Infrastructure Assisted Automation of Lane Change Manoeuvre for Connected and Autonomous Vehicles

Alexandre Lissac, Soufiene Djahel and Jack Hodgkiss
Department of Computing and Mathematics, Manchester Metropolitan University, UK
{ALEXANDRE.LISSAC@stu.mmu.ac.uk, s.djahel@mmu.ac.uk and jack.hodgkiss@stu.mmu.ac.uk}

***Abstract*—This paper proposes an innovative way of performing efficient lane changes in road traffic composed of autonomous vehicles. In a world where more and more objects are connected to the Internet and are involved in the constant flow of data, it seems logical that automobile traffic made of inter-connected vehicles will soon arise. To be widely used by the public, Connected and Autonomous Vehicles (CAVs) need to safely and efficiently perform lane change manoeuvre which is one of the most dangerous manoeuvres on the road. This paper proposes an original lane change protocol inspired by the way mutual exclusion works in an Operating System. The protocol benefits from the smart road infrastructure support to efficiently track open spaces between moving vehicles and prepare the most suitable space for a vehicle aiming to change its current lane. The evaluation results highlight the efficiency of our protocol and its potential to make journeys by vehicle enjoyable for all occupants.**

***Keywords* – ITS, Smart Cities, Connected and Autonomous Vehicles (CAVs), Lane change, Road Safety.**

I. Introduction

Efficient and green transportation is a key enabler of smart cities which will revolutionise our lives in the near future. Indeed, achieving faster and smarter mobility of people and goods in and around cities depends on the sophistication level of traffic congestion mitigation techniques as well as the advancement of the deployed road infrastructure [1]. Nowadays, congested traffic is the root cause of a number of problems such as air pollution and the excessive use of fossil fuel. In [2], the authors state that "the fuel consumed by vehicles stopping and idling accounts for approximately 40% of network wide vehicular fuel consumption". Their research work has solved this issue by developing an intelligent system that would better handle the operations of the traffic lights to avoid idle vehicles on the roads. This is achieved using a series of sensors deployed around traffic lights to sense and report the presence or absence of cars waiting to go through. This data would then be processed and used to enhance the fluidity of the traffic at the crossing. As this paper was written in 2007, it misses out on an important feature of current and future road traffic, i.e., autonomous vehicles. This breakthrough technology could lead to a more efficient intelligent traffic system capable of interacting with the cars at any point on the road.

To create an Intelligent Traffic System, all its entities are required to be intelligent themselves. Therefore, vehicles are required to become more than just mechanical machines moving from point A to point B and piloted by human drivers. Instead, it is required that they acquire data from their environment, understand it and share it with the system. Thus, data acquired from all vehicles of a system would lead to a better understanding of how traffic flow evolves. To easily acquire such data, vehicles' features need to be upgraded to equip them with the required sensing and transmission technology. A distributed architecture for autonomous vehicles was proposed in [3] and essential features that the vehicles must incorporate, such as vehicle control, planning, localisation, system management and perception, were highlighted. These features require specific enabling technologies; *perception* can be achieved with a combination of radar, cameras and computer vision technology, while the use of GPS is needed to enable the *Localisation* function. Finally, both the system management and planning functions need adequate computational power to fulfil their task. The planning function interprets data from the *perception* function and gives orders to the *vehicle control* system. On the other hand, the *system management* function handles the overall operations of autonomous vehicles. Since it is a basic design, other autonomous vehicles' architecture might differ from this one. For instance, other architectures add a *perception* system at the back of the vehicle for better understanding of the development.

Although autonomous vehicles are on their way to become part of our everyday traffic (Google launching Waymo, a company of self-driving taxis[1]), most of them are still in the prototyping stage. Therefore, testing them in normal traffic could be quite challenging, both technically and ethically. Thus, it is interesting to analyse the closest type of vehicle to the autonomous vehicle: the self-driving or smart car. Though they are not fully autonomous, cars such as the Tesla Model S or the Volvo XC90 could be autonomous for a large amount of time. In his paper, published in 2017, Barry White analyses 10.5 hours of footage showing how these vehicles were behaving with other human drivers in normal traffic [4]. One of the main flaws that White noticed is the difficulty that self-driving cars face to understand the social rules of driving. In one of his examples, he explains how a Tesla car, wanting to change its current lane, was not able to understand the politeness of surrounding drivers. To avoid such issues in

[1]https://waymo.com/

a fully autonomous traffic system, it is compulsory that each entity is equipped with a common set of lane changing rules that would mirror the existing one amongst current human drivers. To this end, several attempts have been made in recent years to automate the lane change manoeuvre by relying on connected vehicles technology such as the works presented in [5], [6], [7] and [8]. These works, however, did not exploit the smart road infrastructure capabilities to support the lane change manoeuvre, therefore we will develop in this paper a novel road infrastructure assisted coordination protocol that automates the lane changing manoeuvre in a fully autonomous environment.

The remainder of this paper is organised as follows. Section II gives an overview on the required logistics for our algorithm followed by a brief overview on lane change manoeuvre in Section III. In Section IV, we present a detailed explanation of the operational steps of our proposed algorithm. In Section V, we present and analyse the obtained performance evaluation results. Finally, in Section VI, we conclude the paper and present our future work plan.

II. Logistics underpinning the design of our algorithm

This section presents the logistics needed to support our proposed algorithm.

A. Hardware needed

To successfully apply our algorithm in a real world scenario, several hardware equipment are required to ensure the connectivity among the vehicles as well as the exchange of information between the vehicles and the smart road infrastructure in place. First, each vehicle in the traffic flow needs to send and receive periodic beacons and data packets following IEEE 802.11p protocol rules [9], using IEEE 802.11p transceivers. This increases their awareness of the traffic in their surrounding and enables better coordination among nearby vehicles. This protocol transforms vehicles into communicating nodes, receiving and emitting information through short-range transmissions regardless of their velocity. Secondly, Road Side Units (RSUs) are required to be sparsely spread alongside the road network to receive and emit the needed information. These RSUs are needed to relay information between the Local Traffic Controller (LTC) and the vehicles. On one hand, the communication between the RSUs and the vehicles will be via IEEE 802.11p wireless protocol. On the other hand, the communication between the RSUs and the LTC is performed via Optic Fiber Cables. These Optic Fiber Cables will allow a stable flow of data to navigate between the LTC and the RSUs. Finally, the LTC could be any computing device capable of performing efficient computational tasks, within the target application time constraints, and powerful enough to efficiently run our algorithm. Additionally, the LTC could be linked to a server that would store traffic data about the specific segment of the road that this LTC is managing.

Notation	Meaning
OS	Open Space
OSL	Open Space Length
SUP	Space Under Preparation
BC	Back Car
FC	Front Car
LCC	Lane Changing Car
LS	Locked Space
SGD	Safety Gap Distance

Table I: Notations summary

B. Data flow explanation

Figure 1 depicts how data should flow in our system where vehicles transmit wireless IEEE 802.11p packets to the RSUs that relay the information to the LTC. The LTC will process this data and send back its recommendations to the vehicles via the road transceivers. This figure shows the overall setup of the logistics needed to transpose our system in a real world scenario. Wavy arrows indicate a 2-way transmission of wireless IEEE 802.11p packets between the vehicles and the RSUs while the solid arrow lines indicate a 2-way transmission between the LTC and the RSUs using Optic Fiber Cables.

III. Lane change mechanism

This section describes the main steps for performing a lance change manoeuvre and outlines the main requirements to safely perform it.

A. Overview

The lane change manoeuvre can be split into the following steps as depicted in Figures 2 - 6.

B. Safe lane change requirements

In order to assure the highest safety level, all vehicles involved in a lane change manoeuvre must follow a set of rules. First, the vehicles must follow the driving rules of that specific road, such as not exceeding the speed limit when performing lane change, and alerting the LTC of any intention of changing lane. Secondly, the system must assure that the vehicle desiring to change lane is moving into a sufficient space (i.e., the identified Open Space (OS) should be large enough to welcome the Lane Changing Car (LCC) that guarantees the inter-vehicles safety distance. To achieve this, once the best open space is picked, the system must assure that, after preparation, there is still enough room for the LCC to move into the space. After locking the identified space, the system must also assure that there is still enough room for the car to move into. Finally, the system must ensure that the car is moving into a "safe spot" of the open space, not too close to either of the two vehicles delimiting the OS.

IV. Our proposed algorithm

In this section we will present our proposed algorithm's key principle and its different operational steps. The notations to be used throughout the remainder of this paper are explained in Table I.

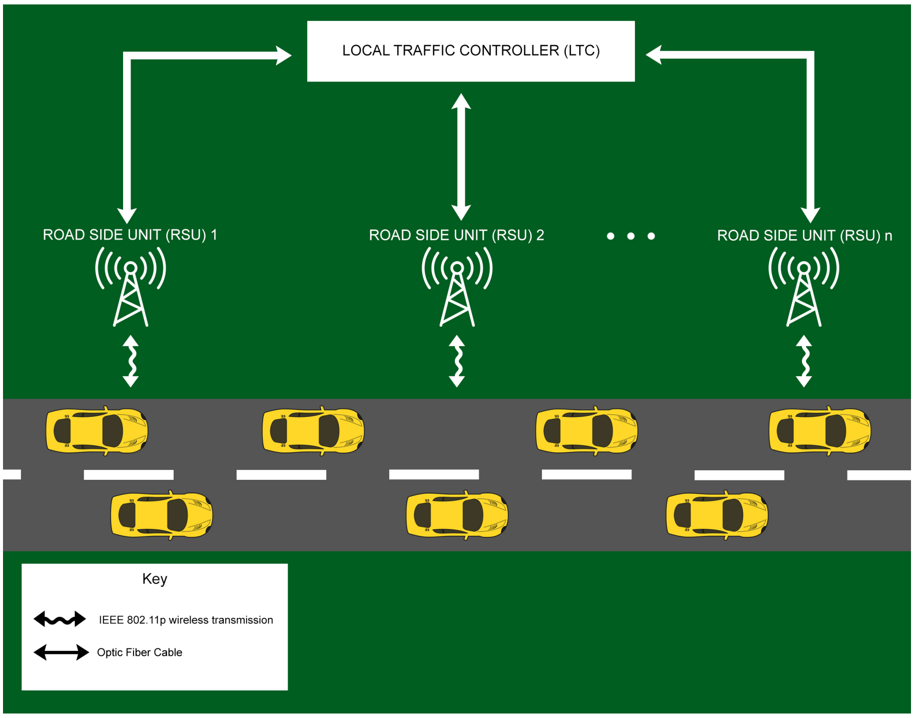

Figure 1: Illustration of the required infrastructure for implementing our algorithm

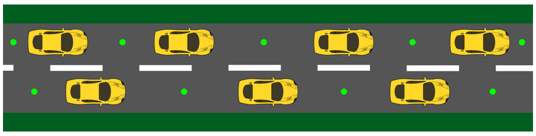

Figure 2: Step 0: Keeping track of open spaces

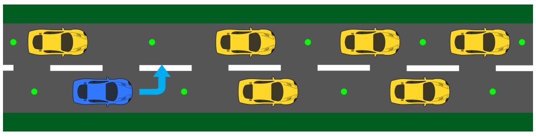

Figure 3: Step 1: Sending a request to change lane (by the blue vehicle in this example)

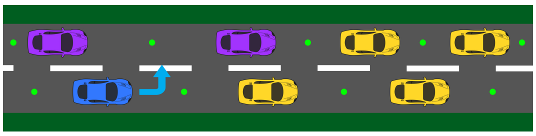

Figure 4: Step 2: Finding the best open space for the requester vehicle (i.e., the blue vehicle)

A. Key principle

This algorithm is built upon the principle of identifying and locking open spaces in the traffic and then allowing a designated vehicle to move towards it. In order to achieve this, our algorithm must undertake a series of sub-tasks as explained below.

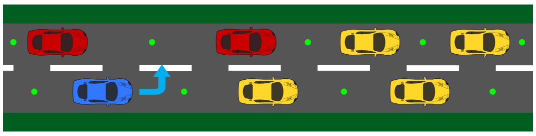

Figure 5: Step 3: Locking the speed of both red cars delimiting the best identified open space

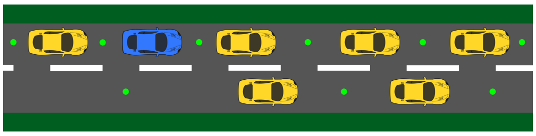

Figure 6: Step 4: Performing lane change manoeuvre

First, our algorithm must constantly detect open spaces and acknowledge them as being unique entities in the traffic system. Secondly, once a vehicle "A" informs the LTC about its desire to change its current lane, our algorithm must find the most suitable space for this vehicle to insert itself into. Thirdly, once that space is found, both vehicles "B" and "C" delimiting that space must synchronise their speed to prevent any alteration to its length until the vehicle "A" has moved safely. Finally, our algorithm must check if the vehicle "A" can safely insert itself into the space, without any risk of collision.

B. Main steps

The main operational steps of our algorithm are described below.

1) Detecting open spaces

As shown in Figure 2, the algorithm's perpetual work is the detection of the available Open Spaces (OS). For the sake of accuracy, the algorithm must detect these open spaces lane by lane and, thus, it needs to learn the list of the current vehicles on each lane and their corresponding positions. The length of the open spaces (OSL) is calculated using the following formula:

$$OSL = |x_{Vehicle\,(n)} - x_{Vehicle\,(n+1)}| \tag{1}$$

Where n and $n+1$ denote the vehicles delimiting the open space and the position of the open space's Middle Point (MP) is calculated as follows:

$$MP_{Position} = x_{Vehicle\,(n)} + / - (\frac{OSL}{2}) \tag{2}$$

In most cases the "$-$" sign will be used as the open space will be located behind $Vehicle(n)$ (i.e., the front vehicle). The special case arises when detecting the open space ahead of the leading vehicle of the lane, in this case the "$+$" sign will be used.

Finally, the algorithm must uniquely identify each OS on each lane. To perform such task, a Merkle Tree will be used to identify each OS according to the ID of the back and front vehicles. A Merkle Tree, or Hash Tree, is a technique allowing a data packet to be uniquely named according to its content. This technique is precious for our work since an open space x is unique on the traffic because it is the only space between vehicle y and vehicle z. Therefore, by putting the ID of vehicles y and z at the bottom of our Merkle Tree, it is possible to attribute a unique ID to the Open Space x. In this work we use SHA-256 hashing function as it is the most common hashing function used for a Merkle Tree. Figure 7 shows the actions performed by the algorithm to uniquely identify each open space.

2) Finding the best open space

As shown in Figure 3, when a vehicle desires to change its current lane, it sends a request to the LTC. Once this request is received, the algorithm analyses the open spaces on the concerned lane and identifies the most suitable OS for the vehicle to move into. To identify this ideal space, the algorithm will apply a series of tests on each of the relevant open spaces to judge their suitability to accommodate the LCC vehicle request.

The first test consists in checking if the open space is not too far from the LCC. This allows us to avoid scenarios where the best identified open space found is at the other end of the lane, thus making it impossible or too long for the lane changing vehicle to reach it, especially in highway scenarios. The second test aims to detect if any of the two vehicles delimiting the identified OS is already "locked", which would mean that one of them is involved in another lane change manoeuvre. In our system, the space ahead and behind of a locked space cannot be used by another vehicle to move into as it would interfere with the "locking" of an already locked space, meaning that delaying or endangering another lane change manoeuvre. The third test needed is to find out if the vehicle desiring to change lane can actually reach the identified OS. To analyse that, the algorithm must find out if the open space is ahead of or behind the vehicle as well as the open space's velocity. If the open space is going faster than the vehicle and is ahead of it, it would be impossible for the vehicle to reach the space. Below is the formula to compute the open space's velocity.

$$OS_{Velocity} = \frac{Back\ Car_{velocity} + Front\ Car_{velocity}}{2} \tag{3}$$

The fourth test consists in detecting if the distance from the vehicle to the identified OS is the shortest distance the algorithm has found so far. If the space turns out to be the closest space analysed yet, its data would be saved and its distance to the LCC will be used as reference for analysing the following spaces. Finally, the fifth test will check if the identified OS is sufficient to welcome the LCC. To decide so, the algorithm needs to know the current length of the identified OS as well as calculate the safety gap of each of the vehicles. A space might be, at first glance, long enough to welcome the LCC. However, it might be the case that, once the LCC has moved in, the safety gab distance between vehicles will not be respected. Therefore, the algorithm must calculate the safety gap in relation to both vehicles delimiting the identified OS and subtract it from the current length of the open space. If the remaining distance is larger than the length of the LCC then this latter can move into the space. In our work, the stopping distance is calculated using the AASHTO (American Association of State Highway and Transportation Officials) formula [10], shown in Eq. 4 and it is assumed that the reaction time is null as we are dealing with autonomous vehicles. However, an OS without sufficient space to welcome the LCC vehicle can still be considered If the open space length is growing, meaning that the front vehicle is travelling faster than the back vehicle.

Our interpretation of the AASHTO stopping formula [10] for autonomous vehicles is the following:

$$SafetyGapDistance = \frac{Vehicle^2_{velocity}}{(254 * (f + G))} \tag{4}$$

Where f is the coefficient of friction and will always be 0.7 because we are constantly dealing with dry roads. G represents the slope of the road and will always be 0 as we are constantly dealing with flat roads.

The mathematical expression calculating the landing distance is the following:

$$Landing_{Dist} = OSL - (SGD_{BackCar} + SGD_{FrontCar}) \tag{5}$$

Finally, once all the open spaces have been analysed and the ideal space has been found, the best identified open space is

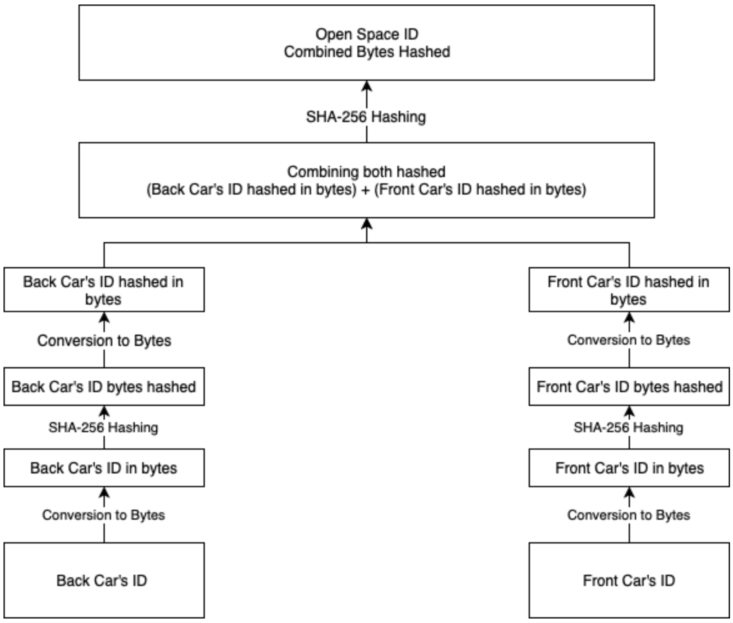

Figure 7: Uniquely identifying each open space on road traffic using a Merkle Tree

moved to the preparation stage. In cases where no suitable OS has been found for a given LCC, this latter will be put on a waiting list and another attempt will be made later. The process of identifying the best OS is summarised in Figure 8.

3) Preparing the identified best open space

As shown in Figure 9, once the system has found the most suitable open space for a given LCC to insert itself into, it will add it to the list of spaces that must be prepared. This step is important as it will condition the space to safely welcome the LCC.

The first step of preparing the space is to detect if this space must grow or not. As previously explained in section IV-B2 regarding the identification of the most suitable open space, there are some cases where the identified OS is not sufficient to welcome the LCC yet. In this case, during the preparation step, the algorithm must decrease the speed of the open space's back vehicle and increases the speed of the open space's front vehicle. Thus, increasing the distance between these two vehicles and therefore making more room for the LCC. As long as there is not enough room for the lane changing vehicle to insert itself into, the open space will stay in the preparing list and carry on growing. If the space does not need to grow, the two vehicles delimiting the Space Under Preparation (SUP) will alter their speed to reach a target speed needed for speed synchronisation. The target speed is the average speed of the two vehicles, thus allowing both vehicles to alter their speed without tremendously changing their original speed. As long as both vehicles have not reached the target speed, the open space will stay in the preparing list. Once both vehicles have reached the target speed, a final check will be done to verify if there is still enough room to welcome the LCC. If the space does not have enough room anymore to welcome the LCC, it will be removed from the preparing list and a new space will be searched for. Otherwise, if the space has enough room to welcome the LCC, the SUP will be ready to be locked.

4) Locking an identified OS

As shown in Figure 5, once a space preparation is ended it can then be locked to welcome the lane changing vehicle. While the space is locked the algorithm must perform certain tasks to assure that the lane change manoeuvre can be performed safely. First, the system must always check if the locking of the space is not being overruled by some unforeseen incidents such as an emergency break. The algorithm must therefore check if the locked space is sufficient to welcome the LCC. If not, the lock is cancelled and the system must find another suitable open space for the LCC. Finally, the system must always ascertain that the two vehicles delimiting the locked space are moving at the same speed. This is achieved by adapting the speed of the back car to the speed of the front car so that if the latter has to slightly reduce its speed due to unforeseen circumstances, the former will slow down at the same time and therefore preserves the length of the locked space.

5) Safety assessment and performing lane change

As shown in Figure 6, once an open space is locked, the system attempts to move the lane changing vehicle, linked to this open space, into it. To do so, the algorithm must follow a set of instructions to assure the safe lane change of the lane changing vehicle. First, the system must assure that the locked space is large enough to welcome the LCC. If this is not the case, the locked space is unlocked and the system must find a new open space for the LCC to insert itself into. Secondly, the system must verify if the LCC is close enough to the locked

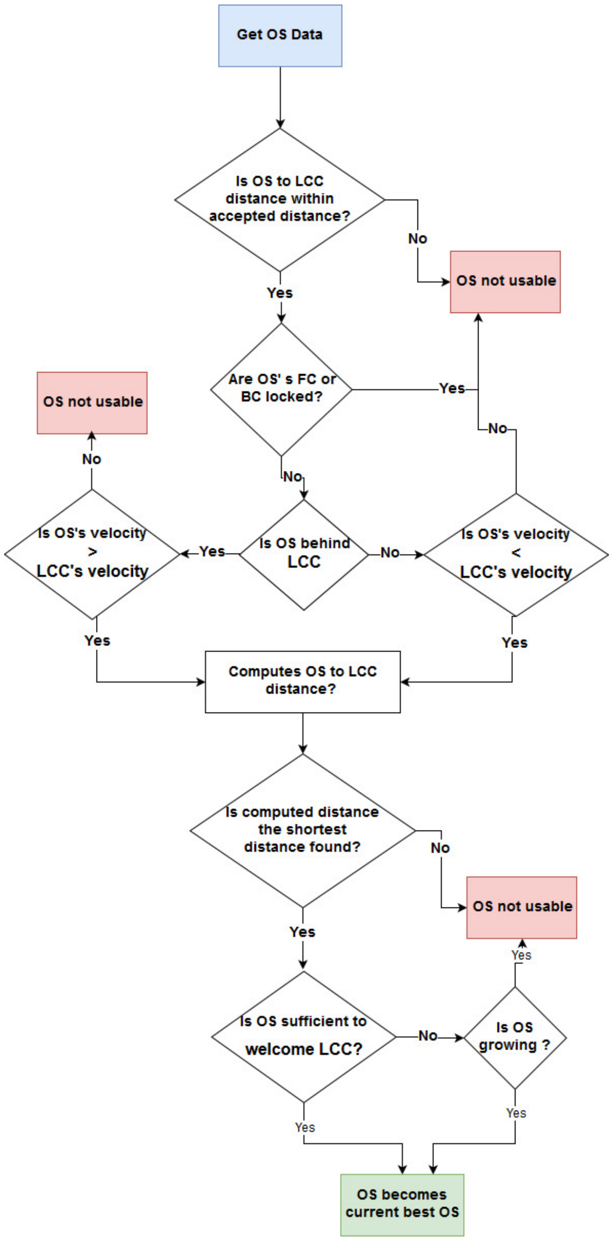

Figure 8: Flowchart illustrating the decision making process of our algorithm to identify the best OS

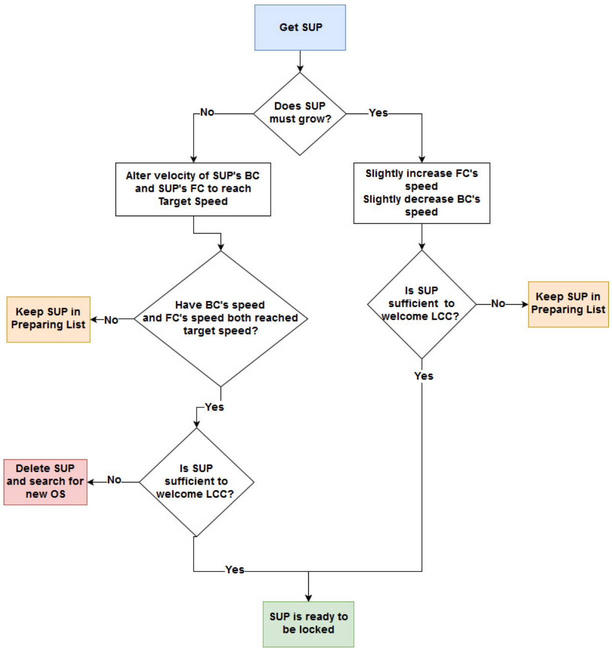

Figure 9: Flowchart describing the process of preparing an open space before locking it

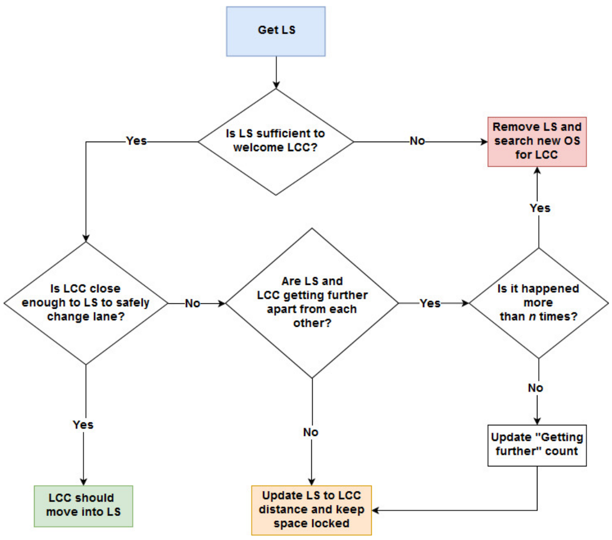

Figure 10: Flowchart illustrating the process that determines if a lane changing vehicle can safely move into the locked space

space's middle point to perform a safe lane change. It would be dangerous to let the car move into the locked space either right in front of the locked space's back car or just behind the locked space's front car. The system must assure that the safety distance between vehicles is always preserved. To do so, the algorithm must center the landing distance, previously calculated, around the open space's middle point. If the LCC finds itself within this landing distance, the system will order the vehicle to move into the space.

To avoid the LCC being endlessly trying to reach its dedicated space, the system must detect if the locked space is still reachable. To do so, the distance between the LCC and the locked space will be calculated. If the distance between the two entities is growing, a counter value will be updated and incremented by one. If this counter reaches n number of attempts, the locked space will be unlocked and the system will need to find a new open space for the LCC to insert itself into. During our evaluation scenario, the threshold value n was set to 5. The above steps are summarised in Figure 10.

V. Performance Evaluation

To implement our proposed algorithm and evaluate its performance, we used the microscopic road traffic simulator SUMO. Our algorithm has been implemented in Python 3, using both SUMO and TraCI's (Traffic Control Interface) Python 3 libraries. The algorithm is tested on a 2 km road representing 5 lanes highway scenario. The departure of each vehicle, and thus the traffic flow, is randomised using a seed value which is a random number between 0 and 100000. In this scenario, vehicles change their current lane for a reason (i.e., to change their directions or drive faster etc.) using a TraCI's API function that detects if a vehicle wants to change its current lane, either to the left or to the right, and if it can do so. This way has been picked over a randomisation of lane change along the traffic as the latter might force a vehicle that does not need to change lane to actually do so. The SUMO default vehicle's $laneChangeMode$ has been amended to prevent any lane change required by SUMO to be performed. Thus, all the lane changes occurring during the simulation are triggered by our algorithm. The simulation settings are summarised in Table II.

Parameter	Value
Topology	2 km long road
Nb of Vehicles	100,500,1000
Nb of Simulation epochs	30
Seed Value	range (0,100000)
Scenario tested	5 lanes highway scenario

Table II: Simulation setting

A series of tests were performed on the algorithm to assess its validity. Below are the results extracted from the simulation measured in terms of Average Trip Duration (ATD), the average "Time Loss" and safeness data regarding crashes and Safety Gap distances. "Time Loss" is a metric calculated by SUMO during the simulation by computing the ideal speed that each vehicle should move at, and at any point if the vehicle is moving below this speed the corresponding time loss is measured. The results depicted below represent the average values of 30 simulation runs, with a different seed value, for each run. Please note that Non-LCC vehicles refer to vehicles which did not change their lane during the simulation.

Figure 11 shows that the ATD achieved by the LCCs is slightly higher than that achieved by Non-LCC's across the three scenarios. The LCCs' ATDs are higher by +3% for 100 vehicles, +5% for 500 vehicles and +4% for 1000 vehicles. This increase is due to the speed adaptation required by our algorithm as well as the time taken to move to the new lane by the LCCs.

To better understand the variability or dispersion of ATD data we plot a Box-plot in Figure 12. From this figure we can see that the majority of ATDs range between 95 seconds to 114 seconds across the three scenarios with some trips having duration outside this range (i.e., higher than 114 seconds). These trips are associated with vehicles involved in the Lane change process as being either a LCC or one of the vehicles delimiting the selected best OS. A snapshot of a single simulation run representing the frequency of trip duration for both LCCs and Non-LCCs is shown in Figure 13.

Figure 14 shows that the incurred Time Loss for the LCCs is slightly higher than that incurred for Non-LCC's across the three scenarios. The LCCs' time loss is higher by +9% for 100 vehicles, +7% for 500 vehicles and +10% for 1000 vehicles. This increase in time loss is due to the speed adaptation required by our algorithm, especially when the OS is locked and the LCC starts performing the lane change manoeuvre.

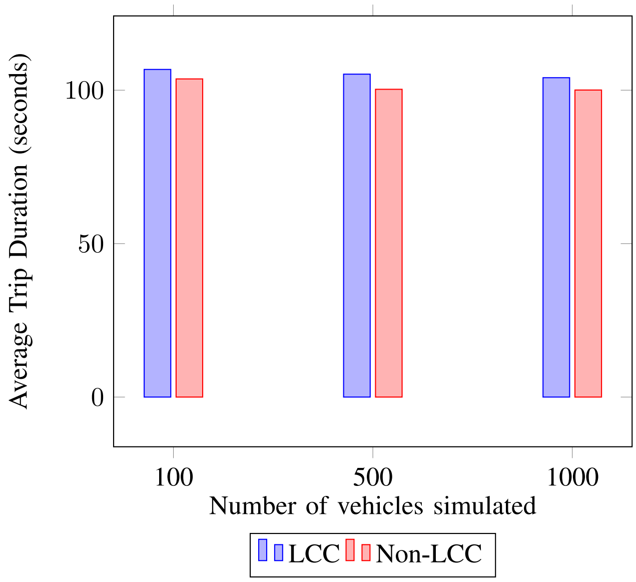

Figure 11: Illustration of the variation of ATD based on the traffic volume

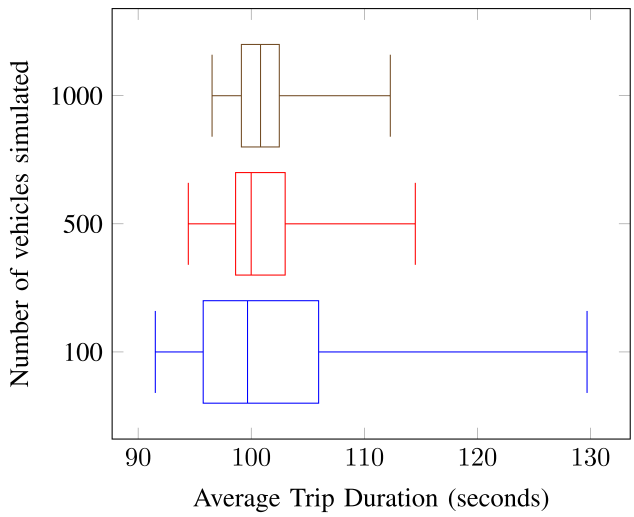

Figure 12: Box-Plot representing all ATD values gathered for each of the 3 scenarios

Vehicles Simulated	Crashes	Lane change respecting Safety Gap Distances
100	0	100%
500	0	100%
1000	0	100%

Table III: Data highlighting the safeness of our algorithm in 3 scenarios

The data depicted in Table III highlight that our algorithm

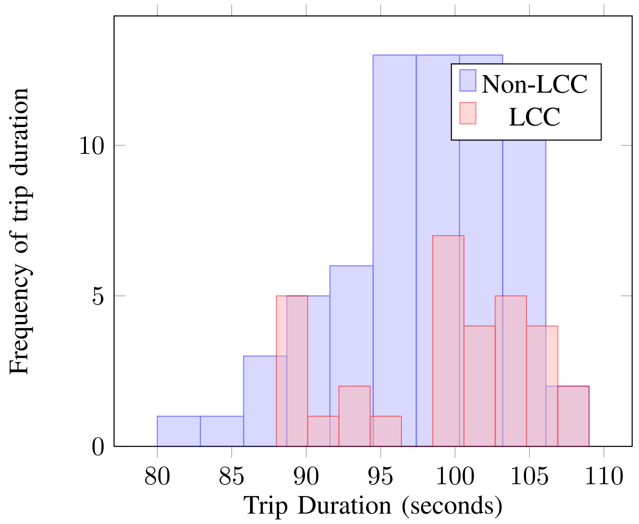

Figure 13: Histogram showing the frequency of trip duration in a simulation scenario of 100 vehicles with a seed value of 35818

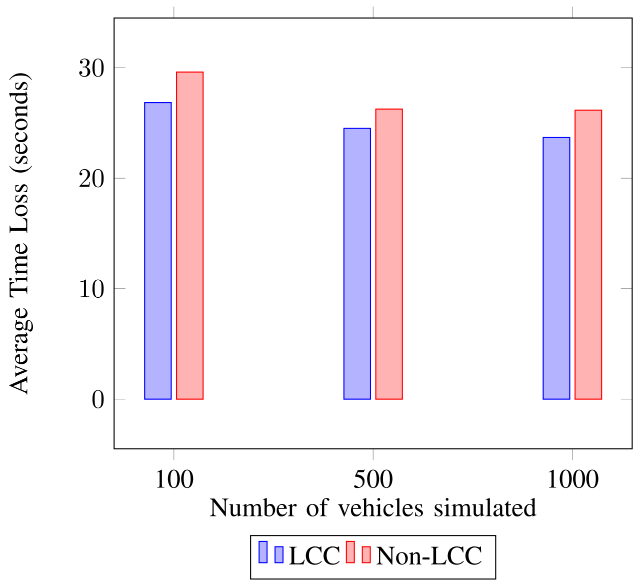

Figure 14: Illustration of the variation of the average "Time Loss" based on the traffic volume

successfully managed to fulfil its most important task, the safeness of the lane change manoeuvre. Out of all the simulations ran, no crash were recorded, no matter how many vehicles were in the traffic. Moreover, every single lane change positioned the LCC into the new lane within the required safety distance of its new leading and following vehicles.

VI. Conclusion

This paper has introduced an original infrastructure assisted lane change algorithm to aid Connected and Autonomous Vehicles (CAVs) to safely and efficiently change lane during a given trip. The obtained evaluation results, using computer simulation, were promising despite the slight increase of the average trip duration for the lane changing vehicles due to the steps involved in our algorithm. The proposed algorithm is the first step towards designing a more sophisticated lane change protocol for CAVs and could be improved in many ways. For example, the selection of the best open space can be improved by predicting the available open spaces in the near future based on the current speed of the traffic flow. We also strongly believe that our open spaces identification and tracking mechanism could be used in any other system that monitors open spaces in road traffic. As a future work, we aim to test the efficiency of our algorithm under the presence of human driven vehicles to estimate the minimum required penetration rate of CAVs that makes such algorithm beneficial.

References

[1] S. Djahel et al. A communications-oriented perspective on traffic management systems for smart cities: Challenges and innovative approaches. *IEEE Communications Surveys Tutorials*, 17(1):125–151, Firstquarter 2015.

[2] J. M. Bilal and D. Jacob. Intelligent traffic control system. In *2007 IEEE International Conference on Signal Processing and Communications*, pages 496–499, Nov 2007.

[3] K. Jo et al. Development of autonomous car-part i: Distributed system architecture and development process. *IEEE Transactions on Industrial Electronics*, 61(12):7131–7140, Dec 2014.

[4] B. Brown. The social life of autonomous cars. *Computer*, 50(2):92–96, Feb 2017.

[5] A. Kesting et al. General lane-changing model mobil for car-following models. *Transportation Research Record*, 1999(1):86–94, 2007.

[6] M. B. Younes and A. Boukerche. A vehicular network based intelligent lane change assistance protocol for highways. In *2017 IEEE International Conference on Communications (ICC)*, pages 1–6, May 2017.

[7] C. Hatipoglu et al. Automated lane change controller design. *IEEE Transactions on Intelligent Transportation Systems*, 4(1):13–22, March 2003.

[8] J. Hodgkiss et al. An advanced coordination protocol for safer and more efficient lane change for connected and autonomous vehicles. In *2019 16th IEEE Annual Consumer Communications Networking Conference (CCNC)*, pages 1–4, Jan 2019.

[9] Ieee standard for wireless access in vehicular environments (wave) – multi-channel operation. *IEEE Std 1609.4-2016 (Revision of IEEE Std 1609.4-2010)*, pages 1–94, March 2016.

[10] AASHTO. *Guidelines for Geometric Design of Very Low-volume Local Roads (ADT 400)*. AASHTO, Mar. 2019.