A Framework for MEC-enabled Platooning

Antonio Virdis, Giovanni Nardini, Giovanni Stea
Dipartimento di Ingegneria dell'Informazione
University of Pisa
Pisa, Italy
{antonio.virdis,giovanni.stea}@unipi.it, g.nardini@ing.unipi.it

Abstract—**In this paper we argue that Multi-access Edge Computing can be an enabler for platooning services. We design a MEC-based application for platooning, whereby vehicles report their speed and position periodically, and the MEC runs a platoon formation algorithm to form platoons, and a platoon coordination control algorithm, to set the acceleration of each vehicle to maintain formation with the necessary safety distance. We simulate the above framework in a realistic scenario, where communications occur through a LTE-Advanced network, and we show that it is effective and inexpensive from a communication point of view.**

Keywords—Multi-access Edge Computing, platooning, LTE, control

I. INTRODUCTION

Nowadays, more than 1 billion registered motor vehicles are circulating worldwide, and that number is expected to double in the next 10 years. As a result, a traffic congestion, accidents, energy waste, and pollution are becoming critical issues. An effective approach to mitigate the above issues is to shift the paradigm from individual to platoon-based driving. The latter is a cooperative driving pattern for a group of vehicles with common interests, in which vehicles form a line with small, near-constant inter-vehicle distance. The benefits of platoon-based driving have been shown in the literature [1], [2]: first, vehicles in the same platoon can be closer to each other, increasing road capacity and mitigating traffic congestion. Second, platooning can reduce energy consumption [3] and exhaust emissions since streamlining minimizes air drag. Third, driving in a platoon can be made safer and more comfortable through advanced technologies. Last, but not least, platoon-based driving facilitates cooperative applications (e.g., data sharing or dissemination) due to the near-constant relative position of the vehicles, which may significantly improve the performance of vehicular networking. However, a platoon is a complex physical system and requires cooperation on part of the drivers to control and manage it. Many new technologies appeared in the past ten years have acted as enablers for platooning. For instance, adaptive cruise control (ACC) can use sensors to detect the distance between adjacent vehicles and autonomously maintain the speed and/or distance. Wireless communications are another enabling technology, promoting the development of intelligent transportation system (ITS). More specifically, ubiquitous mobile communications, such as those provided by cellular networks, can support vehicle-to-vehicle and vehicle-to-infrastructure communications, making a vehicular platoon a complex cyber-physical system (CPS), in which all vehicles communicate via vehicular networking and are driven in a platoon-based pattern, with a closed feedback loop between the cyber process and physical process. In the mobile world, Multi-access Edge Computing (MEC) has recently been promoted to provide cloud and IT services within close proximity of mobile subscribers, thus making available to vehicles computing power with a short communication latency. A typical Multi-access Edge Computing scenario, envisioned in [4], consists in deploying the MEC server at the LTE macro base station (eNB) site, regarding it as an application host, with storage and computing resources, to enrich the service capabilities of the network.

MEC can act as an enabler for platooning. In fact, there are clear benefits in offloading the computations required to form and maintain a platoon to a MEC server: more complex algorithms can be used for the control loop, without the need of expensive hardware on board; these algorithms can be varied over time, adjusting to the road conditions, or the network and server congestion; applications running on a MEC server can also communicate easily with each other, achieving either distributed or centralized coordination (e.g., for optimal fleet-wise control). Despite the above, there is a substantial lack of literature on the topic of MEC-enabled platooning [18][19]. It is not even clear if such a solution is feasible at all, given the tight latencies involved in closed-loop control of a CPS.

In this paper we provide a proof of concept that MEC-based platooning is feasible: we describe the architecture of a MEC service for the formation and stability of vehicle platoons, offered by MEC servers deployed at eNodeB sites [4]. The service combines a clustering algorithm, to identify platoons, and a longitudinal controller, taken from the literature [5], to control the platoon stability. We prove the correctness of the clustering algorithm and tune the parameters for the longitudinal controller. Finally, we show results about the stability performance, obtained by simulating the above application in a network system in scenarios including a 4G cellular network as well as vehicle mobility. Our results are obtained by extending the OMNeT++-based SimuLTE simulator [14][15], that enables simulations of LTE/LTE-Advanced networks with MEC capabilities.

The rest of this paper is organized as follows: Section II discusses the background and the related work. In Section III we present our system architecture, while Section IV discusses our framework for MEC-enabled platooning. We evaluate our framework in Section V, and we conclude the paper in Section VI.

II. BACKGROUND AND RELATED WORK

This section introduces some background on MEC, LTE-advanced and platooning, also discussing the related works on the topic.

A. Multi-access Edge Computing

With MEC, the edge of the communication network is enriched by nodes with computation capabilities. Such nodes, typically called Mobile Edge (ME) servers or ME hosts, can be placed, for example, close to the radio base stations of the cellular network and can interact tightly with the latter in order to obtain valuable information on the status of the radio network and its users. This information can be exploited to offer

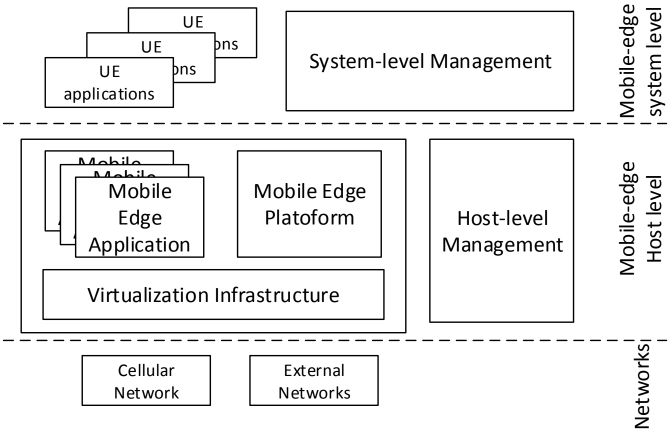

Figure 1 – Overview of the Multi-access Edge Computing Framework

new context-aware services that can take advantage of the reduced latency between the service and the end user, compared to, e.g., a cloud-based service. Examples of MEC-based use cases include computational offload for Internet-of-Things applications, smart transportation and dynamic content optimization [4].

Moreover, MEC is flexible, since ME applications run in a virtualized environment. This means that computational resources can be allocated on demand to users requesting a particular service or task. This allows the supervisor of the ME architecture (e.g., the network operator) to optimize the utilization of computational resources and possibly migrate user application to another ME host, based on a multitude of criteria, including both computation and communication requirements.

A framework for MEC is being standardized by the European Telecommunications Standards Institute (ETSI) [6]. According to that architecture, shown in Figure 1, functions are organized in two layers, namely the ME System Level and the ME Host Level. The ME System Level is responsible for keeping a global vision of the status of all the ME Hosts in the system. In particular, it receives ME Application Instantiation requests from applications running at the user side (or from the operator or third-party applications). It first checks the requirements needed by the application, such as maximum communication latency, computational resources and availability of ME services. Then, it selects – and instructs accordingly – the most suitable ME Host where the corresponding ME Application has to be instantiated, i.e. the ME Host that can satisfy the above requirements. Within the ME Host, the ME Platform provides services defined in [7] that can be exploited by ME Applications, such as: the Smart Relocation Service handles migration of ME applications to other ME Hosts; the Radio Network Information Service (RNIS), which is used to gather information from the network elements (e.g. number of users connected to a specific radio base station); the Bandwidth Manager, which defines the priority of data traffic destined to ME Applications within the ME Host; the Location Service provides information on the users' position. The Virtualisation Infrastructure is the core of the ME Host, and it runs ME Applications as instances of virtual machines, allowing them to communicate both within the ME Host (e.g., with the services of the ME Platform) and outside it (e.g., with users' local applications).

B. LTE-Advanced

MEC is expected to interact with the cellular network. In particular, LTE-Advanced (LTE-A) – or possibly its evolution being discussed in 5G forums – will be considered as one of the underlying communication networks.

The architecture of LTE-A is composed of a Radio Access Network (RAN) part and an Evolved Packet Core (EPC) part. The main components of the LTE RAN are the radio base stations, called eNodeBs (eNBs), and the cellular users, called User Equipments (UEs), as shown in Figure 2. The eNB takes care of handling cell-wise operations, including allocating radio resources to UEs for communication. In particular, every millisecond the eNB schedules a subset of frequency resources, namely Resource Blocks (RBs), to allow UEs to *i)* receive data from the eNB in the downlink (DL) direction, and *ii)* send data to the eNB in the uplink (UL) one. Recently, device-to-device (D2D) communications have been introduced so as to allow UEs in proximity to communicate directly without traversing the entire path through the eNB, hence reducing communication latency and possibly exploiting better channel quality. The EPC side is a flat IP-based network, where the Packet Data Network Gateway (PGW) represents the entry/exit point of the LTE-A network.

In such context, ME Hosts can be deployed anywhere in the EPC, possibly close to the eNBs as depicted in Figure 2 so that a single ME Host is responsible to handle the UEs connected to a small number of adjacent eNBs.

C. Platooning

Vehicle platooning is part of a suite of features that self-driving cars might employ. A platoon is a group of vehicles that can travel safely at high speed, communicating with each other to maintain short inter-vehicle distances. A lead vehicle, the Platoon Leader, controls the speed and direction, and all following vehicles (which have precisely matched braking and acceleration) adapt to the Leader's movement.

Platoon-based applications consist of three categories: 1) vehicular traffic flow optimization, 2) traffic greening and cost reduction and 3) infotainment services [8]. The primary objective for vehicle platooning is to reduce traffic congestion and improve traffic flow throughput. To this end, several platoon-related projects have been implemented in the recent past. The California Partners for Advanced Transit and Highways (PATH) project [9] aimed at improving traffic throughput by deploying platoons in highways. The EU-sponsored SARTRE

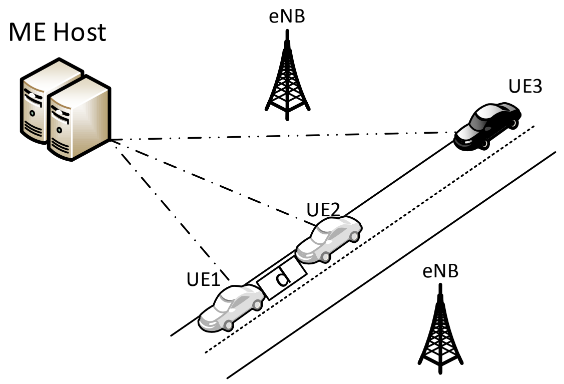

Figure 3 – Example of a MEC-enabled platooning system.

program [10] deployed a platoon on highway with a lead vehicle (typically, a truck) followed by autonomous-driving cars in close formation. Another critical issue is to improve traffic efficiency and promote greener traffic environments, such as saving traveling time, cutting down fuel consumption and reducing exhaust emissions. The Energy ITS project in Japan [11] aimed at reducing the CO_2 emissions. Experimental results show that fuel reductions of 15%-20% can be achieved by cutting the inter-vehicle distance [3]. The last category includes wireless communications that boost various infotainment applications in vehicular networking, such as platoon-aware data delivery among vehicles, cooperative local service, etc. [16][17].

To the best of our knowledge, little has been done concerning the use of MEC for platooning applications. In [18], centralized coordination of a platoon is conveyed to the platoon leader, which is responsible of gathering other vehicles' information, computing updated speeds and broadcasting the latter to other vehicles. In [19], an architecture for centralized control of platoons is proposed, where control processes are executed at road-side units. However, the feasibility of such architecture and its performance are not discussed.

III. SYSTEM ARCHITECTURE

In this section we describe the logical architecture of our MEC-enabled platooning system, identifying its main entities and connections. We consider the scenario of Figure 2 where vehicles travelling along roads are requesting platooning services to the network. For example, UE1 and UE2 belong to the *white* platoon and are coordinated by the system to maintain a safety distance *d*. Each vehicle is equipped with an LTE-A interface and is therefore seen as a UE by the network. In the following we will refer to them as either UEs or vehicles interchangeably. Multiple eNBs provide network connectivity to UEs and connect them to the MEC infrastructure and to the rest of the network (e.g. the Internet). The platooning application is run by both UEs and ME Hosts, with different roles: the former periodically report their position and movement direction, whereas the latter collect this information from all UEs, form platoons and coordinate vehicles belonging the same platoon. Each UE will receive platooning services from one ME Host at any time, but the serving host can change over time due to multiple factors, such as UE-to-ME Host proximity, communication load of the network, computation load of the ME Host, etc. Optimal pairing of UE and ME hosts on multiple MEC servers is a well-known problem [1][12], which is however outside the scope of the paper. For the sake of simplicity, in the following we will assume that the MEC infrastructure already paired each UE together with a ME host.

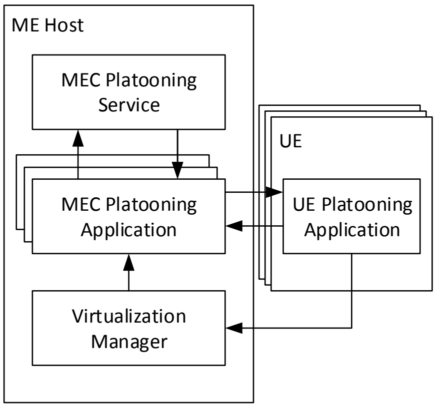

Figure 2 – Logical view of the modules within the ME Host and the UE that form the platooning service.

The high-level architecture of the ME Host and of UE that take part to the platooning service is shown in Figure 3. The ME Host is composed of three modules: the MEC Platooning Service (MPS), the MEC Platooning Application (MPA) and the Virtualization Manager. The MPS receives from the MPAs information on the cars that subscribed the platooning service and uses it to obtain an overall view of the system. It runs a platoon-formation algorithm periodically, to cluster vehicles into platoons, and a platoon-coordination algorithm, which coordinates vehicles within the same platoon to maintain a configured speed and safety distance. The decision made by the MPS are called *platoon configurations* and are forwarded to every vehicle through the MPAs by means of platoon-configuration packets. The format of the latter is shown in Figure 4a and specifies information on the platoon organization (platoon ID, follower and following car), the platoon control (target acceleration and distance) and a dissemination method. The latter information can be used to implement various dissemination techniques, e.g. using a DL transmission to the platoon leader first, then exploiting D2D transmission among platoon members in a leader-to-follower fashion. The MPS can implement multiple formation and coordination policies. In Section IV we will detail this logic and describe a possible implementation. The MPA acts as an interface between a UE Platooning Application (UPA) and the MPS. More in detail, on one hand, it receives periodically from the associated vehicle *car-information* packets, whose format is shown in Figure 4b, and forwards them to the MPS. The car-information packet contains the current car position, speed, acceleration, angular position and angular speed. On the other hand, the MPA obtains the platoon configurations computed by the MPS, and sends them to the UE. MPAs are dynamically instantiated by the Virtualization Manager upon request from the UPAs. Separating MPS and MPA allows our system to decouple the period of UPA-MPA communications from the one of platooning formation and configuration. This makes sense, since UPA-MPA communications (which travel on the uplink) need not be as frequent as platooning formation communications (which are instead in the downlink). In fact, if the UPA reports position and speed and the MPS controls platoon members by varying their acceleration (see the next

Platoon ID	Target acceleration	Target distance	follower	following	Tx mode

a

Position X	Speed X	acceleration	Angular Pos. X	Angular Speed X
Position Y	Speed Y		Angular Pos. Y	Angular Speed Y
Position Z	Speed Z		Angular Pos. Z	Angular Speed Z

b

Figure 4 – Format of platoon configurations packets (a) and car-information packets (b).

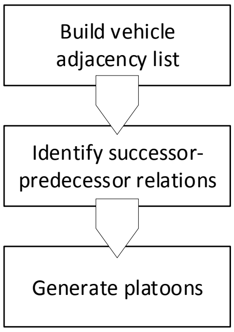

Figure 6 – Main steps of the platoon-formation service.

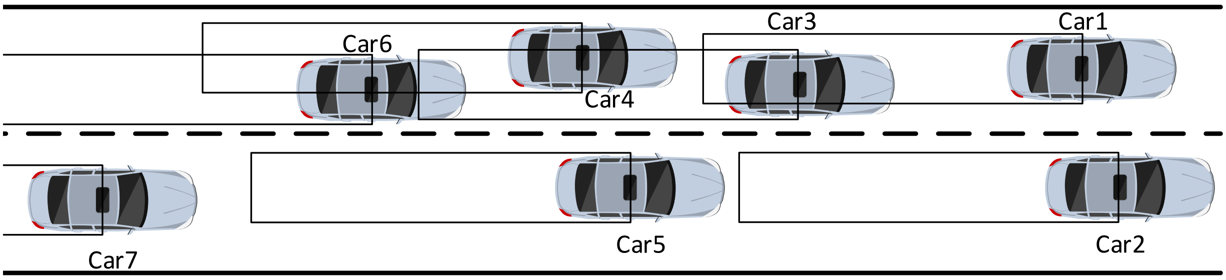

Figure 5 – Exemplary two-lane scenario with seven vehicles.

section for the details), then can compute their position accurately, and UPA-MPA communications act to correct these estimates. On the other hand, a finer-grained acceleration control is required to ensure stability and safety.

On the UE side, the UPA first requests a platooning service by sending start/stop messages to the Virtualization Manager, thus triggering the instantiation of an MPA. Once the service is up and running, the UPA retrieves information on the position and direction of its vehicle, e.g. via GPS, and periodically sends reports to its MPA. Finally, it receives platoon configurations generated by the MPS, and regulates the vehicle speed accordingly. Moreover, the UPA can be configured to relay platoon configuration to other members of the platoon, e.g. using direct communications (D2D), thus possibly saving network resources. Investigating the impact of different transmission modes (e.g., D2D vs. infrastructure-based) on the performance of the system is part of the ongoing work.

IV. PLATOON FORMATION AND COORDINATION

In this section we describe an approach to platoon formation, by means of a centralized ME Service, and a Longitudinal Controller (LC) that ensures platoon stability at safe and speed-dependent distances, first proposed in [1]. The ME Platoon-Formation Service proposed consists of a clustering algorithm to identify vehicles eligible to form a platoon along the roads and a cruise controller to regulate their accelerations in such a way to converge in the desired formation.

A. Platoon Formation

The Clustering Algorithm deals with Platoon Formation by building up chains of vehicles able to form a platoon. As shown in Figure 5, the algorithm comprises three phases:

1. For each vehicle, identifying its vehicle adjacency list: e.g. considering vehicles with similar characteristics, i.e. same road-line, traveling direction, speed, destination, etc.;
2. for each vehicle, selecting its *follower* among its vehicle adjacency list according to a given policy, e.g. closest vehicle;
3. identifying each vehicle that is *not* following another one and mark it as *platoon leader,* then piece together the vehicle chain, forming the platoon, by adding all the follower vehicles.

The above algorithm is defined in a general manner and can be specialized depending on the scenario. We propose here an exemplary implementation of the above clustering algorithm that works on a scenario wherein vehicles are travelling alongside straight roads. The implementation takes into account positions and directions of all the vehicles within the covered area to identify possible platoons:

1. The vehicle adjacency list is populated by adding vehicles with the same traveling direction and within a specific geometric area (i.e. rectangle, triangle) among all vehicles within the ME System covered area running the related ME Application. The geometric area is specified by parameters defining the shape. Figure 6 shows an example of computation: for example, car1 vehicle adjacency list comprises only car3, whereas car3 vehicle adjacency list comprises car4 and car6. Finally. car2, car5 and car7 have an empty vehicle adjacency list.
2. The follower vehicle is chosen among a vehicle adjacency list considering the closest one given their positions. Considering again Figure 6, car3 is the follower of car1, and car4 follows car3 because it is closer to car3 than car6.
3. The platoon is assembled starting from the leader vehicle, and recursively adding its followers. Taking in consideration Figure 6, car1 is the platoon leader of a platoon formed by: car1, car3, car4 and car6; whereas car2, car5 and car7 are leaders of (degenerate) single-vehicle platoons.

B. Platoon coordination

The Longitudinal Platoon Stability problem can be formulated as the control in acceleration of a vehicle trying to follow its predecessor as closely as possible. In normal driving conditions, the state of each vehicle can be characterized by its location and orientation in the plane (three degrees of freedom) and its longitudinal and angular speeds (linked through its front wheel angle). We adopt as a proof of concept the controller in [1] that ensures collision avoidance by maintaining

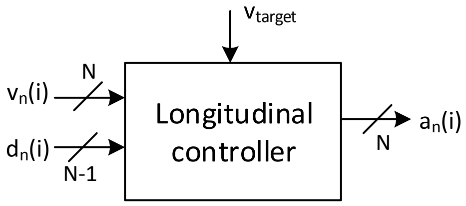

Figure 7 – Input-output representation of the platoon-coordination controller

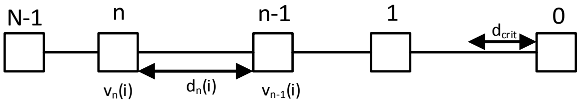

Figure 8 – Main steps of the platoon-formation service.

safe inter-vehicle distances. Evaluating the performance of different controllers within our framework is part of the ongoing work. A controller for each platoon operates periodically and with a constant period; we denote with *i* the i-th iteration of the execution of the controller. A vehicle's movements are constrained by bounds on its maximum and minimum speed and acceleration, as follows:

$$0 \le v_{\min} < v_{\max} \quad , \quad a_{\min} < 0 < a_{\max} .$$

At the beginning of every period, the controller takes as input the target speed of the platoon, the speed of each vehicle, and the distance between each vehicle and its follower. The controller then computes a new acceleration value for each vehicle, that is maintained until the subsequent period and ensures a collision-free behavior. A logical view of the platoon-coordination controller in shown in Figure 7.

Considering instead Figure 8, vehicle n must be able to avoid collision with vehicle $n-1$ at iteration i, regardless of the behavior of the latter. Thus, the worst case is considered: the safety distance should be such that, when vehicle $n-1$ brakes at maximum capacity, the maximum braking of vehicle n is enough to avoid collision. This is true when:

$$d_n(i) > d_{crit} + max\left(0, \frac{v_n(i)^2 - v_{n-1}(i)^2}{-2a_{\min}}\right)$$

where $d_n(i)$ and $v_n(i)$ are respectively the distance between vehicles n and $n-1$ and the speed of vehicle n at instant i, and d_{crit} is the critical distance below which even braking at the maximum capacity would not avoid collision. Additional details on the controller can be found in [1]. For our purposes the most significant detail is that the C++ code that implements the above controller amounts to roughly 30 lines, and only involves few simple algebraic operations. This shows that having the MPS run *N* instances of this on a MEC host is not a problem, computationally speaking, even for large values of *N* (e.g., in the hundreds or thousands).

V. PERFORMANCE EVALUATION

In this section we first validate the behavior of our system in a simple scenario, then we perform a preliminary performance evaluation of the impact of our system on the network communications, in terms of occupied transmission resources in the LTE frame.We use for this purpose SimuLTE, a system-level simulator of LTE-A cellular networks [14]. SimuLTE is based on the OMNeT++ framework and provides models for UEs and eNBs. These include implementations of LTE radio capabilities thanks to the LTE Network Interface Card (NIC), which has one submodule for every layer of the LTE protocol stack, namely Packet Data Convergence Protocol (PDCP), Radio Link Control (RLC), MAC and PHY. It also offers models for upper-layer protocols – from IP to applications – that are instead provided by the INET framework. SimuLTE has also been extended towards MEC [15], by implementing a model of a ME Host. The main simulation parameters are summarized in Table 1. Note the asymmetry between the reporting period (1s) and the platoon-coordination period (20ms).

TABLE I. SIMULATION PARAMETERS

Parameter name	***Value***
Carrier frequency	2 GHz
Bandwidth	10 MHz (50 RBs)
Path loss model	ITU Urban Macro
eNB Tx Power	46 dBm
UE Tx Power (UL)	30 dBm
UE Tx Power (SL)	15 dBm
eNB Antenna gain	18 dB
Noise figure	5 dB
Cable loss	2 dB
Mobility model	Stationary
Simulation time	100 seconds
#eNBs	20
#ME Hosts	1
#UEs	{180 , 240 , 300}
Car-information reporting period	1 second
Platoon-coordination period	20 milliseconds

We first consider a scenario composed of 1 ME Host, 20 eNBs and 5 UEs moving along a straight road, each starting at a different speed. All vehicles are included in the same platoon. The target speed is initially set to 20 m/s, and then slowed down to 7 m/s. Figure 9 and Figure 10 show how the inter-vehicle distance and the speed converge within a platoon. Cars are numbered from 4 to 0, *car[4]* is the platoon leader and does not appear in the distance plot.

We then analyze the impact of the proposed platooning scheme on the communication system, in terms of occupied transmission resources. We consider the same configuration of parameters as the previous scenario, having now an increasing number of UEs from 180 to 300. Figure 11 and Figure 12 show the average number of allocated RBs in DL and UL respectively. In the first case, the number of allocated RBs caps to a value of 2 even for 300 vehicles, thus having a low impact on network resources, even considering the relatively small period used for the execution of the platoon coordination. As for the UL, the number of occupied RBs always below one, confirming a negligible impact on the UL resources.

VI. CONCLUSIONS

This work has presented a preliminary discussion on the feasibility of MEC-assisted platooning. We have implemented a prototype of a MEC-based platooning application, which uses an available closed-loop controller, and simulated it in a scenario with an increasing number of vehicles. Our results show that our framework is feasible for three reasons: first, the loop delays are small (few milliseconds in both directions), which allow short control loop periods, and good stability and safety. Second, the impact on the communication infrastructure is low to negligible, up until a very large number of vehicles. Third, the computations required at the centralized entity are trivial, so that running even a large number of instances of the same algorithm generates a tolerable computation burden.

Future works on this topic are in several directions: on one hand, we plan to model the *computation* load at the MEC host realistically, so as to be able to include computation delays in the simulation. We could neglect this component in our present work, since the controller algorithm in [1] is computationally trivial, but this is not always the case. This would allow us to capture the behavior of computation-resource-constrained scenarios, besides communication-resource-constrained ones. Second, we plan to extend the above framework to implement different control logics, including globally optimal control (or approximations thereof), to evaluate the feasibility of controlling strategies. Third, and no less important, we plan to assess the impact of communication paradigms and network resource allocation on the feasibility of MEC-based platooning: for instance, whether and how D2D communications can help – e.g., by reducing communication delays and saving DL resources, possibly leading to shorter inter-vehicle distances, or what are the costs and benefits of platoon-aware resource allocation schemes at the eNBs.

ACKNOWLEDGMENTS

Work partially supported by the Italian Ministry of Education and Research (MIUR) in the framework of the CrossLab project (Departments of Excellence). The authors would like to thank Angelo Buono, former MSc student at the University of Pisa, who helped with the coding of the framework.

REFERENCES

[1] B. van Arem, C. J. G. van Driel, R. Visser, "The Impact of Cooperative Adaptive Cruise Control on Traffic-Flow Characteristics," in IEEE Trans. on Intelligent Transportation Systems, vol. 7, no. 4, pp. 429-436, Dec. 2006. doi: 10.1109/TITS.2006.884615

[2] P. Kavathekar, Y.Q. Chen, "Vehicle Platooning: A Brief Survey and Categorization", 2011 ASME/IEEE MESA, Washington, DC, USA, August 28–31, 2011

[3] C. Bonnet, H. Fritz, "Fuel consumption reduction in a platoon: Experimental results with two electronically coupled trucks at close spacing," SAE paper 2000 - 01 - 3056, 2000.

[4] Y. Ch. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile Edge Computing A key technology towards 5G-First edition", 2015

[5] A. Scheuer, O. Simonin, F. Charpillet, "Safe longitudinal platoons of vehicles without communication," 2009 IEEE International Conference on Robotics and Automation, Kobe, 2009, pp. 70-75. doi: 10.1109/ROBOT.2009.5152629

[6] ETSI GS MEC 003, "Mobile Edge Computing (MEC); Framework and reference architecture", 2016-03

[7] ETSI GS MEC 002, "Mobile Edge Computing (MEC); Technical requirements", 2016-03

[8] D. Jia, *et al*, "A Survey on Platoon-Based Vehicular Cyber-Physical Systems," in IEEE Comm. Surveys & Tutorials, vol. 18, no. 1, pp. 263-284, Firstquarter 2016. doi: 10.1109/COMST.2015.2410831

[9] California PATH project website: https://path.berkeley.edu/ (last accessed January 2019)

[10] A. Dávila, "SARTRE: Safe Road Trains for the Environment", Conference on Personal Rapid Transit PRT@LHR 2010

[11] Energy ITS project (Japan) website: http://itsasia-pacific.com/about-its-asia-pacific/examples-of-its-deployment-by-countryarea/2008-2012-energy-its%EF%BC%88japan/ (last accessed: January 2019)

[12] S. Wang, J. Xu, N. Zhang, Y. Liu, "A Survey on Service Migration in Mobile Edge Computing," in IEEE Access, vol. 6, pp. 23511-23528, 2018. doi: 10.1109/ACCESS.2018.2828102

[13] F. De Vita, D. Bruneo, A. Puliafito, G. Nardini, A. Virdis and G. Stea, "A Deep Reinforcement Learning Approach For Data Migration in Multi-Access Edge Computing," 2018 ITU Kaleidoscope: Machine Learning for a 5G Future (ITU K), Santa Fe, Argentina, 2018, pp. 1-8. doi: 10.23919/ITU-WT.2018.8597889

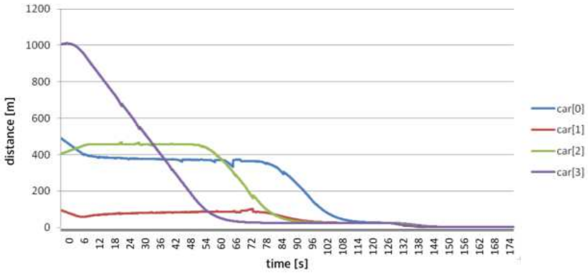

Figure 9 – Variation of the inter-vehicle distance over time.

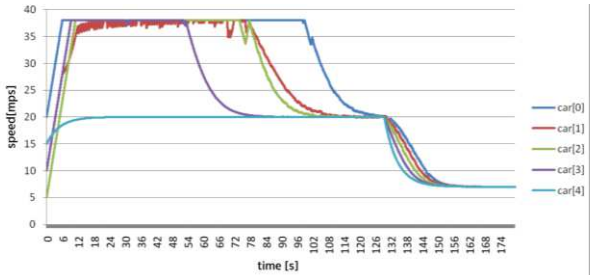

Figure 10 - Variation of the speed over time.

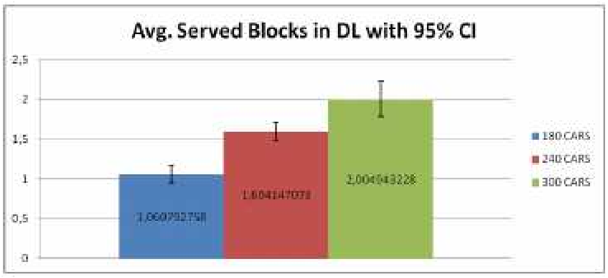

Figure 11 – Average number of allocated RBs in the DL direction for a varying number of vehicles.

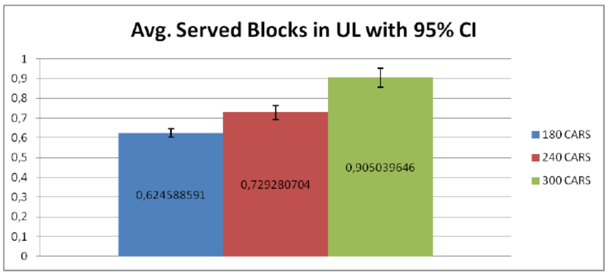

Figure 12 - Average number of allocated RBs in the UL direction for a varying number of vehicles

[14] A. Virdis, G. Stea, G. Nardini, "SimuLTE: A Modular System-level Simulator for LTE/LTE-A Networks based on OMNeT++", SimulTech 2014, Vienna, AT, August 28-30, 2014

[15] G. Nardini, A. Virdis, G. Stea, A. Buono, "SimuLTE-MEC: extending SimuLTE for Multi-access Edge Computing", OMNeT++ summit 2018, Pisa, Italy, September 2018

[16] Y. Zhang, G. Cao, "V-PADA: Vehicle-Platoon-Aware Data Access in VANETs," in IEEE Transactions on Vehicular Technology, vol. 60, no. 5, pp. 2326-2339, Jun 2011. doi: 10.1109/TVT.2011.2148202

[17] X. Li, *et al*, "Toward Effective Service Scheduling for Human Drivers in Vehicular Cyber-Physical Systems," in IEEE Trans. on Parallel and Distributed Systems, vol. 23, no. 9, pp. 1775-1789, Sept. 2012. doi: 10.1109/TPDS.2012.89

[18] R. Huanget, *et al*, "Mobile Edge Computing-Based Vehicular Cloud of Cooperative Adaptive Driving for Platooning Autonomous Self Driving," 2017 IEEE SC2, Kanazawa, 2017, pp. 32-39. doi: 10.1109/SC2.2017.13

[19] U. Montanaro, *et al*, "Cloud-Assisted Distributed Control System Architecture for Platooning," 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, 2018, pp. 1258-1265. doi: 10.1109/ITSC.2018.856