2018 21st International Conference on Intelligent Transportation Systems (ITSC)
Maui, Hawaii, USA, November 4-7, 2018

PERMIT – A SUMO Simulator for Platooning Maneuvers in Mixed Traffic Scenarios

Jesús Mena-Oreja and Javier Gozalvez

Abstract— **Automated driving will have a major impact on traffic. The impact can be particularly relevant under mixed traffic scenarios where automated vehicles will coexist with conventional and connected vehicles. In these scenarios, non-automated vehicles can obstruct automated driving maneuvers, and the maneuvers can influence the mobility of non-automated vehicles. Studies are hence necessary to understand the impact on traffic of a gradual introduction of automation. This paper contributes towards this objective by presenting PERMIT, an open-source platooning simulator based on SUMO and its platooning extension Plexe. PERMIT extends the current state of the art by simulating platooning maneuvers in mixed traffic scenarios where automated and non-automated vehicles coexist. PERMIT currently implements (as finite state machines) the join, merge, leave, split and leader leave maneuvers. This paper publicly releases PERMIT to the community.**

Keywords—Platooning, CACC, maneuvers, autonomous driving, automated driving, simulation, SUMO, open-source, Plexe, mixed traffic scenarios.

I. INTRODUCTION

Automated driving will have a major influence in future transportation systems and mobility services. The introduction of automation in vehicles will be gradual with some of the first innovations including Cooperative Adaptive Cruise Control (CACC) and platooning. CACC is an extension of Adaptive Cruise Control (ACC) that automates the vehicle's longitudinal control. To this aim, CACC complements radar measurements with information wirelessly received from other vehicles. Platooning is the organization of vehicles in groups, convoys or platoons where vehicles drive close to each other to augment the capacity of the road network and reduce air drag, fuel consumption and emissions [1]. Platoons use CACC and wireless communications to organize convoys and manage platooning maneuvers. Platoons include a lead vehicle that can control the formation and management of the platoon.

Several studies have analyzed the impact of automated driving on traffic. For example, [2] studies the impact of platooning on traffic under mixed traffic scenarios where automated and non-automated vehicles coexist. The report presented in [3] analyzes the impact of CACC on travel times. Existing studies provide important insights into the impact of automated driving. However, they do not consider the impact of automated driving maneuvers. Such maneuvers can have a significant impact since automated vehicles will have to coexist with conventional and connected vehicles for quite some time. Estimating the impact of automated driving maneuvers on traffic requires simulation platforms capable of realistically simulating such maneuvers. The platforms must be able to simulate the execution of the maneuvers also under mixed traffic scenarios where different types of vehicles will coexist. In these scenarios, non-automated vehicles could obstruct automated driving maneuvers. The maneuvers could also influence the dynamics of non-automated vehicles. Several studies have reported platforms for simulating platooning and CACC. However, the existing platforms have limitations to simulate platooning maneuvers. This paper progresses the current state of the art by presenting PERMIT, a new SUMO-based platooning simulator that can simulate platooning maneuvers in mixed traffic scenarios. PERMIT currently implements the join, merge, leave, split and lead platooning maneuvers. All maneuvers are implemented using finite state machines (FSM). The simulator uses the open-source microscopic traffic simulator SUMO [4] and its extension Plexe [5]. PERMIT implements all the logic needed to execute platooning maneuvers without requiring any changes in an existing Plexe scenario. PERMIT is open-source and we release the code with this paper on a public repository[1].

The rest of this paper is organized as follows. Section II reviews the existing platooning and CACC simulators. Section III introduces SUMO and Plexe, and describes the main components of PERMIT. Section IV describes the implemented platooning maneuvers and their FSMs. Section V discusses the validation of PERMIT, and Section VI summarizes the main contributions of this study.

II. RELATED WORK

Hestia [6] is one of the first platooning simulators. It simulates most of the vehicle's dynamics and sensors, but does not implement platooning maneuvers and cannot simulate mixed traffic scenarios. Similar limitations characterize the simulators presented in [7] and [8]. The simulator in [7] is an extension of SUMO for platooning simulation but is not publicly available. It implements CACC as a car-following model, and simulates inter-vehicle communications. Communications is also the focus of the platooning simulator presented in [8] that authors use to analyze the effect of communications on the string stability of a platoon.

Several CACC and platooning platforms have been developed that can simulate mixed traffic scenarios but not automated driving maneuvers. This is the case of the

This work was supported by the local government of Valencia (ACIF/2017/160) and the Spanish Dirección General de Tráfico (SPIP2017-02204).

J. Mena-Oreja and J. Gozalvez are with the UWICORE Laboratory of the Universidad Miguel Hernández de Elche (UMH), Elche, Alicante 03202 Spain; e-mail: {jmena, j.gozalvez}@umh.es.

[1] https://github.com/susomena/PERMIT

simulator presented in [9]. The simulator implements the MIXIC stochastic microscopic traffic model and assumes vehicles are capable to communicate. The authors used the simulator to analyze the impact of CACC on traffic flows. [10] presents a related study where the authors analyze the impact of CACC on the road capacity. The simulator models automated and non-automated vehicles on the AIMSUN microscopic traffic simulator, but does not implement automated driving maneuvers. VISSIM is the simulator utilized in [3] to study the impact of CACC on travel times in mixed traffic scenarios. To the authors' knowledge, the simulations did not take into account the effect of automated driving maneuvers. The platooning simulator used in [2] models mixed traffic scenarios with automated and non-automated vehicles. The authors investigate the impact of platooning on traffic, but the simulator does not implement platooning maneuvers. In fact, the platoons are already formed when they enter the simulation, and never dissolve or execute any other maneuver.

Plexe [5] is currently the most relevant open-source platooning extension for SUMO [4]. Plexe implements different CACC car-following models, and simulates wireless communications for CACC and platoon management. Plexe can implement platooning maneuvers, but it currently only includes a join maneuver that is available in a branch of its repository[2]. Plexe provides the tools needed to simulate mixed traffic scenarios. Plexe and SUMO are the reference platforms for creating PERMIT given their characteristics and potential. PERMIT adds to Plexe and SUMO the capacity to simulate platooning maneuvers, in particular the join, merge, leave, split and leader leave maneuvers. The maneuvers are implemented following key state of the art contributions such as [11] and [12]. The study reported in [11] presents a simulator to manage platoons using the SUMO-based platform VENTOS. The simulator implements different platooning maneuvers that are used as a reference to build PERMIT. The main objective is to evaluate the maneuvers, the CACC controllers, and the impact of wireless communications on the maneuvers. Since the impact of platooning maneuvers on traffic is not an objective of [11], the platform does not simulate mixed traffic scenarios like in [12] and [13]. In [12], the authors implement platooning maneuvers for a vehicle to join an existing platoon and for two platoons to merge. [13] uses Finite State Machines (FSM) to create platooning maneuvers. Unfortunately, the paper does not provide sufficient details to be able to implement the maneuvers using their FSMs.

III. Simulation Environment

PERMIT is a Python program that implements platooning maneuvers in mixed traffic scenarios with automated and non-automated vehicles. PERMIT uses Plexe-SUMO to simulate the traffic. Fig. 1 depicts the architecture of PERMIT, and the next sub-sections describe its different modules.

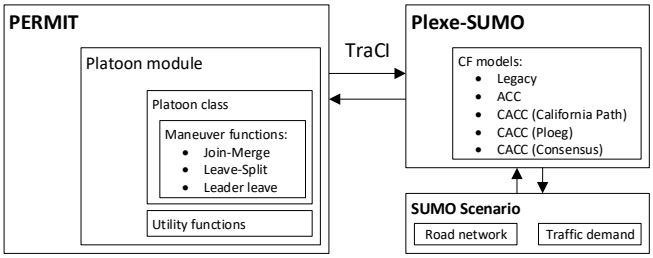

Figure 1. PERMIT architecture.

A. SUMO

SUMO [4] is an open-source microscopic traffic simulator. The simulations in SUMO are continuous in space and discrete in time. SUMO models the road network and the traffic demand. A user can create a road network in SUMO using a graph where nodes represent intersections and edges represent road segments. Alternatively, the road network can be imported from an external source. The traffic demand is defined as a series of traffic flows consisting of a stream of vehicles with a common origin and destination. SUMO also allows users to generate and configure individually each vehicle in a simulation. In fact, SUMO controls every vehicle individually, and every vehicle has its own route and vehicle type. The vehicle type includes the car-following model that determines the longitudinal dynamics of the vehicle, and the lane-changing model that determines its lateral dynamics. Our current implementation uses the default car-following (Krauss [14]) and lane-changing (LC2013 [15]) models for non-automated vehicles. However, this can be changed when creating the traffic demand for a particular scenario.

B. Plexe

Plexe [5] defines extensions for the SUMO traffic simulator (Plexe-SUMO) and for the wireless communication simulator Veins (Plexe-Veins). Plexe-SUMO is a version of SUMO that implements new car-following models for the longitudinal dynamics of vehicles with an ACC or CACC controller. Plexe-SUMO uses the ACC controller introduced in [16], and implements three CACC models referred to as California PATH, Ploeg and Consensus. The three models are described in detail in [16], [17] and [18] respectively. PERMIT uses the Krauss car-following model to simulate non-automated vehicles, and by default the California PATH CACC model for the longitudinal control of vehicles in a platoon. However, users can change the CACC model when creating a new scenario. Similarly, users can change the lane-changing model that PERMIT implements by default (the LC2013 model). Plexe-Veins simulates IEEE 802.11p-based cooperative or C-ITS communications between vehicles. Simulating the exact V2X protocols and messages exchanged between vehicles is not critical to analyze at large scale the impact of automated driving on the traffic. As a result, PERMIT simplifies the communications between vehicles in order to reduce the simulation time. In particular, PERMIT does not simulate inter-vehicle communications, and assumes that all vehicles within a given range (this range is a parameter introduced when executing PERMIT) can communicate without errors. Consequently, PERMIT does not use Plexe-Veins.

[2]https://github.com/michele-segata/plexe-veins/tree/plexe-2.0-join-example

C. TraCI

TraCI (Traffic Control Interface) is an interface for connecting to a SUMO simulation (using TCP), and for obtaining or modifying values of the simulated objects. PERMIT uses TraCI to obtain variables of the vehicles (e.g. speed, position and acceleration) required by the ACC and CACC controllers. PERMIT also uses TraCI to get the list of vehicles in the simulation. Using this list, PERMIT access all the data necessary from each vehicle (e.g. position, speed and lane) to manage platoons and perform platooning maneuvers safely. The authors of Plexe give some examples of how TraCI can be used from Python to access a Plexe-SUMO simulation with Plexe-Veins being disabled. These examples are available in an open-source repository[3]. We have used these examples as a starting point to develop PERMIT.

IV. Platooning Maneuvers

This section describes the platooning maneuvers implemented in PERMIT following the architecture illustrated in Fig. 1. PERMIT implements the join, merge, leave, split and leader leave maneuvers. The simulator models every automated vehicle that does not belong to a platoon as a platoon of just one vehicle. In this case, the implemented maneuvers can model most platooning maneuvers. For instance, a platoon is formed by merging platoons of one vehicle each. Most of the maneuvers are implemented as Finite State Machines (FSM) following the contributions in [11]-[13] and [19]. PERMIT controls the movement of platoon leaders using an ACC controller, and the movement of platoon followers or members using a CACC controller. The following sub-sections detail the implementation of the maneuvers and their interaction with non-automated vehicles.

A. Join and merge maneuvers

The *join-merge* function in PERMIT executes the:

- Join maneuver: a vehicle joins an existing platoon. The vehicle can be located behind, in parallel, or in front of the existing platoon. In the latter case, the vehicle joining the platoon will become the platoon leader.
- Merge maneuver: merge of two (or more) existing platoons. If the two platoons circulate on the same lane, the rear platoon catches the front platoon. If they circulate in parallel lanes, both platoons merge into a single lane.

At every time step of the simulation, all automated vehicles (modeled as platoons with just one vehicle) and platoons check if they can merge with another platoon. Vehicles and platoons initiate a join or merging maneuver if:

- The distance between platoons does not exceed a maximum distance introduced as a parameter to the simulator.
- The total number of vehicles in the merged platoon does not exceed a maximum value introduced as a parameter to the simulator.
- The relative speed between both platoons does not exceed a maximum value introduced as parameter to the simulator.
- No other vehicle (automated or non-automated) or platoon obstructs a merge maneuver. A vehicle obstructs a merge maneuver if it is in the trajectory of any of the vehicles involved in the maneuver. If other vehicles obstruct the maneuver, the platoons abort the maneuver as suggested in [19].

The rest of this section focuses on merge maneuvers since PERMIT implements the join maneuver as a particular case of the merge maneuver where automated vehicles behave like a platoon of one vehicle. Fig. 2 illustrates a merge maneuver between two platoons. *i* indicates the position of a vehicle in the platoon. The position of the platoon leader is equal to *0*. Vehicles within a platoon maintain a *desired gap* (*d* in Fig. 2) with their vehicle in front. Fig. 2.a. represents the position of vehicles in the two separate platoons. When a merging maneuver starts, the simulator identifies first the new platoon leader and the position of vehicles in the merged platoon (Fig. 2.b.). The position is determined following the location of the front bumper of each vehicle. Vehicle *1* is a platoon follower in Fig. 2.b. but was a platoon leader in Fig. 2.a. As a result, it must change its controller to CACC at the start of the merge maneuver. At the same time, the leader of the new platoon decides the lane in which the merged platoon will circulate (*desired lane*). Currently, PERMIT defines that the platoon chooses the leftmost lane if vehicles travel at the road's maximum speed and the rightmost lane otherwise. Following the order identified in Fig. 2.b., vehicle *1* must change its lane and follow vehicle *0* in the left lane. To this aim, vehicle *1* decelerates until it is at a distance equal to *safe gap* (*s* in Fig. 2) to vehicle *0* as illustrated in Fig. 2.c. The *safe gap* is introduced as a parameter of the simulator. Its objective is to ensure vehicles safely change lanes. When the distance between vehicles *0* and *1* is equal to the safe distance, vehicle *1* stops the deceleration and maintains the same speed as vehicle *0*. Vehicle *2* starts then decelerating until its distance to vehicle *1* is equal to *safe gap* (Fig. 2.d.). Vehicle *1* safely changes the lane (Fig. 2.e.), and vehicle *3* starts decelerating in order to increase the distance to vehicle *2* and conduct a similar maneuver to that conducted by vehicle *1*. Vehicles *1* and *2* then accelerate to reduce the distance to vehicles *0* and *1* respectively to *desired gap* (Fig. 2.f.). The implemented maneuver prevents all vehicles breaking at the same time to open a safe gap to their front vehicle. This increases the time needed to execute the merge maneuver but avoids excessive decelerations at the tail of the platoon and increases safety and comfort in the maneuver [12]. Fig. 3 illustrates this effect by comparing the evolution of the speed of vehicles involved in a merge maneuver like the one illustrated in Fig. 2. Fig. 3 has been obtained using PERMIT.

Fig. 4 represents the FSMs implemented for platoon leaders and followers in order to execute the merge maneuver illustrated in Fig. 2. The FSMs have been created following indications reported in [11] and [12]. Two states define the dynamics of platoon leaders (Fig. 4.a):

[3] https://github.com/michele-segata/plexe-python-demo

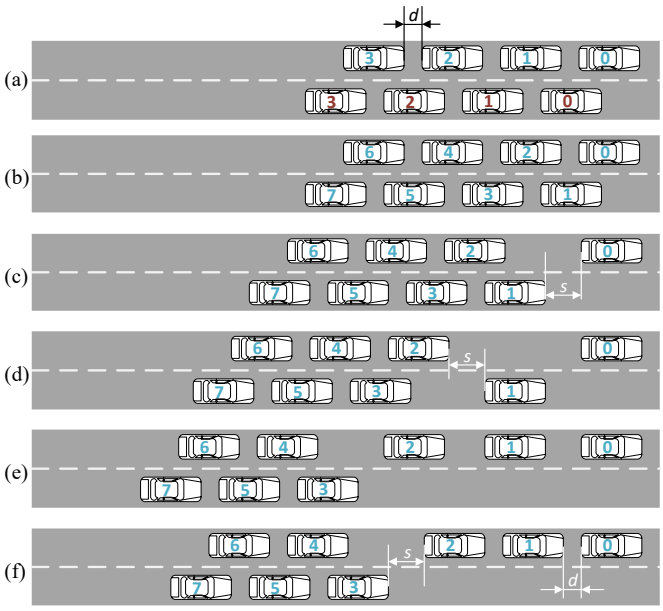

Figure 2. Merge maneuver.

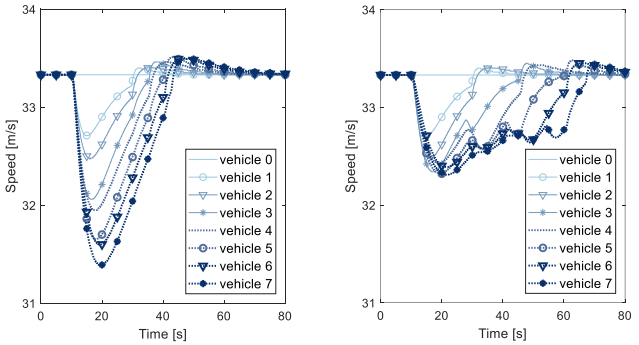

Figure 3. Speed of vehicles participating in a merge maneuver (a) vehicles decelerate at the same time and (b) vehicles decelerate sequentially like in Fig. 2.

- IDLE: Platoon leaders are always in IDLE state except when a merge starts. In this case, leaders move to the MANEUVER state.
- MANEUVER: A platoon leader analyzes if it needs to change the lane to execute the maneuver. If it is the case, the leader changes the lane (when it is safe, i.e. no other vehicle obstructs the maneuver) and returns to IDLE state. If the change is not necessary, the leader automatically returns to IDLE state.

Fig. 4.b. represents the FSM for platoon followers that is next explained using the example in Fig. 2:

- IDLE: vehicles are in IDLE state if there is no active maneuver. When a maneuver starts, vehicles change their controller to CACC if necessary. Such change is for example necessary when the vehicle was previously a leader and becomes a follower during the merge maneuver (vehicle 1 in Fig. 2.b). Vehicles move then to the WAITING state.

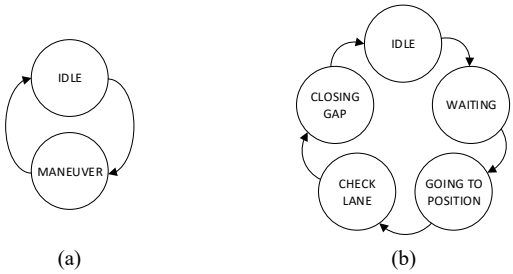

Figure 4. Merge maneuver: (a) FSM for platoon leader, (b) FSM for followers.

- WAITING: If the vehicle is the first vehicle behind the platoon leader (vehicle 1 in Fig. 2.b.), it directly moves to the GOING TO POSITION state. If not, vehicles maintain a desired gap with their front vehicle and wait until the vehicle in front has moved from GOING TO POSITION to CHECK LANE. When this happens, the vehicle moves to GOING TO POSITION.
- GOING TO POSITION: The vehicle decelerates and increases the distance to its front vehicle. The vehicle changes to the CHECK LANE state when such distance is equal to safe gap.
- CHECK LANE: If the vehicle must change the lane in the merging maneuver (e.g. vehicle 1 in Fig 2.d), it waits until its rear vehicle (e.g. vehicle 2 in Fig 2.d) is at a distance equal to safe gap. Then, the vehicle changes the lane and moves to the CLOSING GAP state. If the vehicle must not change its lane, it changes to the CLOSING GAP state as soon as its front vehicle has changed its lane.
- CLOSING GAP: Vehicles close the gap with their front vehicle to desired gap, and then move the IDLE state. A merge maneuver finishes when all the vehicles in the merged platoon return to the IDLE state.

Merge maneuvers between platoons circulating in the same lane also operate following the FSM represented in Fig. 4. In this case, the rear platoon will catch up the front platoon. As a result, vehicles belonging to the rear platoon accelerate instead of decelerate in the GOING TO POSITION state.

B. *Leave and split maneuvers*

The *leave-split* function implements the leave and split maneuvers. A split maneuver breaks a platoon in two or more platoons. Fig. 5 illustrates the leave maneuver where *i* indicates the position of a vehicle in the platoon (the position of the leader is *0*). Fig. 5.a. illustrates the position of the vehicles in the platoon at the beginning of the leave maneuver. The vehicle that wants to leave the platoon (vehicle *2* in Fig. 5.) decelerates until it maintains a *safe gap* with its front vehicle (Fig. 5.b). Vehicle *2* then maintains the same speed as the leader, and vehicle *3* decelerates until it reaches a *safe gap* with vehicle *2* (Fig. 5.c). Vehicle *3* then maintains the same speed as the leader, vehicle *2* changes the lane to leave the platoon (Fig. 5.d). Vehicle *3* then accelerates to reduce the distance to vehicle *1* (its new front vehicle) to

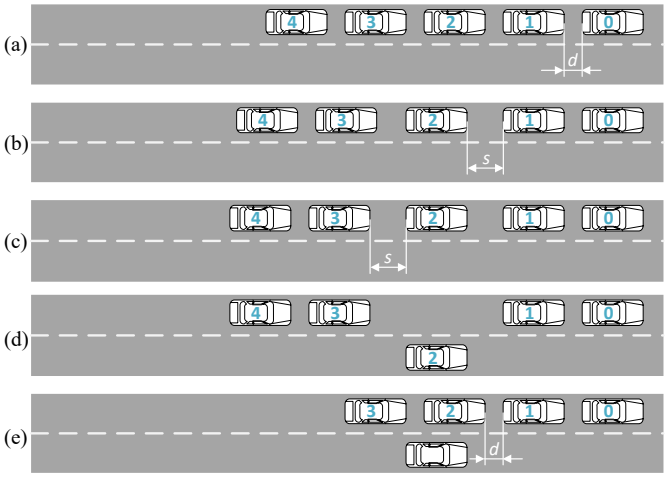

Figure 5. Leave maneuver.

desired gap (Fig. 5.e). The position of the vehicles in the platoon is updated.

Fig. 6. represents the FSMs implemented for the leave maneuver following the indications in [11]. In particular, Fig. 6.a represents the states that define the dynamics of a vehicle that wants to leave a platoon:

- IDLE: Vehicles are in the IDLE state if there is no active maneuver. The vehicle changes to the LEAVING state when the leave maneuver starts.
- LEAVING: The vehicle decelerates and increases the distance to its front vehicle. It changes to the CHECK LANE state when such distance is equal to safe gap.
- CHECK LANE: The vehicle waits until its rear vehicle is at a distance equal to safe gap. Then, the vehicle checks if any vehicles obstructs the lane change. If not, the vehicle changes the lane and returns to IDLE.

Fig. 6.b. represents the states that define the dynamics of the vehicle behind a vehicle that wants to leave a platoon:

- IDLE: Vehicles are in the IDLE state if there is no active maneuver. The vehicle changes to the OPENING GAP state when the leave maneuver starts.
- OPENING GAP: The vehicle decelerates and increases the distance to the vehicle that wants to leave the platoon. The vehicle changes to the WAITING state when such distance is equal to the *safe gap*.
- WAITING: The vehicle maintains the *safe gap* with the vehicle that wants to leave the platoon and waits until it changes the lane. The vehicle then accelerates to close the gap with its new front vehicle, and returns to IDLE. The maneuver ends when all the vehicles in the platoon are back to IDLE.

Vehicles leave a platoon whenever their route (in particular, their next road intersection) does not coincide with

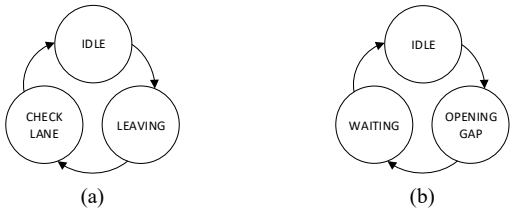

Figure 6. Leave maneuver (a) FSM for the vehicle leaving a platoon, and (b) FSM for the vehicle behind a vehicle leaving a platoon.

the route of the leader. This condition does not apply to the platoon leader. When the leader detects that it has to leave the platoon, it gives the leadership of the platoon to its rear vehicle and changes the lane to leave the platoon. A simpler *leader-leave* function simulates this maneuver.

Several vehicles can execute a split maneuver and leave a platoon at the same time with the FSMs implemented for the leave maneuver. All vehicles leaving the platoon follow the FSM in Fig. 6.a. This is not problematic if two vehicles with contiguous positions in the platoon want to leave. In this case, the 2nd leaving vehicle will maintain a *safe gap* with the 1st leading vehicle in front that can change the lane safely.

V. Experiments

We have produced a series of videos to demonstrate the correct operation of the maneuvers implemented in PERMIT. The videos are available online[4]. The merge maneuver is illustrated in a highway scenario where two platoons in parallel lanes merge, and in a scenario where a platoon merges with a platoon in front. The videos also demonstrate that PERMIT can safely abort a merge maneuver when obstructed by non-automated vehicles. A highway off-ramp scenario visually illustrates the split and leader leave maneuvers. The maneuvers illustrate the situation in which a vehicle in the middle of the platoon or the platoon leader wants to leave the highway, and hence must first leave the platoon. The videos online demonstrate the maneuvers in specific scenarios and settings. However, we have validated the implemented maneuvers in a larger number of scenarios and settings, e.g. changing the number of vehicles in the platoons, other simulator parameters, and varying the presence of non-automated vehicles in the scenario. TABLE I. lists the configurable parameters in PERMIT. Interested parties can experiment with the implemented maneuvers in other scenarios since this paper openly releases PERMIT to the community[5].

VI. Conclusions

This paper has presented PERMIT, a new open-source simulator for platooning maneuvers in mixed traffic scenarios where automated and non-automated vehicles coexist. PERMIT is based on the microscopic traffic simulator SUMO and its platooning extension Plexe-SUMO. PERMIT currently implements the join, merge, leave, split and leader leave maneuvers. The simulator can simulate the platooning maneuvers in existing SUMO scenarios without

[4]https://www.youtube.com/playlist?list=PLbi7IJ3gPs0HyA1tYTGeOR1tmN3HCjNTO

[5] https://github.com/susomena/PERMIT

TABLE I. PERMIT SIMULATION PARAMETERS

Parameter	Definition
max-step	Maximum number of simulation time steps
edge-filter	List of road edges where platooning is allowed
vtype-filter	List of vehicle types with CACC controllers
max-distance	Maximum distance to initiate a join or merge maneuver
desired-gap	Distance between vehicles in a platoon
safe-gap	Safety distance between vehicles for lane changes
platoon-length	Maximum number of vehicles in a platoon
relative-speed	Maximum relative speed allowed for merging (m/s)

any changes to the scenarios. PERMIT can simulate the platooning maneuvers with different ratios of automated and non-automated vehicles. PERMIT is released as an open-source platform for the community to further investigate and develop automated driving. The authors are currently using PERMIT to investigate the impact of platooning maneuvers on traffic under mixed traffic scenarios.

REFERENCES

[1] M. Michaelian and F. Browand, "Field experiments demonstrate fuel savings for close-following," California PATH, Research report UCB-ITS-PRR-2000-14, 2000.

[2] A. Talebpour and H. S. Mahmassani, "Influence of connected and autonomous vehicles on traffic flow stability and throughput," *Transportation Research Part C: Emerging Technologies*, vol. 71, pp. 143-163, 2016.

[3] Atkins Ltd., "Impacts of connected and autonomous vehicles on traffic flow: summary report," Department for Transport, UK Government, UK, Research report SO13994/3.

[4] D. Krajzewicz, J. Erdmann, M. Behrisch and L. Bieker, "Recent Development and Applications of SUMO - Simulation of Urban Mobility," *International Journal On Advances in Systems and Measurements*, vol.5, no. 3&4, pp. 128-138, 2012.

[5] M. Segata, S. Joerer, B. Bloessl, C. Sommer, F. Dressler and R. Lo Cigno, "PLEXE: A platooning extension for Veins," *Proc. IEEE VNC*, Paderborn, Germany, 2014, pp. 53-60.

[6] S. Hall'e and B. Chaib-draa, "A collaborative driving system based on multiagent modelling and simulations," *Transportation Research Part C: Emerging Technologies*, vol. 13, no. 4, pp. 320–345, 2005.

[7] P. Fernandes and U. Nunes, "Platooning of autonomous vehicles with intervehicle communications in SUMO traffic simulator," *Proc. IEEE ITSC*, Madeira Island, Portugal, 2010, pp. 1313–1318.

[8] C. Lei, E. van Eenennaam, W. Wolterink, G. Karagiannis, G. Heijenk, and J. Ploeg, "Impact of packet loss on CACC string stability performance," *Proc. ITST*, Saint Petersburg, Russia, 2011, pp. 381–386.

[9] B. van Arem, C. van Driel, and R. Visser, "The impact of cooperative adaptive cruise control on traffic-flow characteristics," *IEEE Trans. Intell. Trans. Syst.*, vol. 7, no. 4, pp. 429–436, 2006.

[10] S. E. Shladover, D. Su and X.-Y. Lu, "Impacts of cooperative adaptive cruise control on freeway traffic flow," *Transportation Research Record: Journal of the Transportation Research Board*, no. 2324, pp. 63-70, 2012.

[11] M. Amoozadeh, H. Deng, C.-N. Chuah, H. M. Zhang and D. Ghosal, "Platoon management with cooperative adaptive cruise control enabled by VANET," *Vehicular Communications*, vol. 2, no. 2, pp. 110-123, 2015.

[12] E. Semsar-kazerooni and J. Ploeg, "Interaction protocols for cooperative merging and lane reduction scenarios," *Proc. ITSC*, Las Palmas, Spain, 2015, pp. 1964-1970.

[13] A. S. Valente, U. Montanaro, M. Tufo, A. Salvi, and S. Santini, "Design of a platoon management strategy and its hardware-in-the-loop validation," *Proc. VTC Spring*, Seoul, South Korea, 2014, pp. 1-5.

[14] S. Krauß, *Microscopic modelling of traffic flow: Investigation of collision free vehicle dynamics*, Ph.D. dissertation, Univ. of Cologne, Cologne, Germany, 1998.

[15] J. Erdmann, "SUMO's lane-changing model," *Lecture Notes in Control and Information Sciences. 2nd SUMO User Conference*, Berlin, Germany, 2015, Springer International Publishing, 2015, pp. 105-123.

[16] R. Rajamani, *Vehicle Dynamics and Control*, 2nd ed., Springer, 2012.

[17] J. Ploeg, B. Scheepers, E. van Nunen, N. van de Wouw and H. Nijmeijer, "Design and experimental evaluation of cooperative adaptive cruise control," *Proc. IEEE ITSC*, Washington, DC, USA, 2011, pp. 260-265.

[18] S. Santini, A. Salvi, A. S. Valente, A. Pescapè, M. Segata and R. Lo Cigno, "A consensus-based approach for platooning with inter-vehicular communications and its validation in realistic scenarios," *IEEE Trans. Veh. Technol.*, vol. 66, no. 3, pp. 1985-1999, 2017.

[19] M. Segata, F. Dressler and R. Lo Cigno, "A modular approach to platooning maneuvers," *Proc. 2nd GI/ITG KuVS Fachgespräch Inter-Vehicle Communication*, Luxembourg, Luxembourg, 2014, pp. 36-39.