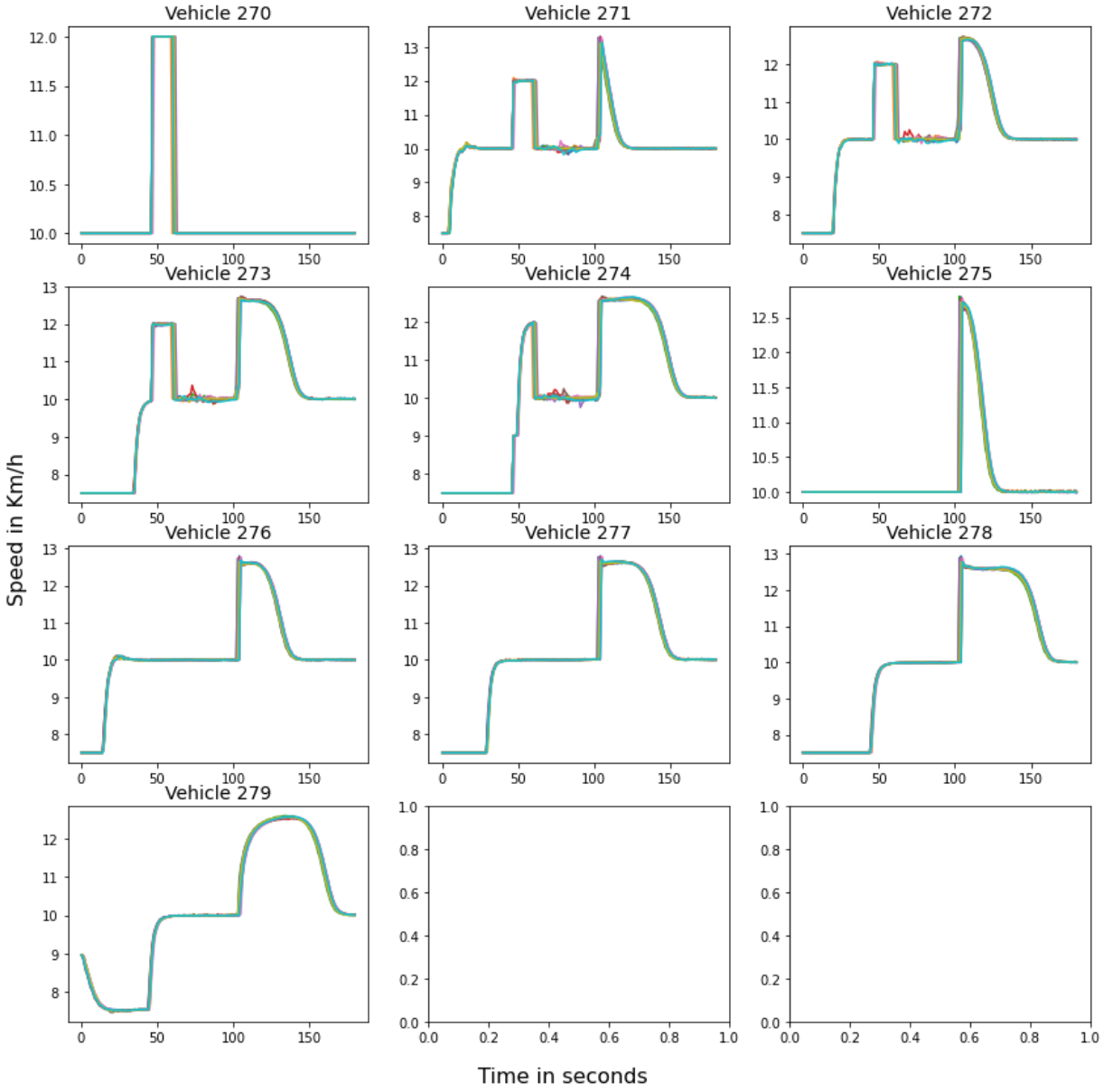


Figure 4.7: Speeds of vehicles in scenario 3

4.2.4 Scenario 4

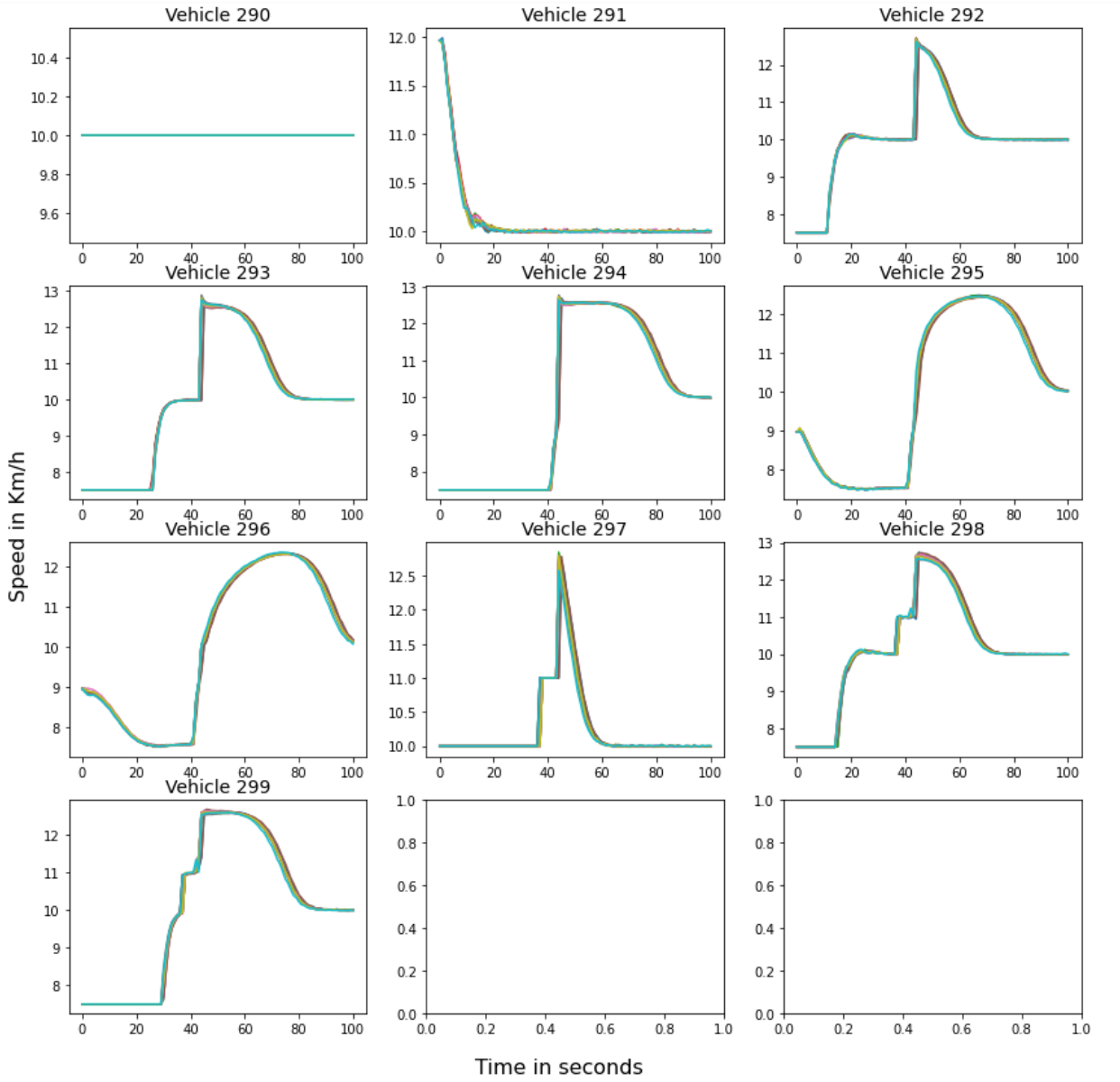


Figure 4.8: Speeds of vehicles in scenario 4

4.2.5 Scenario 5

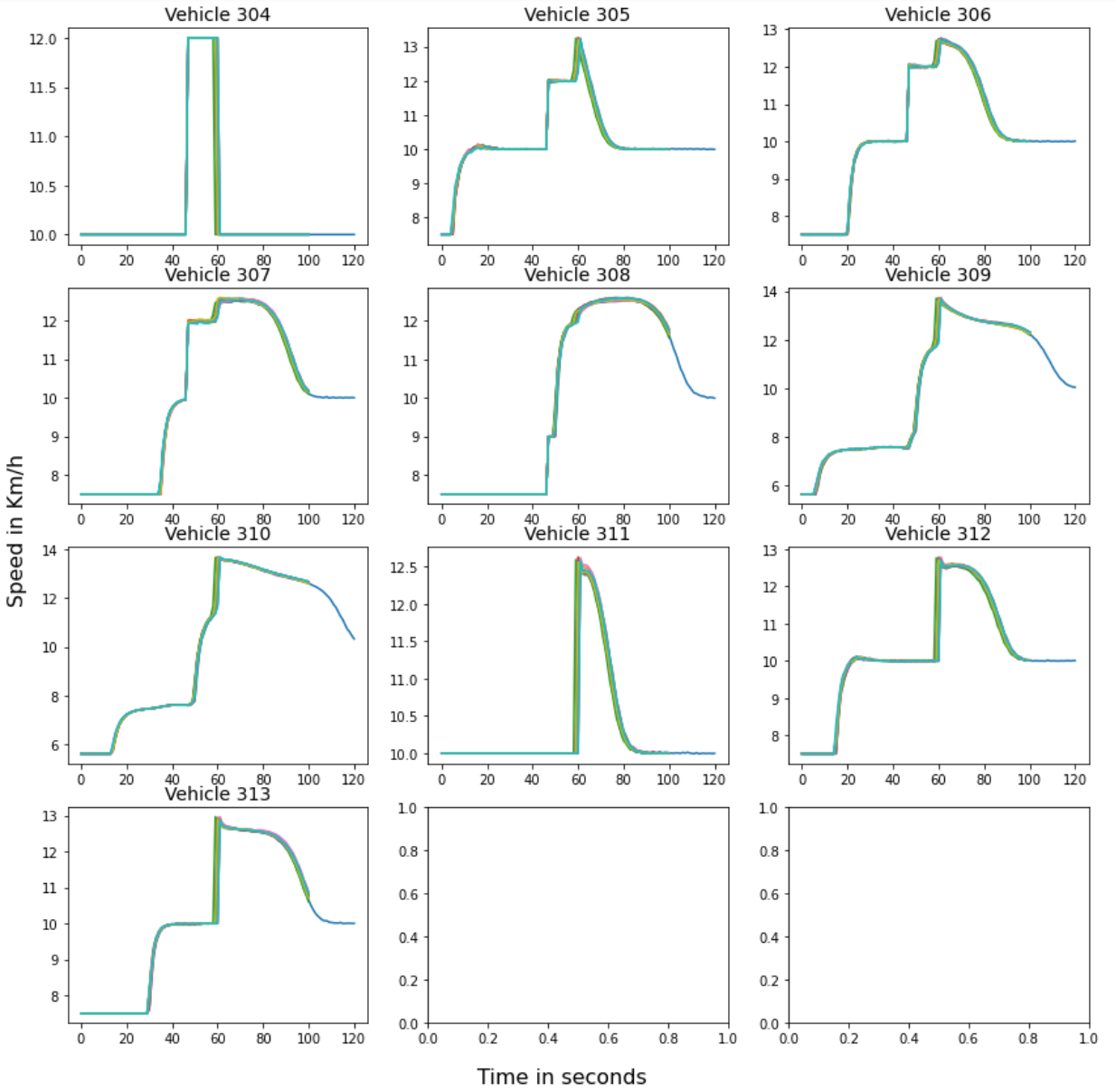


Figure 4.9: Speeds of vehicles in scenario 5

4.2.6 Discussion

Looking at figures 4.5 to 4.9, each graph in these figures represents the speed of a vehicle throughout the merge. For each simulation performed in each scenario, the architecture performed also perfectly as all the vehicles followed the same routines which can be seen as the lines in the graphs are identical. Although at some points the vehicles were instructed to rapidly alter their speed, this only happened a few times a minute. The most rapid fluctuations of speed can be seen in figure 4.7 for scenario 3 between 50 and 100 seconds for vehicles 270 to 274, but these where changes were within a fraction of a Km/h which would not be noticeable for passengers within the vehicle.

5. Software development lifecycle

With the project taking place over months, a suitable plan needed to be created to prevent the project from failing to be completed. Failure could happen by either spending too much time on a component of the project or including too many objectives, therefore an agile approach was taken. This approach would allow for responding to change as the project would be split up into small increments to minimise the initial planning and design. After researching the problem of congestion and the broad idea of how to solve the issue by coordinating vehicles, I defined the main objectives to achieve.

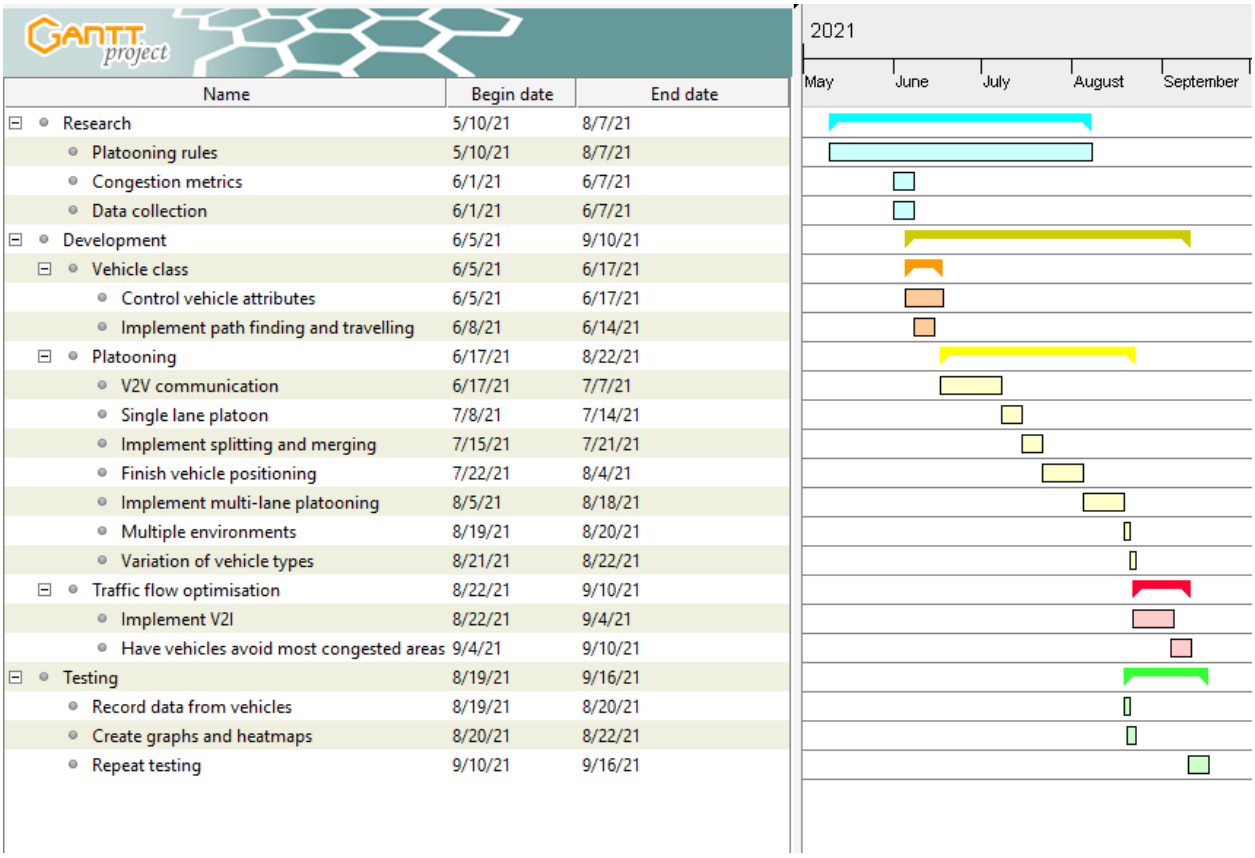
Objectives

The main requirement of the project is to create a platooning architecture for the CARLA simulator which is feasible. The objectives include:

1. Creating a python class to represent and control vehicles. This would involve altering the speed, direction, acceleration, as well as implement pathfinding and travelling feature.
2. Implementing realistic communication between AVs, examples of this would be having AVs purely communicate with each other, or have AVs connect and message each other through road infrastructure. Although vehicles communicating with each other directly is reliable, a messaging infrastructure would allow for an increased messaging range.
3. Allowing vehicles to form platoons. This is the main objective of the project as both the vehicle class and communication methods are used to allow vehicles to follow a set of rules to form platoons as well as split from and merge into them.
4. Creating a traffic flow optimisation system. This would include implementing vehicle to infrastructure to allows platoons to take paths that are the least congested.

The objectives are in the order of what needed to be completed. The vehicle class is the first part of research and development as its representation is needed to add communication and platooning. The vehicle class is required to implement platooning, and communication should be implemented with it. Finally, platooning must be finished before traffic flow optimisation is worked on. Objectives 1 and 2 are feasible therefore can be completed, but objective 3 is feasible depending on the complexity chosen. Due to the “perfect” platoon architecture very likely being impossible to compete within the dissertation time frame, an agile approach to project management would allow it to constantly be improved over time instead of creating a set plan. The final objective’s feasibility is the lowest as although it is not essential for the project, it would have some benefit to reducing congestion, therefore it may not be included in the final architecture, but the theory might be available in the final report. Although the Gantt chart seems to have a linear pattern representing the waterfall model, an agile project management approach will be used for the project. Most of the points in the platooning development period are what was hoped to be completed. Research, design, and development done together to allow literature read to be applied to the architecture simultaneously.

The Gantt chart seen in figure 5.1 was created using the software GanttProject to organise the development lifecycle which allows time constraints to be added to the objectives to ensure the project gets completed within the dissertation time frame.



Name	Begin date	End date
Research	5/10/21	8/7/21
Platooning rules	5/10/21	8/7/21
Congestion metrics	6/1/21	6/7/21
Data collection	6/1/21	6/7/21
Development	6/5/21	9/10/21
Vehicle class	6/5/21	6/17/21
Control vehicle attributes	6/5/21	6/17/21
Implement path finding and travelling	6/8/21	6/14/21
Platooning	6/17/21	8/22/21
V2V communication	6/17/21	7/7/21
Single lane platoon	7/8/21	7/14/21
Implement splitting and merging	7/15/21	7/21/21
Finish vehicle positioning	7/22/21	8/4/21
Implement multi-lane platooning	8/5/21	8/18/21
Multiple environments	8/19/21	8/20/21
Variation of vehicle types	8/21/21	8/22/21
Traffic flow optimisation	8/22/21	9/10/21
Implement V2I	8/22/21	9/4/21
Have vehicles avoid most congested areas	9/4/21	9/10/21
Testing	8/19/21	9/16/21
Record data from vehicles	8/19/21	8/20/21
Create graphs and heatmaps	8/20/21	8/22/21
Repeat testing	9/10/21	9/16/21

Figure 5.1: Initial Gantt chart for the project.

6. Software testing

The testing process in respect to the project plan, allowed for elements of the project to be programmed and tested concurrently. Having programmed and designed the architecture, the appropriate method of testing was White-box testing. This included a mixture of unit testing, integration testing, regression testing, and system testing. There was some consideration to using black-box testing, but it was found to be unnecessary as there were only two inputs for the program, loading the world and vehicles. Additionally, visual debugging tools were utilised, this allows for the directions of vehicles as well as their roles (roaming, member, and captain) and links between vehicles in a platoon to be displayed while observing how changes affect the architecture. In appendix 2, vehicles with a blue box around them are joining a platoon, a green box represents a member of a platoon, and a red box represents a captain.

Unit testing

Unit testing is where individual units or components of the software are isolated and tested to validate that the code performs as expected. A test performed for the vehicle class was to make sure that when a vehicle was instructed to travel at a certain speed that it would reach and maintain it. When developing different methods, regression testing was performed to make sure the changes did not affect any other components of the system. For example, when adjusting the method used to create gaps by automatically altering the speed of vehicles, different manoeuvres and methods were performed to determine if the changes affected them.

Integration testing

Integration testing is where individual software modules are tested as a group to determine the outputs of their interactions are as expected. This would involve testing both the vehicle and Carla module combined. In figure 3.2, a data flow chart for how the combined vehicles and platoon modules were created, to validate that the system follows the design, a control flow test was performed.

System testing

System testing is where the complete integrated system is tested to determine if it has met the objectives and requirements. Non-destructive testing was completed, with the parameters of the software chosen, different platoons were observed to determine if any actions caused unexpected results. In the case errors did occur, the process of starting at unit testing to integration testing was repeated to determine the source of the error. In table 6.1, the result and evidence of the requirements of the final architectures can be seen. The vehicle and single-lane platooning architectures were completed successfully which makes the project a success as they were crucial to solving the dissertation problem. Although the multi-lane merging was successful, it was classified as semi-successful as it could be used to collect useful data, but it was tested in set scenarios instead of being implemented into the original platooning architecture. The communication architecture failed as the cluster-based VANET system took too much computational power to be implemented into the final product and for that reason, no communication

technology was simulated. As discussed previously, the route management was removed due to the size of the project. The evidence in table 6.1 points to the examples of the manoeuvres completed in the simulation. When viewing these examples in the appendix, the simulation follows the design correctly.

Functional requirements	Group	Result and evidence
A Vehicle architecture to control the vehicle's movements	Vehicle	Successful (see appendix 1)
Vehicles must be able to join a platoon	Platoon	Successful (see appendix 2)
Platoons must be allowed to merge	Platoon	Successful (see appendix 3)
A platoon must be able to split into	Platoon	Successful (see appendix 4)
A vehicle must be able to leave a platoon	Platoon	Successful (see appendix 5)
A vehicle must be able to form a platoon	Platoon	Successful (see appendix 6)
Multi-lane merging	Platoon	Semi-successful (see appendix 7)
Simulated communication technology	Communication	Failed
Cluster-based VANET	Communication	Failed
Route management	Traffic flow optimisation	Removed from project

Table 6.1: Results and evidence of functional requirements

7. Critical evaluation

The main three objectives I planned to complete involved creating an architecture for the vehicles, communication, and platooning. The vehicle architecture was finished according to plan, I was able to understand the logic behind the Carla module and documentation which allowed me to have control of different aspects of the vehicles such as acceleration, speed, and pathing. While developing the communication module, I discovered that the Carla simulator had drawbacks that limited the complexity of the architecture as well as the environments available to test it in. This resulted in having to exclude simulating realistic communication and accounting for the possibility of this complexity issue becoming a problem in the future. Although this setback decreased the reliability of the results if they were compared to a real-life experiment, the communication module would have a small effect on the results as vehicles only needed to communicate with others within a short distance.

I relied on getting an understanding of potential simulation software through literature and demonstrations when deciding on which one to use. This backfired as when I was already committed to using the Carla simulator, I discovered the limitations that affected the initial plan of the project. When I have a project in the future that involves simulation, I plan on testing the appropriate software and attempt to push its limits to understand what it can handle which should reduce the chances of software limitations affecting the project.

The preferred end goal of the project was to complete every objective, including the traffic flow optimisation, even though I understood that the viability of its completion was low. The problem was that when I reached the platooning objective, reading and understanding the theory behind it was fairly simple which deflated my idea of how difficult applying it in the Carla simulator would be. After a couple of months of design and development, I finished a draft of the architecture but there was a limited time frame to complete the final objective, therefore I choose to finalise the platooning architecture and remove the last objective. In hindsight, I would of choose to pick a more specific sub-topic of platooning such as multi-lane merging instead of trying to fit as many elements as possible of coordination into the project.

The benefit to completing the project was gaining an understanding of the different aspects needed to create a realistic simulation of platooning. With communication this included simulating the technology, having vehicles connected by a wireless network, as well as how to structure the vehicles in those networks for messages to be forwarded around. With platooning this included the common manoeuvres used by architectures as well as different ways they could be expanded such as implementing multi-lane merging.

The project itself has no ethical implications as it is set to show that platooning can improve congestion, but there can be ethical complications when attempting to make the simulation a reality. When AVs start communicating with each other, it opens the possibility for private information such as the vehicles' routes to be stolen when that data is sent between vehicles. This could be achieved by a man in the middle attack or by a person with malicious intentions mimicking a vehicle to have that information sent over. Additionally, in the real world, it might not be safe for vehicles in a platoon to drive close to each other. An error in the system could cause a vehicle to crash with another member

which could then create a chain reaction of crashes due to the other vehicles being extremely close. Also, this chain reaction could happen if an external vehicle crashes into a member of a platoon. Additionally, these potential scenarios and the short distance between member vehicles could cause distress for passengers of these vehicles.

The final project consists of two folders containing the architectures created and the files used to analyse the data collected. Although the architectures themselves have no real-life applications as they cannot be implemented into an actual AV, they do support the possibility of using platooning to decrease congestion. Simulation wise, the usability of the project does allow for the architecture to be expanded on to explore new ideas, whether this includes platoon route optimisation or experimenting with different platoon spacing methods.

8. Conclusion and Future work

During the dissertation, the problem of congestion increasing yearly around the world was investigated. It was discovered that the road infrastructure built could not compete with the increasing number of vehicles entering the roads each year. The solution was to coordinate vehicles, when comparing this method to the alternative of redesigning and building more infrastructure, the solution seemed like the more beneficial option.

The coordination solution was to research, design, and development a platooning architecture for the Carla simulator to determine how it would affect the capacity of roads. Additionally, the same software development cycle was completed to discover the reliability of including multi-lane platooning. The solution was successfully created with the exclusion of a communication architecture due to reasons previously stated.

The results from single-lane platooning showed that it did increase the capacity of the entire road network by about 15% when compared to autonomous vehicles roaming. Additionally, most roads in the environment experienced a higher capacity with platooning enabled. This shows that platooning is a beneficial solution to reducing congestion. The disadvantage to this solution as seen in the background research was that current autonomous vehicles are transitioning from an SAE level of 2 to 3, where the appropriate level to implement this technology in the real world would be 4 due to the risks involved. Therefore, the solution is only viable for the future of autonomous vehicles. The simulations for multi-lane platooning verified the stability of implementing advanced manoeuvres to the platooning architecture as the speeds of the vehicles did not have any rapid variations.

Many papers have their approach to coordination whether it is the method of performing a manoeuvre or the terminology used. For the future of platooning, the standardisation of the topic needs to be investigated and achieved. Providing structured methods and reliable data would improve innovation as researchers in the topic would have a base to build their ideas on instead of starting from scratch. Additionally, this would increase the chances of vehicles with platooning capabilities made by different companies to coordinate together. Other problems to be researched include platoon emergency scenarios such as if a vehicle in the middle of a platoon suddenly shuts off. Considering the short distance between the vehicles, the platoon members need a method to identify and react to those types of failures.

References

[1] S. Mariani, G. Cabri and F. Zambonelli, "Coordination of Autonomous Vehicles", *ACM Computing Surveys*, vol. 54, no. 1, pp. 1-33, 2021. Available: https://arxiv.org/pdf/2001.02443.pdf.

[2] J. Kennedy and R. Eberhart, "Particle swarm optimization", *Proceedings of ICNN'95 - International Conference on Neural Networks*, 1995. Available: http://ai.unibo.it/sites/ai.unibo.it/files/u11/pso.pdf.

[3] J. Mena-Oreja and J. Gozalvez, "On the Impact of Platooning Maneuvers on Traffic", *2018 IEEE International Conference on Vehicular Electronics and Safety (ICVES)*, 2018. Available: https://ieeexplore.ieee.org/abstract/document/8519515. [Accessed 1 May 2021].

[4] L. Zhang, F. Chen, X. Ma and X. Pan, "Fuel Economy in Truck Platooning: A Literature Overview and Directions for Future Research", *Journal of Advanced Transportation*, vol. 2020, pp. 1-10, 2020. Available: https://www.hindawi.com/journals/jat/2020/2604012/. [Accessed 1 May 2021].

[5] G. Domingues, J. Cabral, J. Mota, P. Pontes, Z. Kokkinogenis and R. Rossetti, "Traffic Simulation of Lane-Merging of Autonomous Vehicles in the Context of Platooning", *2018 IEEE International Smart Cities Conference (ISC2)*, 2018. Available: https://ieeexplore.ieee.org/abstract/document/8656856. [Accessed 2 May 2021].

[6] D. Lin, J. Kang, A. Squicciarini, Y. Wu, S. Gurung and O. Tonguz, "MoZo: A Moving Zone Based Routing Protocol Using Pure V2V Communication in VANETs", *IEEE Transactions on Mobile Computing*, vol. 16, no. 5, pp. 1357-1370, 2017. Available: https://www.xilirprojects.com/wp-content/uploads/2019/11/lin2016.pdf. [Accessed 3 May 2021].

[7] V. Milanes, J. Villagra, J. Godoy, J. Simo, J. Perez and E. Onieva, "An Intelligent V2I-Based Traffic Management System", *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 1, pp. 49-58, 2012. Available: https://hal.inria.fr/file/index/docid/732884/filename/An_Intelligent_V2I-Based_Trafi_c_Management_System.pdf. [Accessed 3 May 2021].

[8] L. Wang and X. Liang, "Phantom Traffic Jam Based on MATLAB", *Journal of Physics: Conference Series*, vol. 1852, no. 4, p. 042036, 2021. Available: https://iopscience.iop.org/article/10.1088/1742-6596/1852/4/042036/pdf. [Accessed 7 May 2021].

[9] A. Dosovitskiy, G. Ros, F. Codevilla, A. López and V. Koltun, "CARLA: An Open Urban Driving Simulator", *Conference on Robot Learning*, vol. 78, pp. 1-16, 2017. Available: http://proceedings.mlr.press/v78/dosovitskiy17a/dosovitskiy17a.pdf. [Accessed 23 May 2021].

[10] S. Jayasooriya and Y. Bandara, "Measuring the Economic costs of traffic congestion", *2017 Moratuwa Engineering Research Conference (MERCon)*, 2017. Available: https://www.researchgate.net/profile/Yapa-Bandara-

2/publication/318473826_Measuring_the_Economic_costs_of_traffic_congestion/links/5c59aed745851582c3cff6f6/Measuring-the-Economic-costs-of-traffic-congestion.pdf. [Accessed 24 May 2021].

[11] S. Bharadwaj, S. Ballare, Rohit and M. Chandel, "Impact of congestion on greenhouse gas emissions for road transport in Mumbai metropolitan region", *Transportation Research Procedia*, vol. 25, pp. 3538-3551, 2017. Available: https://www.sciencedirect.com/science/article/pii/S2352146517305896. [Accessed 25 May 2021].

[12] J. Kinigadner, F. Wenner, M. Bentlage, S. Klug, G. Wulfhorst and A. Thierstein, "Future Perspectives for the Munich Metropolitan Region – an Integrated Mobility Approach", *Transportation Research Procedia*, vol. 19, pp. 94-108, 2016. Available: https://www.sciencedirect.com/science/article/pii/S2352146516308572. [Accessed 25 May 2021].

[13] "Traffic Index", *Tomtom.com*, 2021. [Online]. Available: https://www.tomtom.com/en_gb/traffic-index/ranking/. [Accessed: 25- May- 2021].

[14] J. Du, H. Rakha, F. Filali and H. Eldardiry, "COVID-19 pandemic impacts on traffic system delay, fuel consumption and emissions", *International Journal of Transportation Science and Technology*, 2020. Available: https://www.sciencedirect.com/science/article/pii/S2046043020300769. [Accessed 25 May 2021].

[15] D. Hennessy and D. Wiesenthal, "Traffic congestion, driver stress, and driver aggression", *Aggressive Behavior*, vol. 25, no. 6, pp. 409-423, 1999. Available: https://www.researchgate.net/profile/Dwight-Hennessy/publication/229863510_Traffic_congestion_driver_Stress_and_driver_aggression/links/5ad34582a6fdcc29357eddf5/Traffic-congestion-driver-Stress-and-driver-aggression.pdf. [Accessed 25 May 2021].

[16] K. Zhu, "China's Great Wall of Traffic Jam: 11 Days, 74.5 Miles", *ABC News*, 2010. [Online]. Available: https://abcnews.go.com/International/chinas-traffic-jam-lasts-11-days-reaches-74/story?id=11550037. [Accessed: 26- May- 2021].

[17] "Using CARLA", *CARLA simulator*, 2021. [Online]. Available: https://forum.carla.org/c/using-carla. [Accessed: 29- May- 2021].

[18] *RoadRunner*. MathWorks.

[19] A. Afzal, D. Katz, C. Goues and C. Timperley, "A Study on the Challenges of Using Robotics Simulators for Testing", 2021.

[20] Q. Li, Z. Peng, Q. Zhang, C. Liu and B. Zhou, "Improving the Generalization of End-to-End Driving through Procedural Generation", 2021.

[21] K. Hamad and S. Kikuchi, "Developing a Measure of Traffic Congestion: Fuzzy Inference Approach", *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1802, no. 1, pp. 77-85, 2002. Available:

https://www.researchgate.net/publication/234940318_Developing_a_Measure_of_Traffic_Congestion_A_Fuzzy_Inference_Approach. [Accessed 2 June 2021].

[22] A. Mohan Rao and K. Ramachandra Rao, "MEASURING URBAN TRAFFIC CONGESTION – A REVIEW", *International Journal for Traffic and Transport Engineering*, vol. 2, no. 4, pp. 286-305, 2012. Available: https://www.researchgate.net/publication/271344150_Measuring_Urban_Traffic_Congestion_-_A_Review. [Accessed 1 June 2021].

[23] Z. Mir and F. Filali, "Simulation and Performance Evaluation of Vehicle-to-Vehicle (V2V) Propagation Model in Urban Environment", *2016 7th International Conference on Intelligent Systems, Modelling and Simulation (ISMS)*, 2016. Available: https://ieeexplore.ieee.org/document/7877245. [Accessed 16 June 2021].

[24] A. Adebisi, "A review of the difference among macroscopic, microscopic and mesoscopic traffic models", 2017. Available: https://www.researchgate.net/publication/321624654_A_REVIEW_OF_THE_DIFFERENCE_AMONG_MACROSCOPIC_MICROSCOPIC_AND_MESOSCOPIC_TRAFFIC_MODELS. [Accessed 8 July 2021].

[25] K. Ramamohanarao et al., "SMARTS", *ACM Transactions on Intelligent Systems and Technology*, vol. 8, no. 2, pp. 1-22, 2017. Available: https://dl.acm.org/doi/10.1145/2898363. [Accessed 8 July 2021].

[26] C. Cooper, D. Franklin, M. Ros, F. Safaei and M. Abolhasan, "A Comparative Survey of VANET Clustering Techniques", *IEEE Communications Surveys & Tutorials*, vol. 19, no. 1, pp. 657-681, 2017. Available: https://ieeexplore.ieee.org/abstract/document/7572026. [Accessed 11 July 2021].

[27] W. Liang, Z. Li, H. Zhang, S. Wang and R. Bie, "Vehicular Ad Hoc Networks: Architectures, Research Issues, Methodologies, Challenges, and Trends", *International Journal of Distributed Sensor Networks*, vol. 11, no. 8, p. 745303, 2015. Available: https://journals.sagepub.com/doi/10.1155/2015/745303. [Accessed 11 July 2021].

[28] J. Heinovski and F. Dressler, "Platoon Formation: Optimized Car to Platoon Assignment Strategies and Protocols", *2018 IEEE Vehicular Networking Conference (VNC)*, 2018. Available: https://ieeexplore.ieee.org/document/8628396. [Accessed 14 July 2021].

[29] X. Xiong, J. Sha and L. Jin, "Optimizing Coordinated Vehicle Platooning: An Analytical Approach Based on Stochastic Dynamic Programming", 2020.

[30] S. Jeong, Y. Baek and S. Son, "Distributed Urban Platooning towards High Flexibility, Adaptability, and Stability", *Sensors*, vol. 21, no. 8, p. 2684, 2021. Available: https://www.mdpi.com/1424-8220/21/8/2684. [Accessed 15 July 2021].

[31] C. Hidalgo, R. Lattarulo, C. Flores and J. Pérez Rastelli, "Platoon Merging Approach Based on Hybrid Trajectory Planning and CACC Strategies", *Sensors*, vol. 21, no. 8, p. 2626, 2021. Available: https://www.mdpi.com/1424-8220/21/8/2626. [Accessed 21 July 2021].

[32] J. Zhou and F. Zhu, "Analytical analysis of the effect of maximum platoon size of connected and automated vehicles", *Transportation Research Part C: Emerging Technologies*, vol. 122, p. 102882, 2021. Available: https://www.sciencedirect.com/science/article/pii/S0968090X20307828. [Accessed 1 August 2021].

[33] R. Firoozi, X. Zhang and F. Borrelli, "Formation and reconfiguration of tight multi-lane platoons", *Control Engineering Practice*, vol. 108, p. 104714, 2021. Available: https://doi.org/10.1016/j.conengprac.2020.104714 [Accessed 2 August 2021].

[34] M. Abdel-Aty et al., "Using Smartphone as On-board unit (OBU) Emulator Implementation Study", University of Central Florida, Orlando, 2020 [Accessed 18 September 2021].

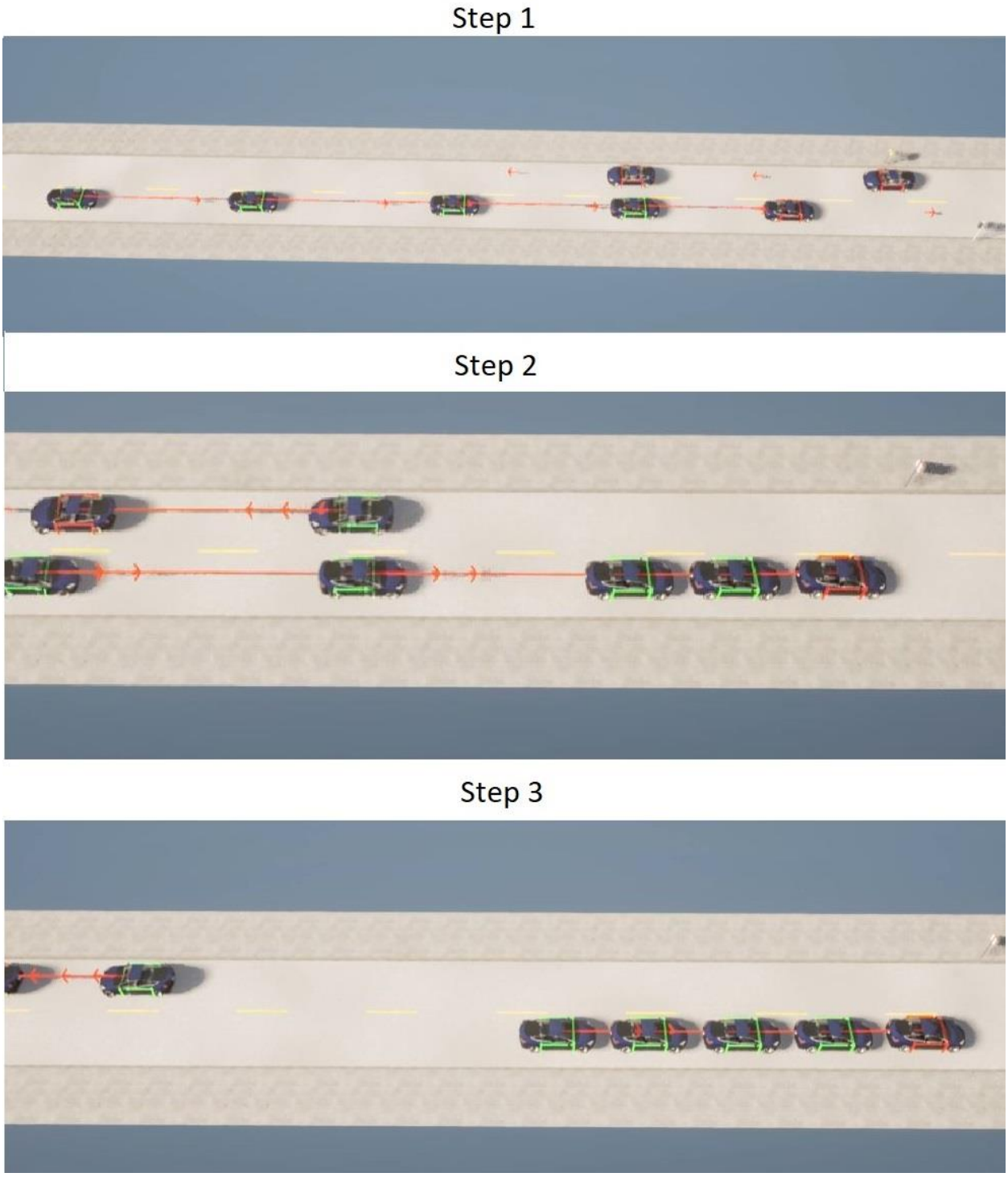
[35] SAE International, "Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles", 2021.

[36] E. Sugiura, "Honda launches world's first level 3 self-driving car", *Nikkei Asia*, 2021. [Online]. Available: https://asia.nikkei.com/Business/Automobiles/Honda-launches-world-s-first-level-3-self-driving-car. [Accessed: 28- Sep- 2021].

Appendix

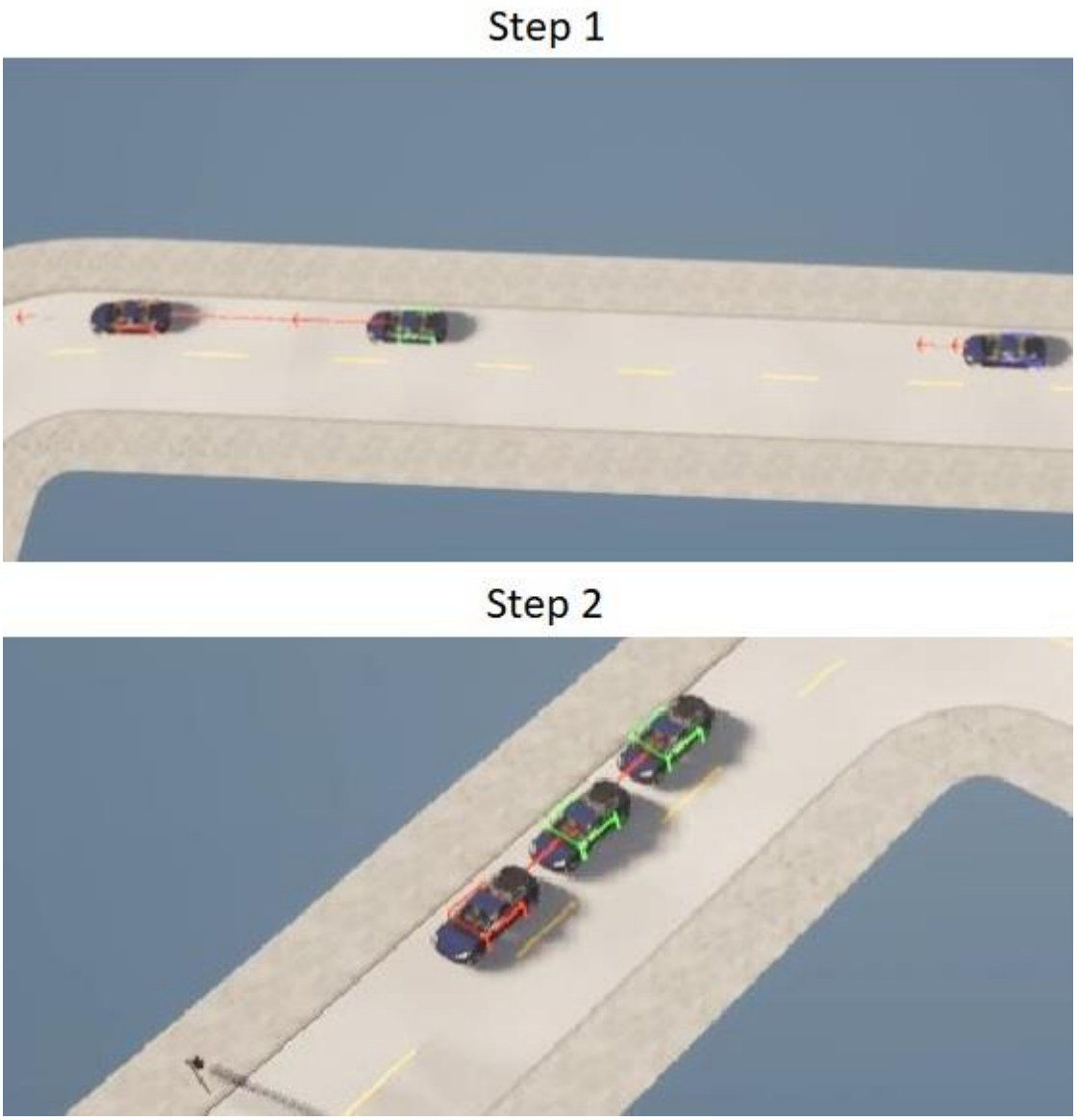
Testing

In appendix 1, the vehicle class is demonstrated. In each step, the platoon leader uses the vehicle class to control the speed of the vehicles.



Appendix 1: Speed of vehicles controlled

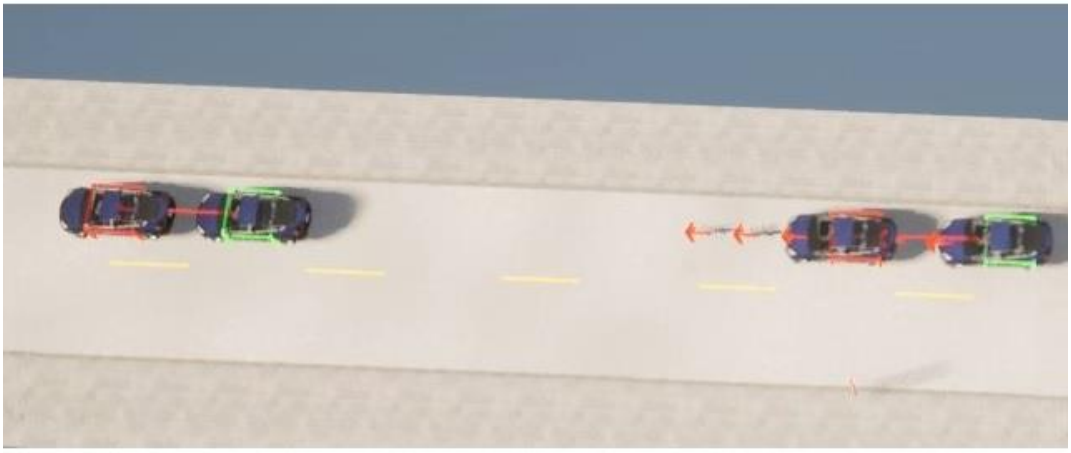
In appendix 2, the join manoeuvre is demonstrated, any vehicle preforming the manoeuvre will have a blue box surrounding as seen in step 1. In step 2, when the vehicle gets close enough, it becomes a part of the platoon.



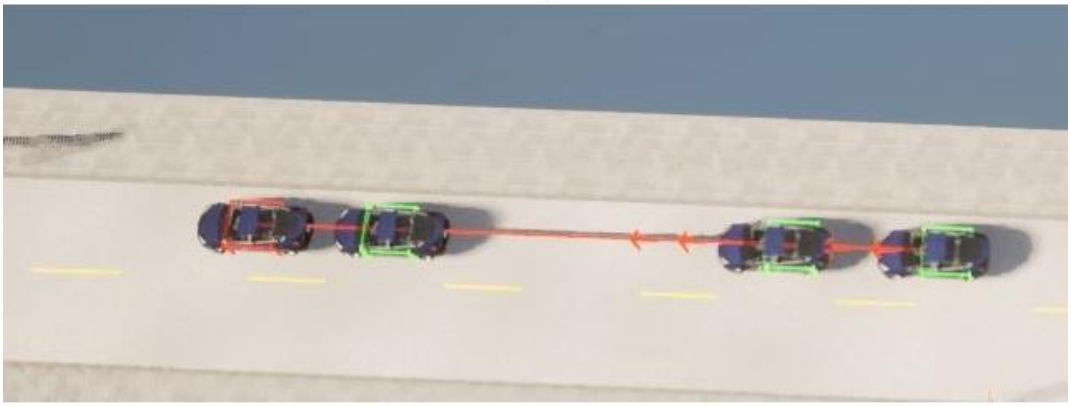
Appendix 2: Join manoeuvre

In appendix 3, the merge manoeuvre is demonstrated. In step 1, the platoon furthest from the end of the road notices that there is another platoon in front of it and speeds up. In step 2, the platoons are close enough and merge into 1. In the step 3, the close the gap to get as close as possible.

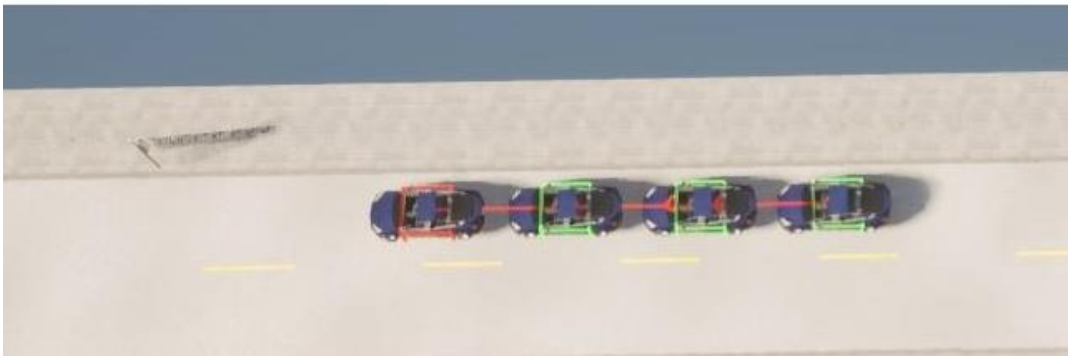
Step 1



Step 2

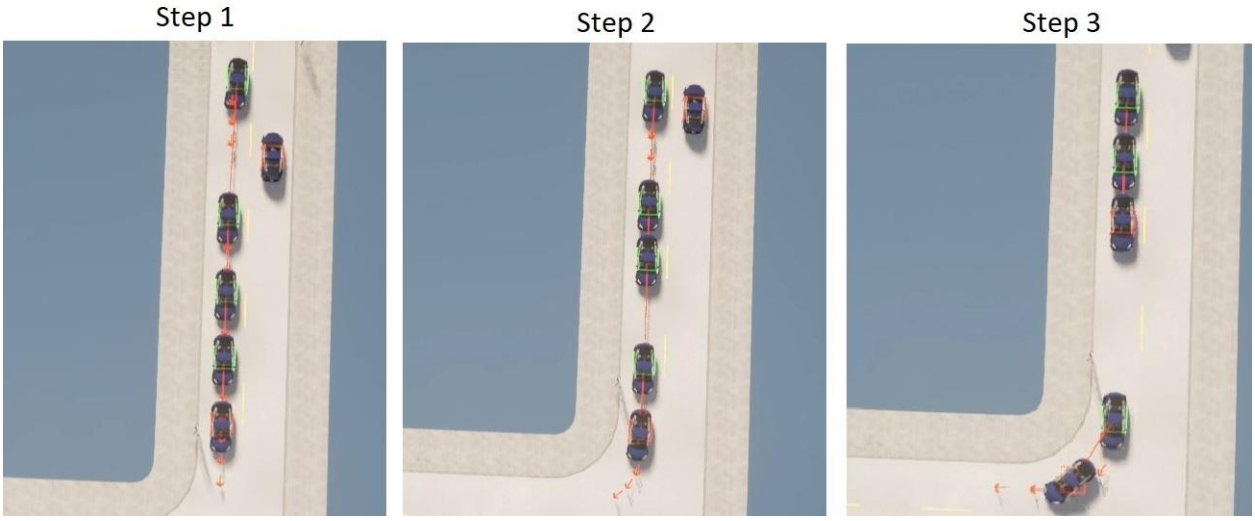


Step 3



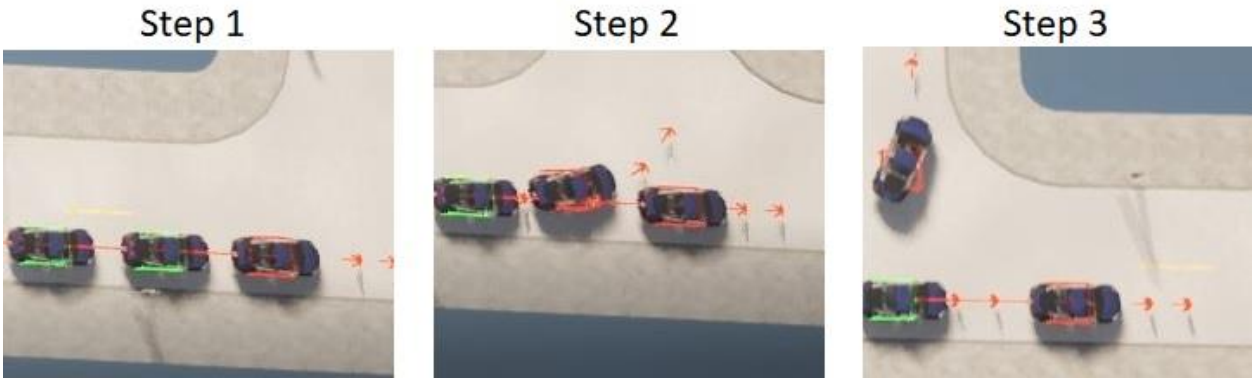
Appendix 3: Merge manoeuvre

In appendix 4, the split manoeuvre is demonstrated. In step 1, the platoon drives past a traffic light. In step 2, the traffic light turns to red so some of the platoon vehicles stop. In step 3, the captain notices this and splits the platoon into two groups.



Appendix 4: Split manoeuvre

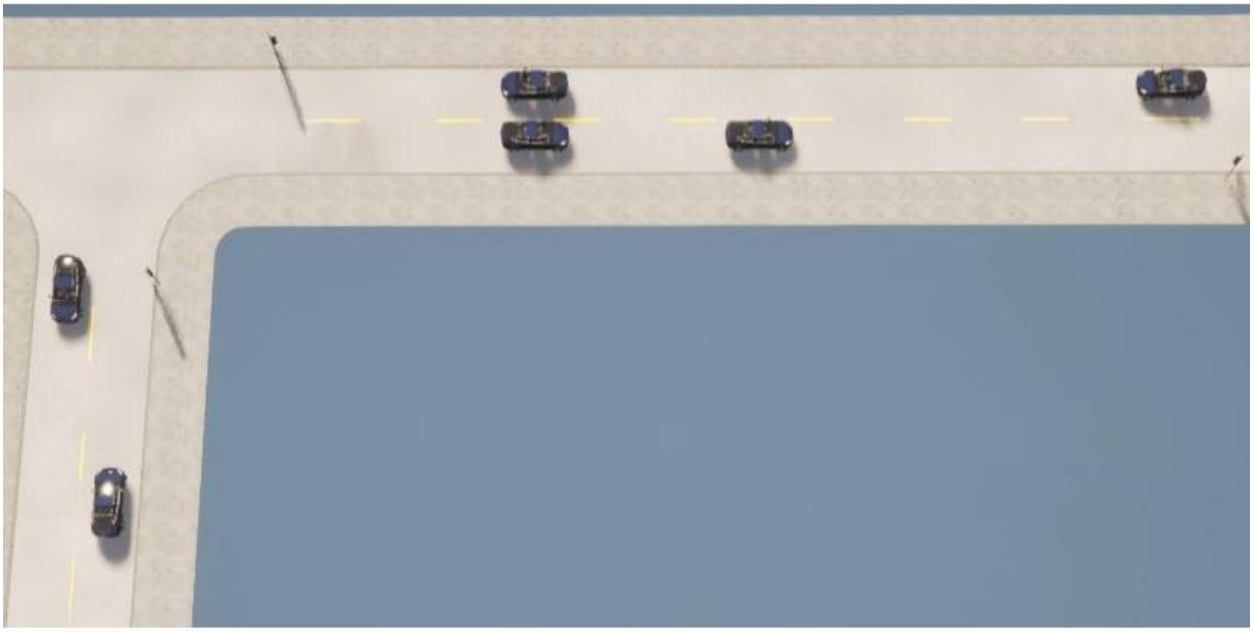
In Appendix 5, the leave manoeuvre is demonstrated. In step 1, the platoon is reaching the point where the middle member will leave. In step 2, the member is at the leaving point, so the captain removes the member from the platoon. In step 3, the member that left cannot find a new platoon to join so it creates its own, and the gap the member left in the previous platoon starts to close.



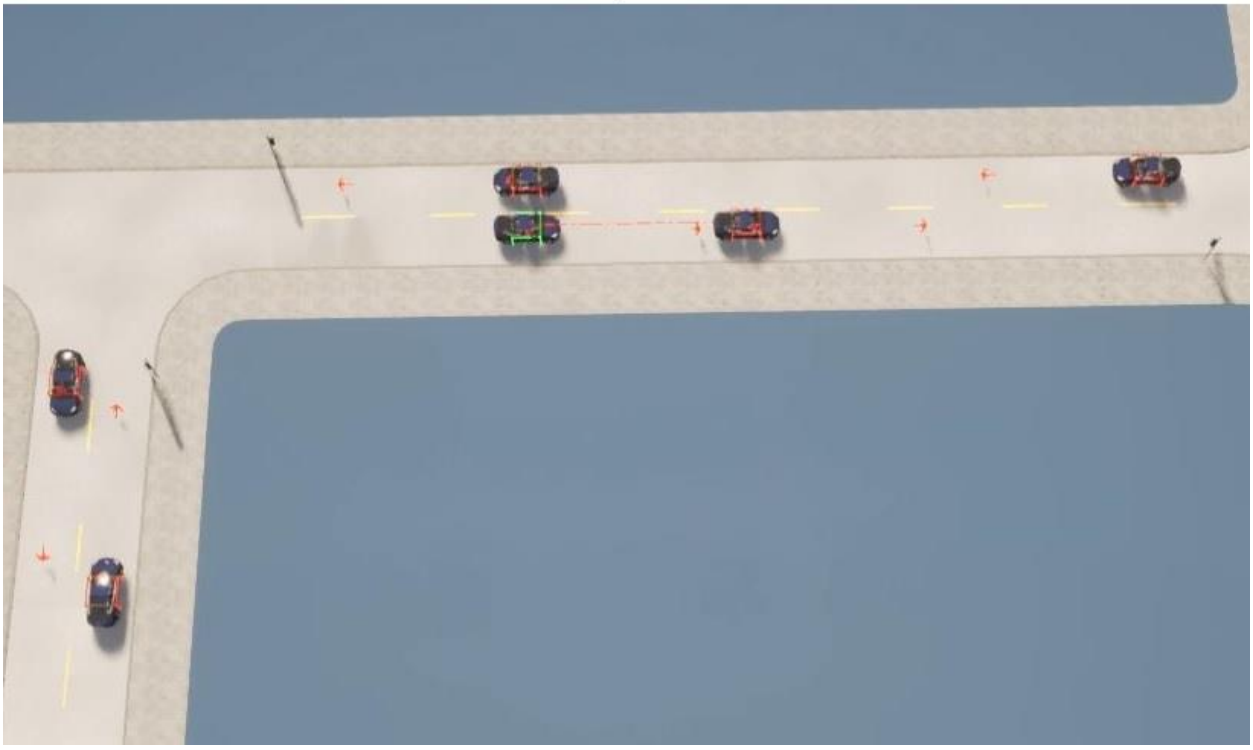
Appendix 5: Leave manoeuvre

In Appendix 6, the form manoeuvre is shown. In step 1, the vehicles are spawned in. In step 2, most vehicles do not have a platoon to join so they form their own.

Step 1



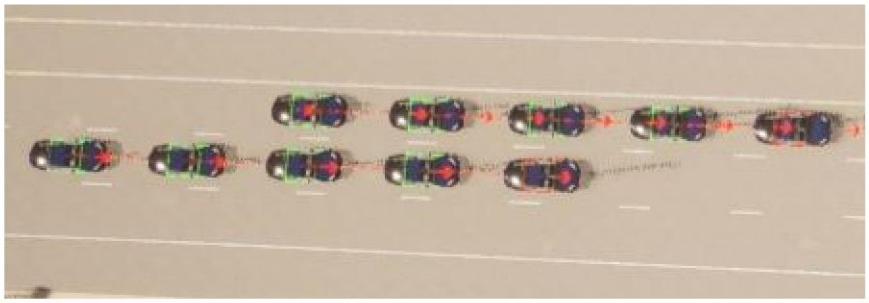
Step 2



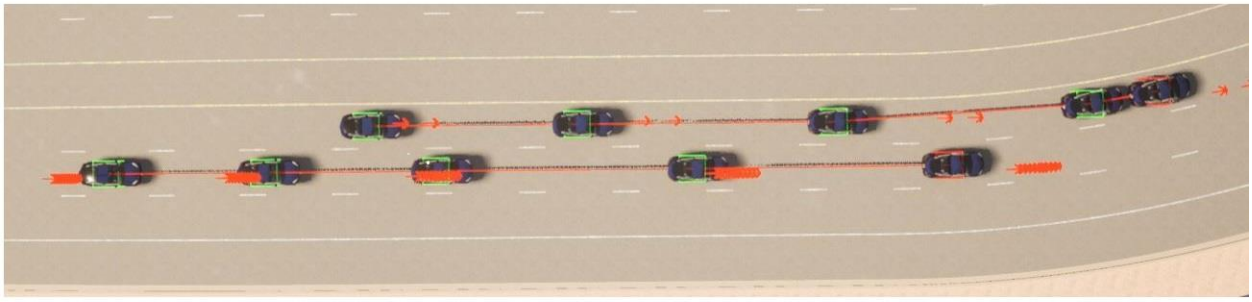
Appendix 6: Form manoeuvre

In Appendix 7, the multi-lane merge is shown. In step 1, two platoons are spawned next to each other with the platoon in the middle lane wanting to merge with the platoon on the inner lane. In step 2, the vehicles in both lanes create gaps. In step 3, the platoon in the middle lane uses the gaps to insert its vehicles in the inner lane. In step 4, the vehicles have merged into one platoon and are instructed to drive closely to each other.

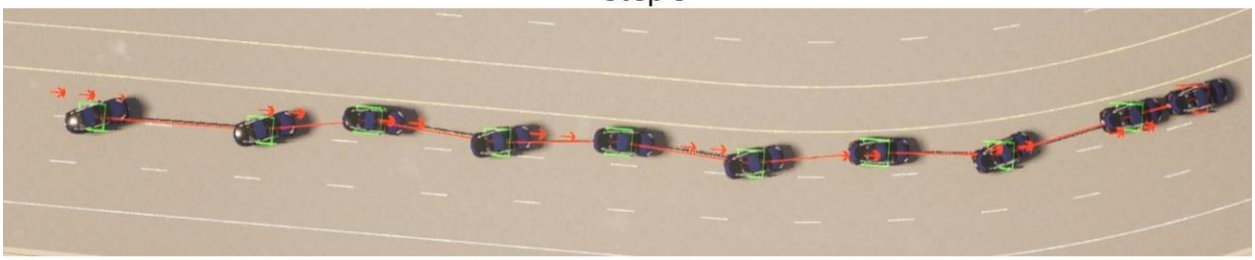
Step 1



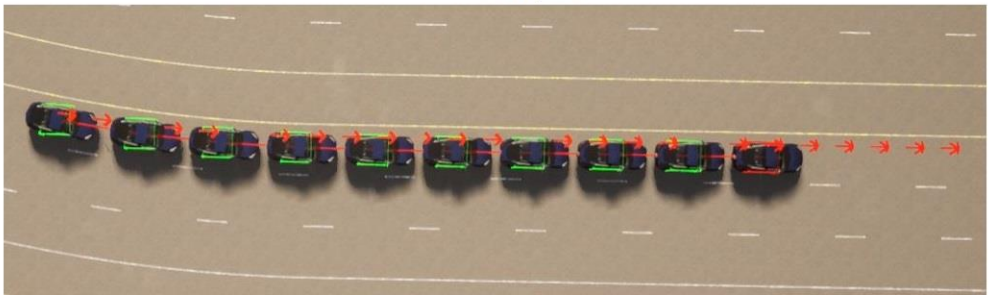
Step 2



Step 3



Step 4



Appendix 7: Multi-lane merge manoeuvre