Platooning on the edge

Christian Quadri
christian.quadri@unimi.it
Computer Science Department, Università degli Studi di Milano
Milano, Italy

Vincenzo Mancuso
vincenzo.mancuso@gmail.com
IMDEA Networks Institute
Madrid, Spain

Marco Ajmone Marsan*
marco.ajmone@polito.it
Electronics and Telecommunications Department, Politecnico di Torino
Turin, Italy

Gian Paolo Rossi
rossi@di.unimi.it
Computer Science Department, Università degli Studi di Milano
Milan, Italy

ABSTRACT

Platooning of cars or trucks is one of the most relevant applications of autonomous driving, since it has the potential to greatly improve efficiency in road utilization and fuel consumption. Traditional proposals of vehicle platooning were based on distributed architectures with computation on board platoon vehicles and direct vehicle-to-vehicle (V2V) communications (or Dedicated Short Range Communication - DSRC), possibly with the support of roadside units. However, with the introduction of the 5G technology and of computing elements at the edge of the network, according to the multi-access edge computing (MEC) paradigm, the possibility emerges of a centralized control of platoons through MEC, with several significant advantages with respect to the V2V approach. For this reason, in this paper we investigate the feasibility of vehicle platooning in a centralized scenario where the control of vehicle speed and acceleration is managed by the network through its MEC facilities, possibly with a platooning-as-a-service (PaaS) paradigm. Using a detailed simulator, we show that, with realistic values of latency and packet loss probability, large platoons can be effectively controlled by MEC hosts.

KEYWORDS

Platooning; MEC; Simulation

*Also with IMDEA Networks Institute.

1 INTRODUCTION

Platooning of cars and trucks has been one of the objectives of research in autonomous driving since 1986, when Daimler launched the visionary European EUREKA 45 Project PROMETHEUS (PROgraMme for a European Traffic of Highest Efficiency and Unprecedented Safety) [20]. Standard approaches to platooning are based either on a distributed approach and direct vehicle-to-vehicle (V2V) communications, or on the support of roadside units [10]. However, with the emergence of 5G radio access networks (RANs), their attention to machine type communications (MTC) and to the automotive domain in particular, the possibility of centralized approaches based on cellular communications and multi-access edge computing (MEC) becomes interesting [18].

The advantages offered by a centralized MEC-based platoon control are many. First of all, it allows interaction among vehicles equipped with heterogeneous hardware and the integration with other systems for the management of road vehicles. In addition, it simplifies the update and the upgrade of control algorithms with no changes of the software on board vehicles. Third, it can provide better resilience to errors, since it requires redundancy only for MEC elements. Fourth, it avoids shadowing problems due to vehicles along the road (either belonging to the platoon or not) and enables to simply decouple the control of platoons and the management of emergency situations, which will continue to be in charge of on board facilities. Last, but quite relevant, it offers very good scalability, as we will see later in this paper.

On the negative side is the need for cellular coverage, but this should not be critical on high traffic motorways. In addition, it requires access to the same MEC by users of different mobile operators. However, this latter aspect should not be problematic in future sliced networks, where different tenants will offer services to their customers on a shared infrastructure.

In this paper we look at such a 5G scenario, investigating the feasibility of a platooning application where data about vehicle movement are collected by onboard sensors and transmitted over the RAN to a MEC element, where they are stored and processed to generate actuation commands. Those commands are transmitted from the MEC to vehicles, again through the RAN, so as to effectively control the platoon inter-vehicle distance. In particular, we report on the development of a detailed simulation tool for the

investigation of such a 5G platooning scenario, and we present results that allow for the assessment of the impact of communication delays and packet losses on the platoon performance.

Our main performance metrics relate to the permissible platoon length and the allowed inter-vehicle distance, with the associated saving in fuel consumption. We find that platoons of several tens of vehicles can be safely implemented with latency values easily reachable by 5G RANs, and that it is possible to obtain inter-vehicle distances allowing significant fuel savings. We also show that the introduction of a non-zero packet loss probability, up to 2%, does not undermine performance. Moreover, the paper highlights the relationship between network/computing delays and actuation lags, modelling real power train and braking system conditions, by showing the relevant impact they have on the achievable maximum absolute distance control error. Finally, we show that a MEC-centered platooning can invariably operate when the number of vehicles per platoon scales up, and this result is achieved by abundantly remaining inside the available bandwidth in up/down links and processing capability at the MEC.

The rest of this paper is organized as follows. In Section 2, we comment on previous work. Section 3 overviews legacy platoon control approaches. Section 4 presents our MEC-based approach. Section 5 briefly describes our simulation framework. Section 6 illustrates numerical results, while Section 7 discusses the impact of Quality of Service (QoS) parameters on the platooning application. Finally, Section 8 presents concluding remarks.

2 RELATED WORK

The key feature of platooning systems is the ability to dynamically and stably control the distance between vehicles that join a platoon because they have in common a significant part of their journey and wish to obtain fuel saving by exploiting the drafting effect [22].

V2V communications and road side units are normally used to enable platoon control [6, 10, 21]. However, for tight control of platoons, recent studies [7] have shown that V2V is a viable solution only when visible light communication is used to complement IEEE 802.11p and 3GPP V2V and V2X approaches using low GHz bands [12]. In general, existing works suggest to implement the platoon control function very close to the platoon, while the cloud computing paradigm is deemed as inappropriate, due to its relatively high delay average and jitter when it comes to execute a task and deliver its results to mobile users. Indeed, latency with customary cloud premises is of the order of several tens, if not hundreds of milliseconds [15]. More recently, some researchers have proposed to run platoon control on the edge of the network [8, 9, 19], where delay can be reasonably bounded below a few tens of milliseconds. Such works aim to prove the feasibility of MEC-based platoon control at a system level and based on the opportunistic offloading of computational tasks, although they do not study how delay and its variability affect performance. Instead, in this paper we provide a more fundamental contribution: we present a comprehensive study of the impact of latency on the performance of MEC-based platooning, revealing to which extent network as well as *electro-mechanic actuation lags* typical of, e.g., cars and trucks, can hinder the deployment of platooning-as-a-service (PaaS) applications. The actuation lag can be modeled as a delay resulting from a first order low-pass filter [14, 17], which we use to complement the delay imposed by data processing and communication between vehicles and MEC.

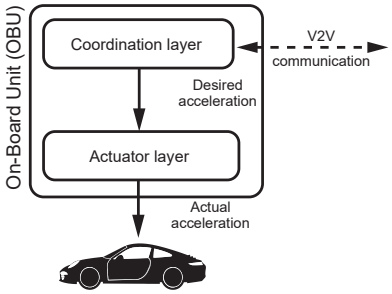


Figure 1: Structure of a legacy vehicle control system

3 LEGACY PLATOON CONTROL

In traditional platooning approaches, each vehicle cooperates with the other platoon members through V2V. Each vehicle is equipped with an *On-Board Unit* (OBU) which is in charge of (i) reading data from on-board sensors, (ii) determining which maneuvers to initiate, and (iii) exchanging sensors data with neighboring vehicles' OBUs.

The OBU. Figure 1 shows the OBU architecture. The *Coordination layer* is responsible for processing, according to the control law, the data from the local on-board sensors and the data received from other vehicles to determine the acceleration (and speed) values suitable to maintain the stability of the platoon. The *Actuator layer* is in charge of defining the throttle and/or brake commands required to achieve the desired acceleration. In practice, acceleration changes do not occur instantaneously, but rather progressively, with an actuation lag. Appendix A reports the expression for evaluating the actuation lag according to [14, 17]. The coordination layer implements a control law that guarantees *string stability*, i.e., perturbations at the head of the platoon must propagate smoothly towards the tail. Cooperative Adaptive Cruise Control (CACC) is a well known class of controllers that result in string-stable platoons [14]. We refer the reader to Appendix B for more details on CACC, which we adopt in our work.

Inter-OBU communications. OBUs exchange sensor data over wireless channels at fixed rate, using data units called Cooperative Awareness Messages (CAMs) [3] in Europe, and Basic Safety Messages (BSMs) [2] in the US. Each OBU operates independently and computes the platoon control law at fixed frequency. Inter-OBU communication may exploit two different radio access technologies: *IEEE 802.11p* operating on a 10 MHz dedicated band at 5.9 GHz, and *LTE-V2V* (or *C-V2X*, in 5G) that uses the cellular spectrum.

Pros and cons of V2V. The main advantage of a pure V2V approach is that it does not require any network infrastructure. However, V2V comes with two important drawbacks: (i) as a consequence of the lack of a centralized scheduling policy, both interference among vehicles/platoons and channel contention reduce the efficiency and the scalability of platooning; and, (ii), the communication range of on-board antennas and the shadowing of vehicles, whether within the platoon or not, limit the length of platoons. To enable longer platoons, some messages should be relayed, thus

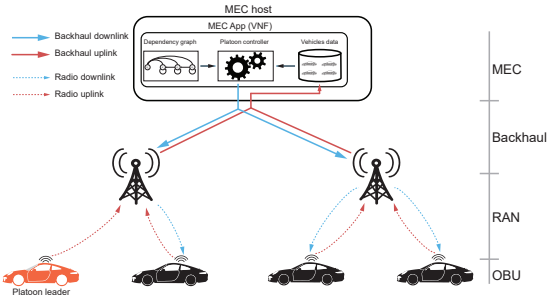


Figure 2: MEC-based platoon controller architecture

causing extra delay and increasing collisions. Cellular-controlled V2V, like LTE-V2V (mode-3), can overcome the first limitation by introducing a centralized scheduling policy at the base station (BS), but it still requires that the CAMs are exchanged by means of the V2V sidelink, and, when vehicles are connected to different BSs, it requires the adoption of an inter-BS scheduling coordination.

4 A MEC-BASED PLATOON CONTROL FRAMEWORK

The introduction of ETSI MEC in the mobile network architecture has significantly reduced delays between UEs and processors. This has made the Vehicle-to-Network (V2N) approach feasible for ITS applications/services that require small and bounded network latency. With the V2N paradigm, each vehicle communicates directly with the service provider through the RAN, i.e., the BSs, which overcomes the channel contention problem of the V2V approach.

We combine MEC and V2N communication paradigms to create a centralized platoon control service deployed as MEC application, possibly paving the way for a PaaS paradigm offering by mobile network operators. V2N communication reduces uplink interference to a negligible factor in the platoon scenario where the uplink band is limited. Moreover, the centralized approach allows virtually unlimited platoon members[1], since it is not necessary that all vehicles are within communication range of each other (not even under the coverage of a same BS). The centralized approach also eases the management of a multi-platoon scenario and the integration of other ITS services such as traffic and emergency management. Another benefit of a centralized controller approach is the simplification of the OBU since it is no longer in charge of computing the control law equations.

Figure 2 shows the architecture of our MEC-based platoon controller application, which consists of four layers:

(1) At the bottom layer lies the OBU which is in charge of (i) reading the values needed for platoon control from on-board sensors at fixed time intervals, (ii) interacting with the User Equipment (UE) module for sending sensor data over the RAN and receiving back relevant instructions, (iii) actuating the control instruction coming from the platoon controller.
(2) The next layer contains the RAN, i.e., the UEs embedded in the vehicles belonging to the platoon, and the BSs that are serving them. Multiple BSs may serve different groups of vehicles within the platoon. Moreover, we assume independent unicast transmission between vehicles and platoon controller. Packet transmission over the RAN incurs delay and may suffer losses.
(3) The backhaul network is organized in a multi-tiered aggregation ring topology, as in [13]. Transmissions over the backhaul network imply delay.
(4) The MEC layer is composed of multiple MEC hosts distributed within the backhaul network of the mobile network operator. MEC hosts are the computing elements where the platoon controller application can be deployed.

[1]The maximum number of cars in a platoon depends on the capabilities of both BSs and the MEC hosts that are serving the platoon.

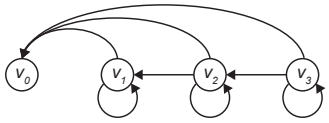


Figure 3: Dependency graph corresponding to the Leader- and predecessor-following control topology.

The platoon controller application residing in the MEC is in charge of computing the CACC algorithm for each platoon vehicle except the leader. Unlike in distributed platooning approaches based on V2V, where each OBU performs the computation, in our MEC framework the computation is centralized and relies on the data provided by each platoon vehicle; as a consequence, we instrument the controller with a database where to store vehicle data (see Figure 2), i.e., speed, acceleration and position of all vehicles. This piece of information is slightly outdated with respect to the current state of the vehicles as a result of the communication delay. Moreover, while the freshness of the stored data depends on the update frequency, commonly used fixed-frequency controller models are unsuitable and inefficient when a centralized controller is deployed on the MEC. The reason is twofold. First, the time shift between uplink transmission of the update of each vehicle's state and the downlink transmission of platoon control information causes extra delay. Second, from the controller viewpoint, the state of the platoon remains unchanged during the time interval between two consecutive updates; thus, multiple computations of the control law within this time interval are useless. Note that an increase of update and control frequency is not the most effective solution; indeed, while it mitigates the phase shift effect, it causes an inefficient use of both network and computing resources.

The above arguments led us to adopt an event-driven control model which is tied to the update frequency and computes the CACC control law only when new data are available. This approach has the advantage of issuing control instructions as soon as the state of the platoon is updated, and of limiting the amount of messages that vehicles receive.

Unlike in a V2V scenario, where each vehicle receives broadcast beacons from all vehicles within its radio coverage and autonomously decides whether or not to use the data of a vehicle according to the control topology, in a centralized approach the controller needs to know the whole control topology and derive the dependencies among platoon members. For example, let us consider the control topology *Leader-and predecessor-following* that

is often proposed for the CACC control law. We build the corresponding dependency graph (Figure 3) by reversing the direction of the edges of the control topology and by adding a loop edge for each following vehicle. Once an update message is received, the controller extracts the proximity graph of the corresponding vehicle and computes the control law for that vehicle. The loop edges are necessary because the control instruction has to be computed for each vehicle, except the leader, as soon as an update of its state is received. The amount of instructions per second sent by the controller is $(3n-4)f_u$, where n is the number of vehicles in the platoon and f_u is the update frequency.

5 A REALISTIC EVALUATION FRAMEWORK

For the evaluation of our MEC-based platoon control, we developed a discrete-event Python simulation framework that incorporates (i) the MEC application described in Section 4 and (ii) network delays and processing delays modeled as random variables with given distributions. Our MEC framework runs on top of a standard mobility simulation tool, SUMO [11], which offers a built-in platoon control interface.

Our MEC-based approach inevitably has to cope with non negligible delays, as shown in Figure 4. Starting from the left, the first delay component (i.e., T_{OBU1}) is the time the OBU requires to read data from the on-board sensors and to create the application layer message for the platoon controller application. The next three components, namely T_{RAN1}, T_{BS1}, and T_{BH1}, refer to the uplink delay from the UE to the MEC: T_{RAN1} takes into account the time required by the UE to successfully transmit the packet over the RAN uplink channel and reach the BS; T_{BS1} accounts for the time required by the BS to process the packet received from the UE and assemble the IP packet to be sent over the backhaul network; T_{BH1} represents the communication delay between the BS and the MEC host where the platoon controller is deployed. Component T_{MEC} is the processing time needed to compute control instructions for the vehicles. The delay T_{MEC} depends on the computational capabilities of the MEC host. The following three delay components, i.e., T_{BH2}, T_{BS2} and T_{RAN2}, refer to the downlink delay. In particular, T_{BH2} represents the communication delay between MEC host and BS, T_{BS2} is the BS processing time, while T_{RAN2} is the communication time between the BS and the UE, which includes scheduling and transmission times. Finally, T_{OBU2} is the time required by the OBU to process the packet received from the UE and to transfer the instructions to the physical controller of the vehicle, while T_{ACT} models the actuation lag of the power/braking system of the vehicle. As described in previous works like [7, 16], this latter delay component strictly depends on the specific vehicle.

The overall delay observed at the platoon is a random variable that results from the sum of all components described above. Since we are interested in modeling the delay impact on performance, our simulator allows the selection of the distribution of delay and its parameters.

6 PERFORMANCE EVALUATION

In this section we briefly describe our simulation setup, and report quantitative metrics derived from our experiments.

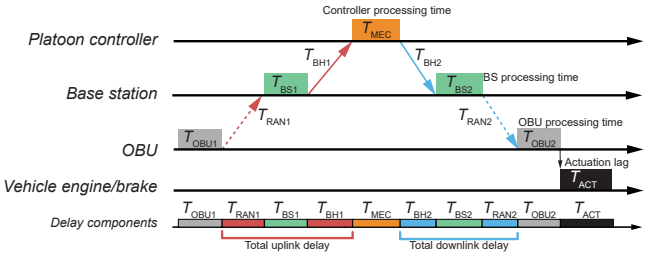


Figure 4: Delay components breakdown

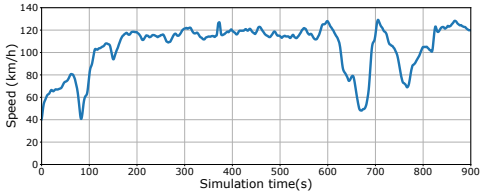


Figure 5: Speed pattern of the *real-trace*.

6.1 Simulation setup

We simulate a highway scenario with a single platoon formed by 20 vehicles travelling at 10 m distance from each other. We assume that the leader of the platoon follows a predefined speed pattern. We simulate two types of patterns, namely *sinusoidal* and *real-trace*. The sinusoidal pattern consists in a sinusoidal variation of the speed of the leader between two values at constant frequency over the entire simulation time. In line with [16] and [17], we set sinusoidal oscillation between 95 and 105 km/h at 0.5 Hz. This synthetic mobility pattern is suitable for testing the string stability to a continuous change of the leader speed. The real-trace pattern is extracted from the Floating Car Dataset [1] and its speed profile is shown in Figure 5. It lasts 15 minutes and combines a wide range of speeds, from 40 to 130 km/h, and of irregular acceleration/deceleration patterns; so we use the real-trace pattern to test the feasibility of our approach under a realistic leader behavior. Simulations are repeated 20 times per configuration, using different random generator seeds, so that confidence intervals can be computed.

In line with [16], for the simulation of the actuation lag, we use values of τ equal to 0.2s and 0.17s if the vehicle is braking or accelerating, respectively; τ is the low-pass filter time constant of the electro-mechanic actuator that enforces the instruction on brakes or engine.

For what concerns the parameters of CACC, in line with the literature [17], we use the following default values: acceleration weighting factor $C_1 = 0.5$, damping ratio $\xi = 1$, and controller bandwidth $\omega_n = 0.2$Hz.

To investigate the effect of latency, in the simulation experiments we consider the following values for delay parameters.

- The average time the OBU requires to read data from the on-board sensors and to create the application layer message for the platoon controller application (component T_{OBU1} in Figure 4) is taken as variable in the range 10 to 50 ms.

- The average uplink delay from the UE to the MEC (sum of components T_{RAN1}, T_{BS1}, and T_{BH1}) is collectively taken as variable in the range 10 to 75 ms.
- The average downlink delay from the MEC to the UE (sum of components T_{BH2}, T_{BS2}, and T_{RAN2}) is collectively taken as variable in the range 10 to 75 ms.
- The average processing time needed for the MEC to compute the control instructions for the platoon vehicles (component T_{MEC} in Figure 4) is taken as variable in the range 0.1 to 1 ms.
- The average time required by the OBU to process the packet received from the UE and to transfer the instructions to the physical controller of the vehicle (component T_{OBU2}) is taken as variable in the range 5 to 20 ms.
- The actuation lag of the vehicle power/braking system (component T_{ACT}) is modelled as a first order low-pass filter (see Equation 1) and the value of τ is equal to 0.2 s and 0.17 s for braking and accelerating, respectively [16].

As a result, in our simulation experiments, the average round trip time (RTT) before the vehicle actuation lag varies between about 30 and 220 ms.

The shape of delay probability density function (pdf), and in particular the tail of the pdf, can have a significant influence on the platoon behaviour. We consider three different distributions: uniform, with a finite tail; exponential, with an infinite but light tail; and lognormal, with unit variance and a heavier tail.

6.2 Numerical evaluation

6.2.1 Sinusoidal mobility. The results for the absolute value of the error in the distance with respect to the previous vehicle in the platoon under sinusoidal mobility are reported in Figure 7 in the case of equal average delay in the uplink and downlink paths. The three plots represent the 95th and 99th percentile, as well as the maximum observed distance error, over all simulation experiments.

Results show that maximum errors of slightly over 3 m are made with average RTT over 120 ms, while average values of RTT below 70 ms generate maximum errors of 1.5 meters only in the case of the lognormal pdf, and the uniform distribution induces maximum errors below 1 m. Looking at 99th and 95th percentiles, we observe errors always smaller than 1.5 m and 1 m, respectively.

As can be seen in Figure 6, for a fixed average RTT value, the delay in the uplink path is more detrimental to the vehicle position error. Indeed, for example, splitting an average delay budget of 125 ms into 110 ms for uplink and 15 ms for downlink yields an average maximum absolute position error of over 2 m under a lognormal delay distribution, while allocating 110 ms to the downlink yields an average maximum absolute position error of about 1 m. The 99th and 95th percentiles of the error show smaller values, but the uplink delay remains more critical, and the lognormal pdf yields largest values.

The introduction of a non-zero packet loss probability on either the uplink, or the downlink, or both, does not significantly jeopardize performance, as can be observed in Figure 8 for packet loss probability values up to 2%, sinusoidal mobility and the usual three delay distributions. This is not surprising in light of the high repetition rate of sensor data, and of the presence of a significant actuation lag.

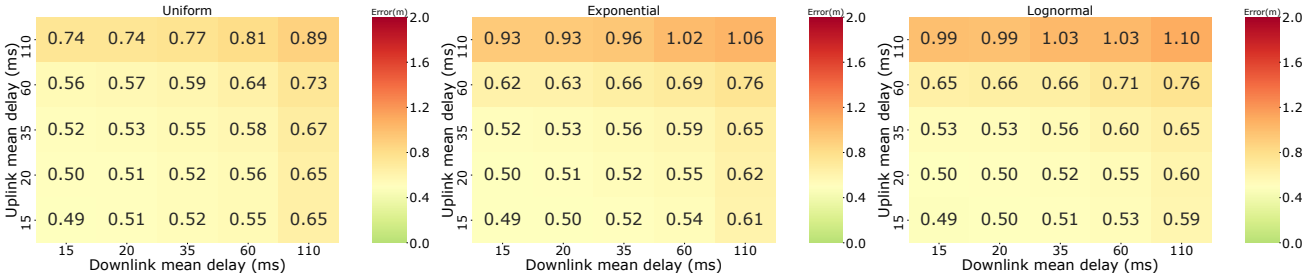
It is interesting to observe that errors in the vehicle position are typically larger for the vehicles immediately behind the leader, as requested for string stability. For example, Figure 9 shows the distributions of the absolute value of distance errors versus the vehicle position in the platoon, under sinusoidal mobility for different average delays and uniform distribution. Errors decrease from almost 2 m for the first follower to less than 1 m in the second half of the platoon. Moreover, the lower the RTT, the higher the platoon stability.

6.2.2 Real-trace mobility. Considering the mobility trace that is shown in Figure 5 provides a less challenging environment with respect to the sinusoidal mobility that we considered so far. In Figure 10 we report the 99th percentile of the absolute distance error for variable RTT. Each point represents a run of simulation, and bars show 95% confidence intervals for average values. In this case we can observe values up to slightly more than 50 cm, while in the sinusoidal mobility case we had about twice as much. In addition, we can see in Figure 11 that the imbalance of delay between uplink and downlink with the trace mobility pattern is much less critical for distance errors.

6.2.3 Actuation lag. The fact that vehicle actuation lags have time constants higher than the whole chain comprising uplink and downlink packet transmission and computation at the MEC, raises a question about the relative impact of communication/computing delays and actuation lag. We thus finally explore the impact of the actuation lag on vehicle position errors by comparing results with the normal actuation lag to results where the actuation lag is set to zero. Figure 12 shows the impact of the actuation lag on the maximum, 99th and 95th percentile of the absolute distance error for each RTT scenario with sinusoidal pattern and uniform delay pdf. We can see that the impact of lag is quite relevant, especially in the intermediate zone of RTT values, which are the ones most likely in practice. In Figure 13 we can observe the impact of the imbalance of delay on the 99th percentile of the absolute distance error with sinusoidal pattern and uniform delay pdf in both cases of normal and zero lag, which confirms that the uplink is more critical.

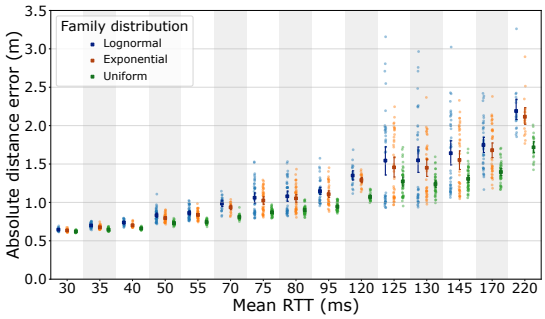
6.3 Scalability of MEC-based platooning

6.3.1 Platoon size. Scalability is quite a strong point of MEC-based platooning. Indeed, a PaaS approach allows the control of large platoons with no significant problem. For example, in Figure 14 we show the average maximum absolute distance error versus RTT for platoons with size 20 and 50, in the cases of lognormal delay distribution and either sinusoidal or real-trace mobility pattern. Results prove that no significant difference in control precision is observed in the case of long platoons. It is also interesting to observe that almost invariably the largest distance error is observed at the vehicle immediately following the platoon leader. In Figure 15 we report the boxplots of the distribution of the absolute distance error under sinusoidal mobility pattern and uniform RTT distribution with average 70ms, for platoons of size 20 and 50 (results are plotted only for vehicle 2, i.e., the one following the platoon leader, 3, 4, the

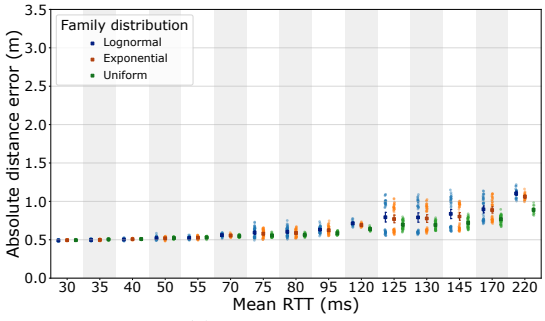


(a) Uniform delay **(b) Exponential delay** **(c) Lognormal delay**

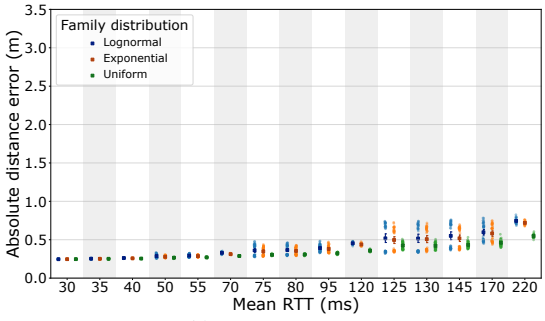
Figure 6: Heatmap of the 99th percentile of the absolute distance errors for unbalanced latency scenarios with sinusoidal pattern (mean over all simulation runs).



(a) Maximum

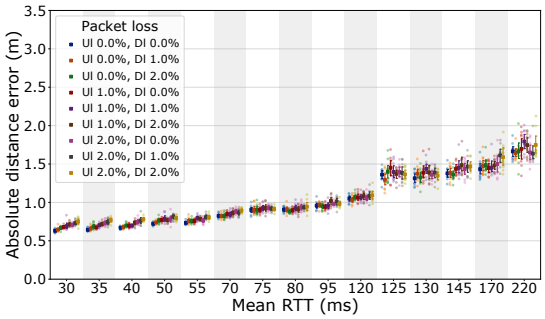


(b) 99th percentile

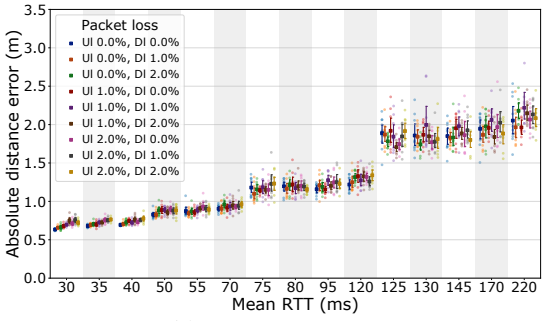


(c) 95th percentile

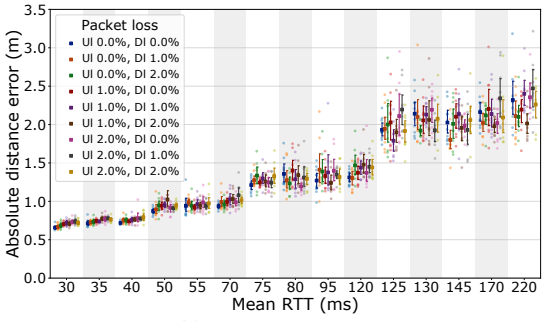
Figure 7: Maximum, 95th and 99th percentile of the absolute distance error for each RTT scenario with sinusoidal mobility pattern. Each point represents a run of simulation, the bars show the 95% confidence interval w.r.t. the average.



(a) Uniform delay



(b) Exponential delay



(c) Lognormal delay

Figure 8: Maximum absolute distance error (average over all runs) versus RTT for equal average delay on uplink and downlink under sinusoidal mobility pattern, with nonzero packet loss probability.

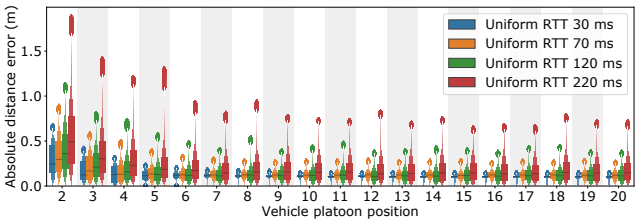


Figure 9: Boxplot of the distribution of the absolute distance error under sinusoidal mobility pattern for different vehicle positions in a platoon with 20 elements.

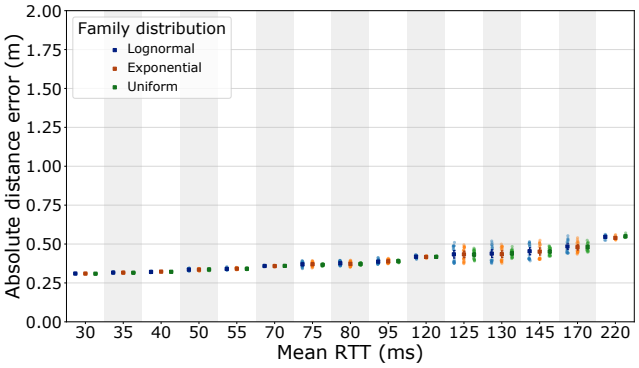


Figure 10: 99th percentile of the absolute distance error for each RTT scenario with real-trace mobility pattern. Each point reports the result of one simulation run, while bars show 95% confidence intervals.

vehicle in the middle position, and the last vehicle). Also from these results we can see that the platoon performance is insensitive to the platoon size, and that largest errors are observed for the vehicle immediately following the platoon leader.

6.3.2 Data rate and computation. As regards the base station bandwidth consumption, we recall that each vehicle transmits to the base station 10 messages per second of 200 bytes each. This means that a 50-vehicle platoon generates 0.8 Mb/s in uplink. As we mentioned in Section 4, the number of messages sent by the controller per second is equal to $(3n-4)f_u$, where f_u is the update frequency and n is the number of vehicles in the platoon. In our case, with a 50-vehicle platoon this means 1460 messages per second, corresponding to a downlink data rate of 2.336 Mb/s. By assuming that a base station controls simultaneously at most 10 50-vehicle platoons (in real settings probably much less) the data rate consumed amounts to 8 Mb/s in uplink and 23.36 Mb/s in downlink, i.e., quite a small fraction of the expected capacity of a 5G base station. Considering that currently available base stations reach capacities of the order of 1 Gb/s, assuming that the platooning application runs in a network slice with dedicated resources amounting to 10% of the base station capacity, and limiting the load of the slice to 50% in order to stay away from congestion, the slice can handle over 20 platoons comprising 50 vehicles each. With 1km coverage radius of the base station, this number of platoons looks much more than what can be expected in the foreseeable future.

Coming now to processing requirements at the MEC, a 50-vehicle platoon requires processing of 500 updates per second coming from vehicles, to compute 1460 instructions per second to be transmitted to vehicles. In total, this means processing 1,960 computations per second. In the unlikely case considered above of a base station controlling simultaneously 10 50-vehicle platoons, the amount of computations per second is 19,600. If we then assume that one MEC is shared by two base stations, the number grows to 39,200 computations, a number still easily manageable by state of the art CPUs. Assuming that the network slice for platooning is allocated 10% of one 3GHz core of one of the processors available in the MEC, that each computation required to control the platoon needs the execution of 1000 instructions, and that the utilization of the allocated computing resources must not exceed 50%, so as to guarantee good performance, the platooning application is able to perform 150 thousand computations per second. This is sufficient to manage over 75 platoons with 50 vehicles each.

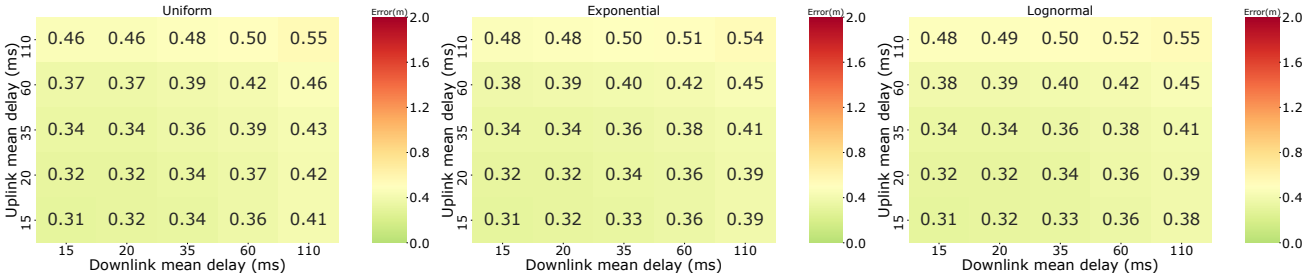
It must also be observed that the MEC-based platooning approach comes with no risk of interference among transmissions of vehicles of the same or different platoons, and with no risk of shadowing, contrary to the case of a V2V approach.

7 DISCUSSION

In the described experiments, we have observed no vehicle collision events, and the maximum distance error was always below 3.5 m. Since the dynamics of communication between vehicles and MEC do not depend on the inter-vehicular distance, this means that the reported absolute distance errors could have been achieved also by shortening the inter-vehicular distance to 5 m or less, instead of 10 m as in our experiments, and significantly less in the case with real-trace mobility. Therefore, our results apply to scenarios in which the inter-vehicular distance is set to about 4 m or more.

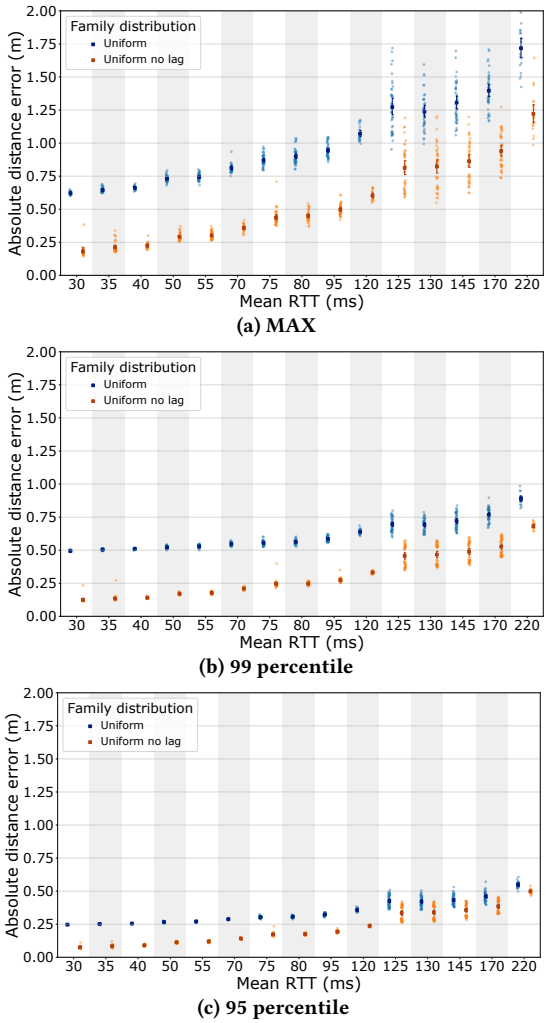
Considering the QoS class identifiers proposed by 3GPP, and in particular QCI-75 and QCI-79 [4], which are specific for V2X packet transmission, the delay budget for guaranteed bit rate (GBR) traffic is 75 ms, which goes down to 50 ms for non-GBR traffic, while the acceptable loss rate cannot exceed 1%. With the above values, we have seen that the MEC-based centralized control of platoons can guarantee sub-meter absolute errors. Instead, it is worth observing that such delay values cannot be achieved if the platoon controller runs in the cloud, since no less than 150 ms RTT can be guaranteed as of today, both due to the distance of cloud resources from vehicles and to the processing time required on remote shared servers [15].

It is important to notice that platoon control has two key motivations, namely traffic control and fuel saving, thanks to the drafting effect. So, on the one hand, platooning makes sense if the vehicles can move harmonically, with smooth transitions and with stable relative distance between vehicles. On the other hand, the inter-vehicle distance has to be short enough so that each vehicle but the leader falls in the slipstream of the preceding one. Therefore, the absolute errors discussed in this paper have to be contextualized in a scenario in which the target is to maintain an almost constant, and short, inter-vehicle distance. If we assume that the relative error cannot exceed 10% of the target distance, this means that the



(a) Uniform delay **(b) Exponential delay** **(c) Lognormal delay**

Figure 11: Heatmap of 99th percentile of the absolute distance errors for unbalanced latency scenarios with real-trace pattern (mean over all simulation runs).

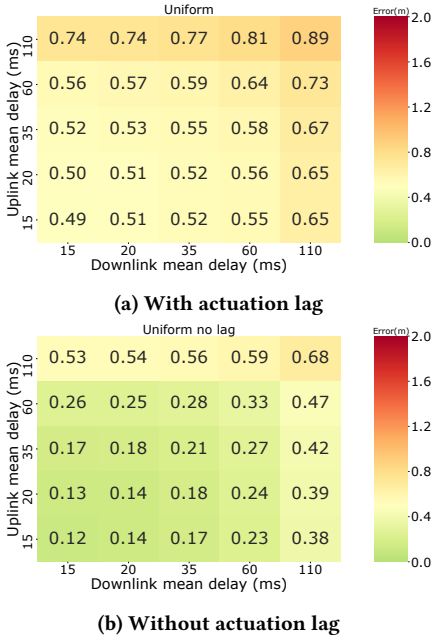


(a) MAX

(b) 99 percentile

(c) 95 percentile

Figure 12: Impact of the actuation lag on the maximum, 99th and 95th percentile of the absolute distance error for each RTT scenario with sinusoidal pattern. Each point reports the result of one simulation run, while bars show 95% confidence intervals.

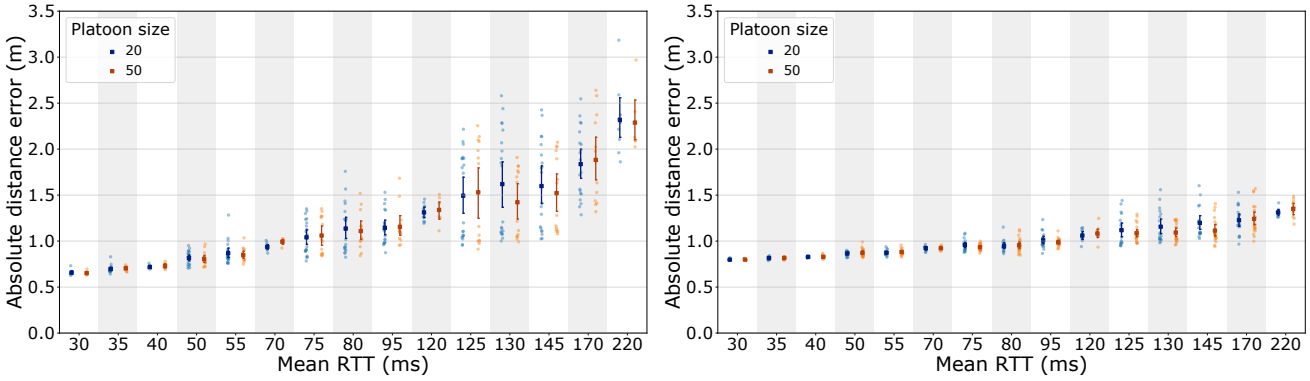


(a) With actuation lag

(b) Without actuation lag

Figure 13: Impact of the actuation lag: heatmap of 99th percentile of the absolute distance errors for unbalanced latency scenarios with real-trace pattern (mean over all simulation runs).

inter-vehicle distance that can be enforced could be about 10 times higher than the error. With the values observed in this study, we should therefore conclude that RTT delays ranging from 20 to 250 ms would result in inter-vehicular distances approximately from 5 to 35 meters, with vehicle speeds in the order of several tens of km/h. Now, while driving a few meters apart seems dangerous unless assisted driving is enabled, maintaining as much as 20 or 30 m distances seems doable without any computer-assisted machinery. In fact, consider that commonly recommended safety distances for drivers can be estimated with a simple formula, i.e., $d = 3v/10$, where d is the safety distance in meters and v is the speed in km/h. So, at 50−60 km/h, the recommended safety distance is about 15−18 m (this also corresponds to the advice to leave about three times the length of a vehicle when driving in a sub-urban environment).



(a) Sinusoidal

(b) Real-trace pattern

Figure 14: Maximum absolute distance error (average over all runs) versus RTT for different platoon sizes, with lognormal delay distribution. (a) sinusoidal mobility pattern, (b) real-trace mobility pattern.

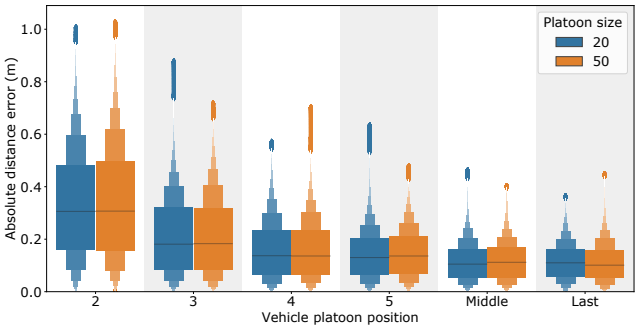


Figure 15: Boxplot of the distribution of the absolute distance error under sinusoidal mobility pattern and uniform RTT; 70ms latency scenario for different platoon sizes.

In highway driving, at 120 km/h, the safety distance should be 36 m. Therefore, it is clear that there is no need of assisted driving for such distances, which means that, with delays that allow to keep the inter-vehicle distance at more than 10 − 15 m, platooning control would bring no great benefit to traffic control. Also, at legal speed, the slipstream of a vehicle is as short as a few meters [5]. So, there would be little fuel saving by keeping inter-vehicle distances at 10 m or higher. In conclusion, our study reveals that platoon control makes sense if the RTT experienced by messages originated at the vehicles can be bounded to about 50 to 75 ms, which is in line with 3GPP recommendations, as per QCI-75 and QCI-79. However, we remark that such delays are only feasible in case of running the platoon controller either on the platoon itself or at the MEC, while cloud resources are too distant to be used. Using the MEC resources deployed by the network operators starting with 5G networks, would allow for efficient implementation of platoon control as a network service offered to drivers. This does not require computing and advanced communication tools directly on board vehicles. Moreover, with respect to V2V-based platooning, having the service handled directly by the network (or a service operator) can offer important advantages, including the possibility to easily coordinate multiple platoons (e.g., merging them when convenient or performing other maneuvers), to apply homogeneous platoon control policies (thus avoiding platoons overtakes that could block faster drivers), and scale the size of platoons beyond what can be handled with V2V communications (which, as of today, can sustain up to a fistful of vehicles using IEEE 802.11p for inter-vehicle communications).

8 CONCLUSIONS

In this paper we have investigated the feasibility of a MEC-based centralized control of platoons of vehicles, exploring the impact of the delay and packet loss probability introduced by the uplink and downlink transmissions over the RAN, of variable delay distributions, of different types of mobility patterns, and of the actuation lag within vehicles.

Our study shows that a MEC-based approach is a viable alternative to the commonly proposed distributed approach, based on V2V communications, for platooning applications. The approach relies on widely available 4G/5G mobile network infrastructures and brings together a few important side benefits—e.g., (i) the independence of performance on the number of involved vehicles and/or platoons, (ii) the natural openness to integrate different traffic control systems, and (iii) the possibility to scale to very crowded scenarios including large numbers of large platoons—that justify the further exploration of MEC-based control approaches for other autonomous driving scenarios, and paves the way to platooning-as-a-service offerings by mobile network operators.

ACKNOWLEDGMENT

V. Mancuso was supported by the Ramon y Cajal grant RYC-2014-16285 from the Spanish Ministry of Economy and Competitiveness. The work was supported by the Spanish Ministry of Science and Innovation grant PID2019-109805RB-I00 (ECID).

REFERENCES

[1] [n.d.]. TIM Big Data Challenge 2015: Floating-Car-Data-Milano. https://dandelion.eu/datagems/SpazioDati/floating-car-data-milano/resource/

[2] 2011. *Vehicle Safety Communications-Applications (VSC-A), Final Report.* Technical Report. DOT HS 811 492A, U.S. Dept. Transp., Nat. Highway Traffic Safety Admin.

[3] 2014. *Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 2: Specification of Cooperative Awareness Basic Service.* Technical Report. ETSI Std. EN 302 637-2 V1.3.2.

[4] 2019. *TS 23.203 (Rel-15); Policy and charging control architecture.* Technical Report. 3GPP.

[5] Christophe Bonnet and Hans Fritz. 2000. *Fuel consumption reduction in a platoon: Experimental results with two electronically coupled trucks at close spacing.* Technical Report. SAE Technical Paper.

[6] G. Cecchini, A. Bazzi, B. M. Masini, and A. Zanella. 2017. Performance comparison between IEEE 802.11p and LTE-V2V in-coverage and out-of-coverage for cooperative awareness. In *2017 IEEE Vehicular Networking Conference (VNC).* 109–114. https://doi.org/10.1109/VNC.2017.8275637

[7] F. Dressler, F. Klingler, M. Segata, and R. Lo Cigno. 2019. Cooperative Driving and the Tactile Internet. *Proc. IEEE* 107, 2 (Feb 2019), 436–446. https://doi.org/10.1109/JPROC.2018.2863026

[8] Xiayan Fan, Taiping Cui, Chunyan Cao, Qianbin Chen, and Kyung Sup Kwak. 2019. Minimum-Cost Offloading for Collaborative Task Execution of MEC-Assisted Platooning. *Sensors* 19, 4 (2019), 847.

[9] Y. Hu, T. Cui, X. Huang, and Q. Chen. 2019. Task Offloading Based on Lyapunov Optimization for MEC-assisted Platooning. In *International Conference on Wireless Communications and Signal Processing (WCSP).* 1–5. https://doi.org/10.1109/WCSP.2019.8928035

[10] D. Jia, K. Lu, J. Wang, X. Zhang, and X. Shen. 2016. A Survey on Platoon-Based Vehicular Cyber-Physical Systems. *IEEE Communications Surveys Tutorials* 18, 1 (Firstquarter 2016), 263–284. https://doi.org/10.1109/COMST.2015.2410831

[11] Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wießner. 2018. Microscopic Traffic Simulation using SUMO, In IEEE International Conference on Intelligent Transportation Systems. *IEEE Intelligent Transportation Systems Conference (ITSC).*

[12] Sam Lucero. 2016. Cellular–vehicle to everything (C-V2X) connectivity. *IHS Technology, Internet Everything* (2016).

[13] J. Martín-Pérez, L. Cominardi, C. J. Bernardos, A. de la Oliva, and A. Azcorra. 2019. Modeling Mobile Edge Computing Deployments for Low Latency Multimedia Services. *IEEE Transactions on Broadcasting* 65, 2 (June 2019), 464–474. https://doi.org/10.1109/TBC.2019.2901406

[14] Rajesh Rajamani. 2012. *Vehicle dynamics and control.* Springer, Chapter 7.

[15] B. P. Rimal, D. Pham Van, and M. Maier. 2017. Mobile-Edge Computing Versus Centralized Cloud Computing Over a Converged FiWi Access Network. *IEEE Transactions on Network and Service Management* 14, 3 (Sep. 2017), 498–513. https://doi.org/10.1109/TNSM.2017.2706085

[16] S. Santini, A. Salvi, A. S. Valente, A. Pescapé, M. Segata, and R. Lo Cigno. 2017. A Consensus-Based Approach for Platooning with Intervehicular Communications and Its Validation in Realistic Scenarios. *IEEE Trans. on Vehicular Technology* 66, 3 (March 2017), 1985–1999. https://doi.org/10.1109/TVT.2016.2585018

[17] M. Segata, S. Joerer, B. Bloessl, C. Sommer, F. Dressler, and R. Lo Cigno. 2014. Plexe: A platooning extension for Veins. In *IEEE Vehicular Networking Conference (VNC).* 53–60. https://doi.org/10.1109/VNC.2014.7013309

[18] Tarik Taleb, Konstantinos Samdanis, Badr Mada, Hannu Flinck, Sunny Dutta, and Dario Sabella. 2017. On multi-access edge computing: A survey of the emerging 5G network edge cloud architecture and orchestration. *IEEE Communications Surveys & Tutorials* 19, 3 (2017), 1657–1681.

[19] Antonio Virdis, Giovanni Nardini, and Giovanni Stea. 2019. A Framework for MEC-enabled Platooning. In *IEEE Wireless Communications and Networking Conference Workshop (WCNCW).* IEEE, 1–6.

[20] M Williams. 1988. PROMETHEUS-The European research programme for optimising the road transport system in Europe. In *IEE Colloquium on Driver Information.* IET, 1–1.

[21] T. Zeng, O. Semiari, W. Saad, and M. Bennis. 2019. Joint Communication and Control for Wireless Autonomous Vehicular Platoon Systems. *IEEE Trans. on Communications* 67, 11 (Nov 2019), 7907–7922. https://doi.org/10.1109/TCOMM.2019.2931583

[22] Y. Zheng, S. Eben Li, J. Wang, D. Cao, and K. Li. 2016. Stability and Scalability of Homogeneous Vehicular Platoon: Study on the Influence of Information Flow Topologies. *IEEE Transactions on Intelligent Transportation Systems* 17, 1 (Jan 2016), 14–26. https://doi.org/10.1109/TITS.2015.2402153

A ACTUATION LAG

According to [14, 17], the actuation lag can be modelled as a first order low-pass filter:

$$\ddot{x}[n+1] = \beta \cdot \ddot{x}_{des}[n+1] + (1-\beta) \cdot \ddot{x}[n] \tag{1}$$

$$\beta = \frac{\Delta_t}{\Delta_t + \tau} \tag{2}$$

where $\ddot{x}[n+1]$ is the next acceleration at the $n+1$-st simulation step which depends on the desired acceleration $\ddot{x}_{des}[n+1]$, computed by the OBU's coordination layer, and on the current vehicle acceleration $\ddot{x}[n]$. Δ_t represents the time-step of the discrete event system operations, and τ is a time constant specific of the electro-mechanic actuator implemented.

B STRING STABILITY AND CACC

The control law implemented by the OBU's coordination layer has to guarantee the platoon *string stability*. A platooning controller is string stable if it is able to attenuate the spacing error from vehicle to vehicle in a string of vehicles [14]. Formally, the string stability is defined as follows:

$$\left\| \frac{e_i(t)}{e_{i-1}(t)} \right\|_\infty \leq 1 \quad \forall i = 2 \cdots n \tag{3}$$

where $e_i(t)$ and $e_{i-1}(t)$ are the spacing errors between the i-th and preceding vehicle at time t, respectively. The platoon string stability is guaranteed by a class of controllers known as CACC. The latter provides the desired acceleration $\ddot{x}_{i\_des}$ for the i-th vehicle in the platoon by using the information of the vehicle itself, along with the information of the platoon leader and the preceding vehicles. The standard definition of CACC (see [14]) is

$$\ddot{x}_{i\_des} = \alpha_1 \ddot{x}_{i-1} + \alpha_1 \ddot{x}_0 + \alpha_3 \dot{\varepsilon}_i + \alpha_4 (\dot{x}_i - \dot{x}_0) + \alpha_5 \varepsilon_i \tag{4}$$

$$\dot{\varepsilon}_i = \dot{x}_i - \dot{x}_{i-1} \tag{5}$$

$$\varepsilon_i = x_i - x_{i-1} + l_{i-1} + d_{des} \tag{6}$$

$$\alpha_1 = 1 - C_1 \tag{7}$$

$$\alpha_2 = C_1 \tag{8}$$

$$\alpha_3 = -\left(2\xi - C_1\left(\xi + \sqrt{\xi^2 - 1}\right)\right)\omega_n \tag{9}$$

$$\alpha_4 = -C_1\left(\xi + \sqrt{\xi^2 - 1}\right)\omega_n \tag{10}$$

$$\alpha_5 = -\omega_n^2 \tag{11}$$

where x_i, $\dot{x}_i$ and $\ddot{x}_i$ are the position, the speed and the acceleration of the i-th vehicle. $\ddot{x}_0$ and $\dot{x}_0$ are the acceleration and the speed of the platoon leader, respectively. x_{i-1}, $\dot{x}_{i-1}$ and $\ddot{x}_{i-1}$ represent the position, the speed and the acceleration of the preceding vehicle. $\dot{\varepsilon}_i$ is the delta speed between the i-th vehicle and the preceding one. ε_i is the distance error w.r.t. the target distance d_{des}. CACC has three parameters: the weighting factor between the accelerations of the leader and the preceding vehicle C_1, the damping ratio ξ and the controller bandwidth ω_n.