Edge-based Platoon Control

Christian Quadri[a,*], Vincenzo Mancuso[b], Marco Ajmone Marsan[c,b], Gian Paolo Rossi[a]

[a] *Computer Science Department, Università degli Studi di Milano, Milano, Italy*
[b] *IMDEA Networks Institute, Madrid, Spain*
[c] *Electronics and Telecommunications Department, Politecnico di Torino, Torino, 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 placing control of platoons on MEC, with several significant advantages with respect to the V2V approach. For this reason, in this article we investigate the feasibility of vehicle platooning in an edge-based 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, as well as of engines and inertia of vehicles, large platoons can be effectively controlled by MEC hosts. On the one hand, we unveil that *platooning on the edge* is a viable and robust solution. On the other hand, we also shed light on the necessity to consider realistic characteristics of vehicles and speed profiles, since they can yield severe, yet not critical, performance degradation with respect to simple models.

Keywords: Platooning, MEC, Simulation

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

*Corresponding author
Email addresses: christian.quadri@unimi.it (Christian Quadri), vincenzo.mancuso@imdea.org (Vincenzo Mancuso), ajmone@polito.it (Marco Ajmone Marsan), rossi@di.unimi.it (Gian Paolo Rossi)

EUREKA 45 Project PROMETHEUS (PROgraMme for a European Traffic of Highest Efficiency and Unprecedented Safety) [1]. 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 [2]. 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 migrating the platoon control from on-board the vehicles to the infrastructure of cellular communications and multi-access edge computing (MEC) becomes interesting [3].

The advantages offered by a 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, and we show that small coverage holes can be tolerated (but large coverage holes can lead to collisions). 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, and that short disconnections of vehicles due to handovers from one base station to another do not impact the platoon behavior in a significant manner. Finally, we discuss the effect of coverage holes, showing that small coverage holes can be well tolerated, while only bigger ones, combined with speed variations of the leader, can lead to vehicle crashes. 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 7 illustrates numerical results, while Section 8 discusses the impact of Quality of Service (QoS) parameters on the platooning application. Finally, Section 9 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 [4]. Additional advantages of platooning lie in the possibility for drivers to rest, and in a better efficiency in the road use.

V2V communications and road side units are normally used to enable platoon control [2, 5, 6]. 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 [8]. 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 [9]. Notably, we show that the edge-based control of platoons is quite robust and could tolerate up to a few hundreds of milliseconds of latency under some ideal conditions, so that it could be implemented in (close enough) cloud premises. However, realistic platooning characteristics practically restrict control site selection options to the edge. More recently, some researchers have proposed to run platoon control on the edge of the network [10, 11, 12], 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. Dabbene *et al.* [13] propose a full MEC-based architecture to run platooning, and show how it can be implemented in a virtualized way, by means of Docker containers. The feasibility of platooning in 5G/MEC has been experimentally proven already, e.g., by Lekidis and Bouali [14], who provide a full library of VNFs for platooning and show that URLLC and the C-V2X connectivity can be effectively leveraged to reduce latency for platooning applications. Accordingly, there are already

studies available on how to leverage edge network resources to enforce machine learning algorithms, with the aim of optimizing cost and performance of platooning [15]. These work assume that platooning requires very low latency and paths to edge resources have to be shortened as much as possible. However, although they show edge assisted platooning feasibility, they do not search for its limits and do not shed light on the robustness of platooning to latency.

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 [16, 17], which we use to complement the delay imposed by data processing and communication between vehicles and MEC.

For what concerns cooperative driving and platooning control laws, there are several studies, most of which were inspired by the seminal work carried out in the PATH project [18]. Initially, control laws were thought for ad hoc vehicular network and required limited communication capabilities, implementing a fully distributed control [19]. Other authors considered more robust control architectures working on vehicular networks, and which are also more tolerant to delay and transmission errors thanks to the adoption of, e.g., a consensus protocol [20] or fuzzy logic techniques [21]. This class of solutions however requires topological platoon information available at the entity that computes the control law and issues directives to vehicles. The control of our edge-assisted platooning is inspired by the CACC controller [16], which is actually an outcome of the PATH project. The CACC is a proportional-derivative controller with feedforward and feedback components, and its architecture is a de-facto standard for assisted driving. In order to maintain inter-vehicle distance and string stability, CACC provides the desired acceleration to each vehicle by using the information of vehicle itself, along with the information of the platoon leader and the preceding vehicle. Some works propose variants and enhancements of CACC, e.g., Navas *et al.* [22] propose to use Youla-Kucera parametrization to build an adaptive controller on top of CACC. Vegamoor *et al.* [23] show how to analytically account for noisy V2V transmission channels in CACC, using a Gilbert channel model in the derivation of the CACC control equations. Other works propose completely different control architectures. For instance, Hao *et al.* [24] and Tian *et al.* [25] evaluate the use of LSTM neural networks to counteract the effects of communication delays. Centralized versions of CACC and cruise control protocols have been proposed as well, often to compare their stability and controllability characteristics to the ones of distributed control options [26, 27] or to more comprehensively tackle fuel reduction or even road's pavement degradation problems [28]. These works show that a centralized platooning control is viable, safe and convenient, at least in theory. Accordingly, we propose to move the CACC computation to the edge and centralize it on a CACC virtual network function. To obviate latency and transmission error effects, we further modify the process of generating platoon directives independently and asynchronously

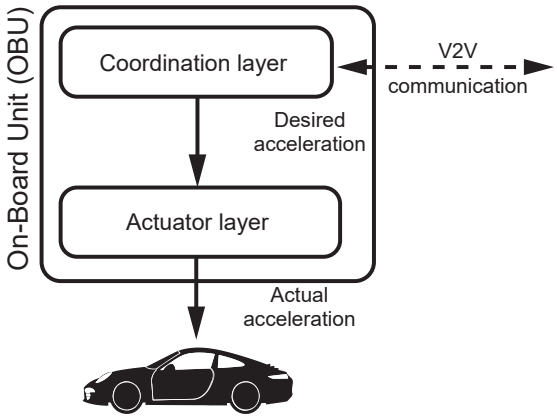


Figure 1: Structure of a legacy vehicle control system

for each vehicle, upon the reception of enough vehicle's updates.

Finally, we remark that this article builds upon and extends our early study [29]. With respect to that publication, here we provide a full description of the simulation framework, we add the study of handover and blackouts on the stability of edge-assisted platooning, and we introduce the performance evaluation of platoons with realistic engines. Specifically, the use of realistic engines leads to realistic inertia in vehicle's maneuvers, which required a complete re-evaluation of all key performance indicators. The results presented in this article are in line with what shown in [29], although here we show that realistic vehicle's characteristics have a non-negligible negative impact on platoon's string stability. We also show that realistically smooth mobility profiles reduce platooning control errors by a substantial factor (up to 5) which contrasts with the increase of error caused by real engine's characteristics studied in this article.

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 [16, 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 [16]. 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) [30] in Europe, and Basic Safety Messages (BSMs) [31] 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) 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. In fact, to enable longer platoons, some messages should be relayed, thus 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 the potential to significantly reduce delays between UEs and processors. This has made the Vehicle-to-Network (V2N) approach feasible for most 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 coverage and channel contention problems of the V2V approach.

We combine MEC and V2N communication paradigms to create an infrastructure centred 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. Such an approach allows virtually unlimited numbers of platoon members[1], since it is not required that all vehicles

[1]The maximum number of cars in a platoon depends on the capabilities of both BSs and

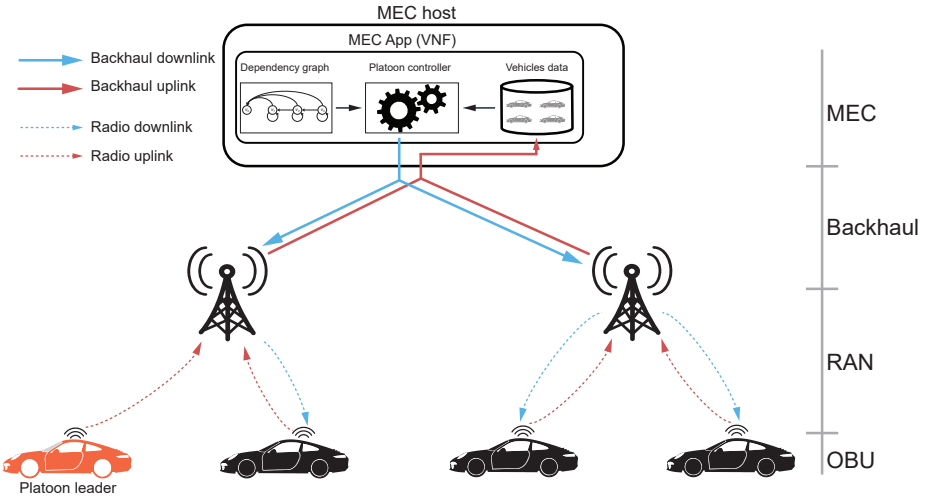


Figure 2: MEC-based platoon controller architecture

are within communication range of each other (not even under the coverage of a same BS). Migrating platoon control from vehicles to the network infrastructure also eases the management of a multi-platoon scenario and the integration of other ITS services such as traffic and emergency management. Moreover, as a further benefit, a MEC-based controller provides 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) managing the actuation of 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 [32]. 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 MEC hosts that are serving the platoon.

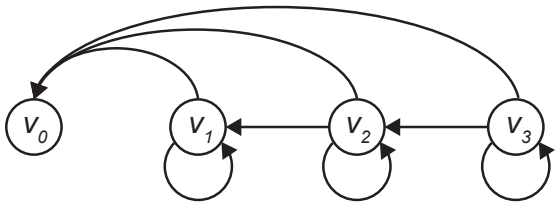


Figure 3: Dependency graph corresponding to the Leader-and predecessor-following control topology.

the computing elements where the platoon controller application can be deployed.

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

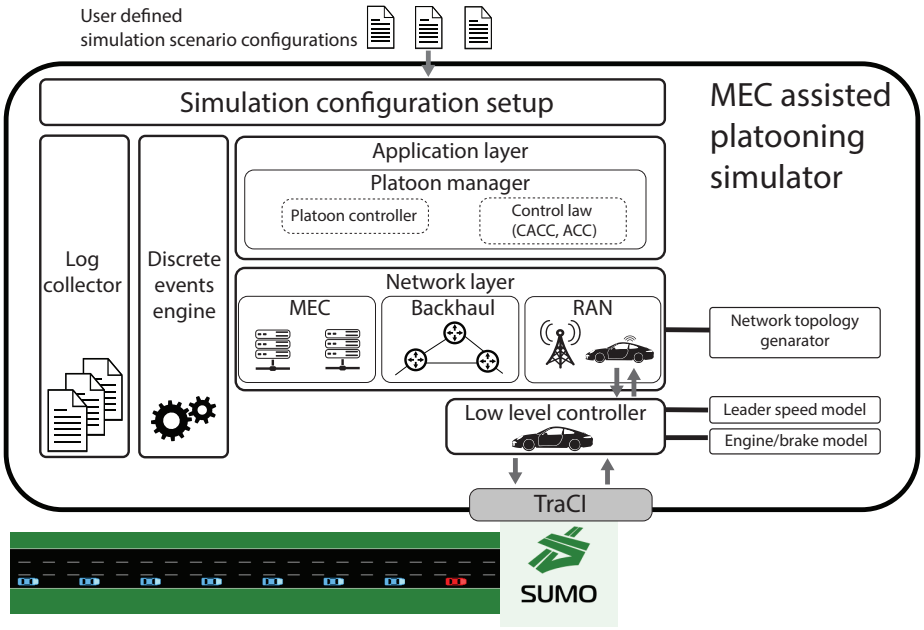


Figure 4: Platoon simulator schema

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

To evaluate our MEC-based platoon control, we developed a discrete-event Python simulation framework which incorporates the MEC application, as described in Section 4, and models network and processing delays as random variables with given distributions. The MEC framework runs on top of the standard mobility simulation tool SUMO [33], which provides a built-in basic on-board unit for reading sensors' data and controlling vehicles. The architecture of such a simulation framework is shown in Figure 4. The SUMO interaction relies upon the Traffic Control Interface (TraCI), which provides a wide set of APIs for retrieving data of simulated vehicles, e.g., to read the values of the on-board sensors, and to act on their behavior from an external controller, for instance, to control the speed of each vehicle within a platoon. The simulator has the following components:

- *Discrete events engine* is the core component of the simulation framework being in charge of scheduling, dispatching and executing simulation events.
- *Low level controller* models the OBU and is in charge of interacting with SUMO via TraCI. It interacts with two sub-modules for modeling the engine/brake system, and enforcing specific speed patterns to the platoon leader.
- *Network layer* models the three network segments relying on three sub-components: RAN, backhaul and MEC. The component RAN provides a basic network model for base stations and vehicles. The backhaul component models the data transfer through the mobile operator transport network. The *network topology generator* is a small component which supports the network layer for the creation of the network topology and the corresponding network delay. The MEC component is responsible for modeling the MEC-host capabilities and VNF instances, and it also provides a basic support for VNF replication and migration, even though this functionality is not used in this work.
- *Application layer* implements the platoon controller VNF as in Figure 2, as well as the control laws. This layer is responsible for the management of platoons and supports join/leave operations to the platoon, and approaching maneuvers.
- *Log collector* is a module that provides a uniform interface for logging to all the simulator components.
- *Simulation configuration setup* is in charge of parsing the configuration files and the parameters from CLI, to generate the simulation scenario. It also offers the opportunity to run multiple simulations in batch mode taking advantage of multi-core CPUs.

Our MEC-based approach inevitably has to cope with non negligible delays, as shown in Figure 5. 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 to prepare the IP packet to be sent over the backhaul network; T_{BH1} represents the 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

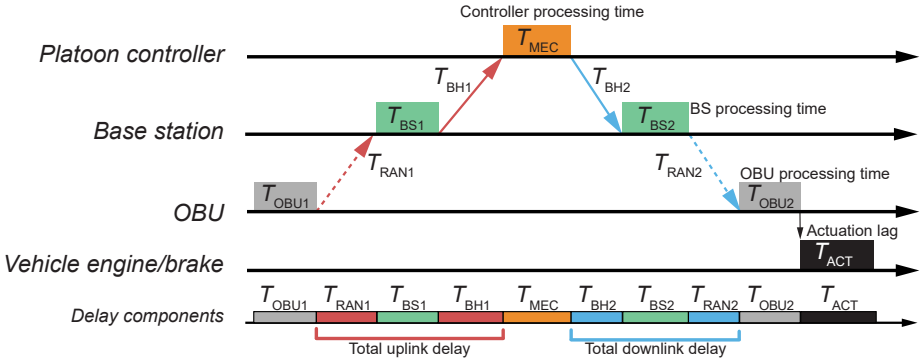


Figure 5: Delay components breakdown

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 [20, 7], 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. Simulation setup

6.1. Mobility

We simulate a highway scenario with a single platoon composed of 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 of a sinusoidal variation of the speed of the leader between two values at constant frequency over the entire simulation time. In line with [17] and [20], 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 against a continuous change of the leader speed. The real-trace pattern is extracted from the Floating Car Dataset[2], which contains GPS traces sampled every 2 seconds. Due to relatively low precision of sampling, the reconstructed speed profiles are noisy and lead to unrealistic accelerations and decelerations, especially in the case of heavy vehicles. To overcome this problem, we apply simple moving average

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

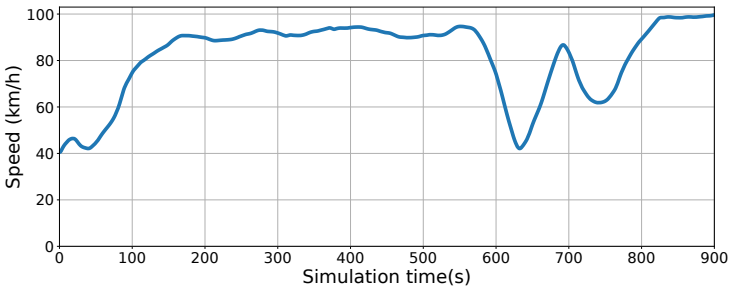


Figure 6: Speed pattern of the *real-trace*.

filter on the trace, using a sample window of 30 points, and we reduce the speed by 25 km/h, which results in a smoother speed profile suitable for comfortable assisted driving and for the engine capabilities of a wide set of vehicles [34], including large lorries. The speed profile is shown in Figure 6, it lasts 15 minutes and combines a wide range of speeds, from 40 to 100 km/h, and includes acceleration/deceleration maneuvers; 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, and by using different random generator seeds, so that confidence intervals can be computed.

6.2. Engine and control law

To simulate the actuation lag, we consider two separate cases: (i) ideal vehicle with fixed lag, independent of the speed, and (ii) vehicle equipped with realistic engine/braking system which takes into account the engine power, the drag force, the mass and inertia. For the first case, we use the approach in [17], where the electro-mechanic actuator is modeled as a low pass-filter with time constant τ equal to 0.2s and 0.17s if the vehicle is either braking or accelerating, respectively. For the realistic engine/braking system, we implement the model presented in [20] and we consider three types of vehicles: passenger car, light delivery vehicle (LDV) and heavy delivery vehicle (HDV). To highlight the differences among these vehicles, in Table 1 we report the main parameters we used in simulations[3]. The substantial differences in the capabilities of vehicles strongly impact the actuation lag component and affect the ability of vehicles to react to the controller instructions. More details about the engine characteristics are given in Appendix A.

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.

[3]Vehicle specifications are taken from https://www.cars-data.com/ for Citroen C5 and FIAT Ducato. For the Volvo truck we use the specification provided by the truck manufacturer.

Parameter	Vehicle type		
	Car (Citroen C5)	**LDV (Fiat Ducato)**	**HDV (Volvo FM)**
Max engine power	110 kW	130 kW	350 kW
Mass	1500 kg	3400 kg	40000 kg
Length	4.5 m	6.4 m	16.5 m
Drag coefficient	0.3	0.45	0.6
Section area	2.0 m^2	4.5 m^2	6.75 m^2

Table 1: Real vehicles parameters

6.3. Connectivity

To investigate the impact of latency, throughout the simulation experiments we consider that on-board sensors generate messages to be sent to the controller with a 10 Hz frequency, which is the highest frequency specified by ETSI for intelligent transportation systems [30]. We also 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 5) 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. We also consider higher delays in the range 115 to 215 ms, to simulate extreme case scenarios.
- 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. As for the uplink, we also consider higher delays in the range 115 to 215 ms to stress the system under limit conditions.
- The average processing time needed for the MEC to compute the control instructions for the platoon vehicles (component T_{MEC} in Figure 5) 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 described above considering four types of vehicle.

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, and between 300 and 500 ms for the extreme cases.

The shape of the 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.

6.4. Simulation of handovers and temporary loss of coverage

In order for a centralized platoon control to maintain string stability and target distance, updates and instructions need to be exchanged at a constant rate between vehicles and controller. When this communication experiences a temporary interruption, the stability of the entire platoon may be seriously affected. In the following, we consider two realistic sources of interruption, and evaluate, by means of simulations, their impact on the platoon.

Handovers between base stations are the first source of interruption we consider. When an ongoing data session is transferred between base stations, connectivity may be lost for a short time leaving vehicles without control. We consider four different mean values of handover time, namely 10, 50, 500 and 1000 ms, with exponential distribution. The higher values may be pessimistic with respect to those normally experienced, but enable us to analyze the platoon behavior under stressed conditions. We assume that base stations are positioned along the highway at 1 kilometer spacing. Of course, while traveling along the road, vehicles experience the handover event at different times, according to their position within the platoon.

The second source of interruptions that we consider is a RAN coverage hole, i.e., a localized loss of connectivity along a road segment of a few hundreds of meters, due, for example, to a short tunnel or a shadowing effect. Although this condition may be rare in most countries, it is worthy of analysis because it describes a condition in which all the platoon vehicles experience a communication breach at the same time, and for a longer time, thus exposing the limits of the MEC approach to platoon control. In the simulations we consider two segment lengths, 200 and 500 m.

We assume that the actuator layer controller (see Figure 1) is unaware of the temporary loss of connectivity and continues to execute the last known instruction until the link is re-established. We also assume that no emergency procedure is active on OBU, as we are interested in analyzing the behavior of platoon under extreme conditions without external interventions.

7. Performance evaluation

In this section we present quantitative metrics computed with our simulation experiments. The section is structured in three parts. First, we analyze the performance of our edge-based platooning scheme using a sinusoidal speed trace for the platoon leader, which causes extreme stress to the CACC, thus yielding worst case performance. With that, we also evaluate the impact of RAN handovers and of packet loss in the network. Second, we use a realistically

smooth speed trace to show differences with respect to the sinusoidal case, and to analyze the impact of realistic engine inertia and of limited duration connectivity blackouts, with three different classes of commercial vehicles. Third, we evaluate how our scheme scales with the size of platoons.

7.1. Sinusoidal speed variations

Sinusoidal speed profiles for the platoon leader are often used in the analysis of platooning performance, because this condition stresses the CACC. We therefore start by assessing the performance of our scheme when the platoon leader's speed over time is sinusoidal between 95 and 105 km/h, with a very short (and unrealistic) period of 2 s. In the figures shown in this subsection, the platoon size is set to 20. However, we remark that we have observed similar trends with different platoon sizes.

7.1.1. Distance error and platoon stability

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.

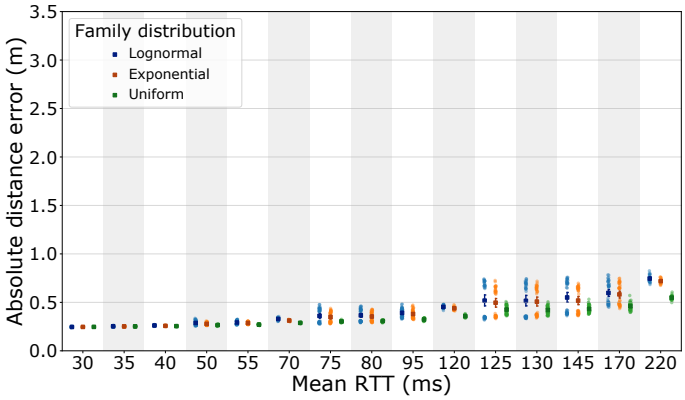
Results show that maximum errors of slightly over 3 m are made with average RTT over 120 ms, while average values of RTT below 70 ms generate maximum errors of 1.5 meters only in the case of the lognormal pdf, and the uniform distribution induces maximum errors below 1 m. Looking at 95th and 99th percentiles,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

As can be seen in Figure 8, 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 99th percentile of the absolute position error of about 1 m under a lognormal delay distribution, while allocating 110 ms to the downlink yields an average maximum absolute position error of about 60 cm, practically independently on the distribution. In general, as expected, the lognormal pdf yields largest control errors.

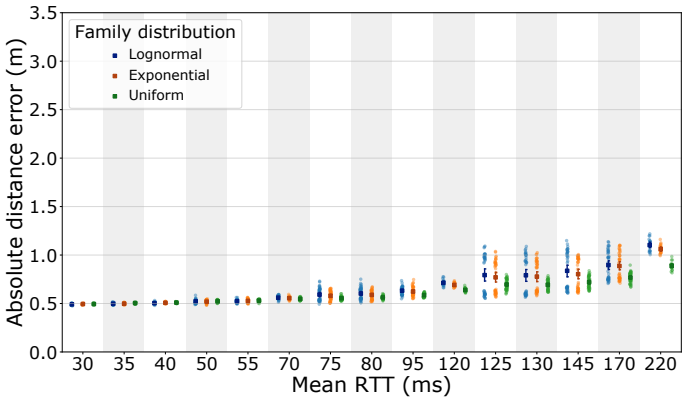
It is interesting to observe that our scheme achieves string stability: distance errors are unevenly distributed amongst vehicles within a platoon, with vehicles just behind the leader that suffer larger errors than the others. This is shown in Figure 9, which depicts the distributions of the absolute value of distance errors with respect to the vehicle position within the platoon, under different average delays with uniform distribution. The extreme values of the error decrease from almost 2 m, for the first follower, to less than 1 m for vehicles in the second half of the platoon. Moreover, the lower the RTT, the higher the platoon stability.

7.1.2. Impact of losses

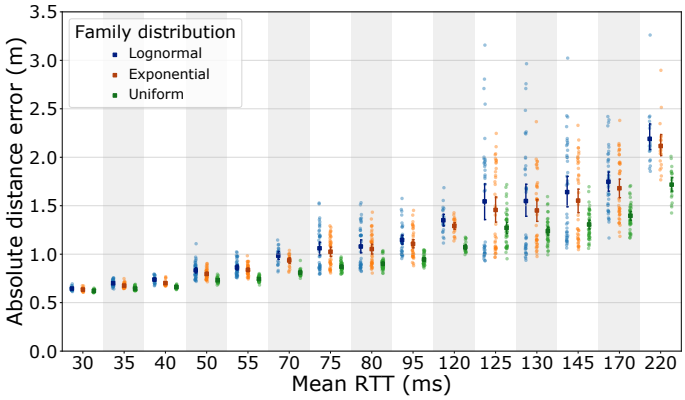
Packet loss on either the uplink or the downlink, or both, does not significantly jeopardize the performance, as can be observed in Figure 10. The figure



(a) 95th percentile

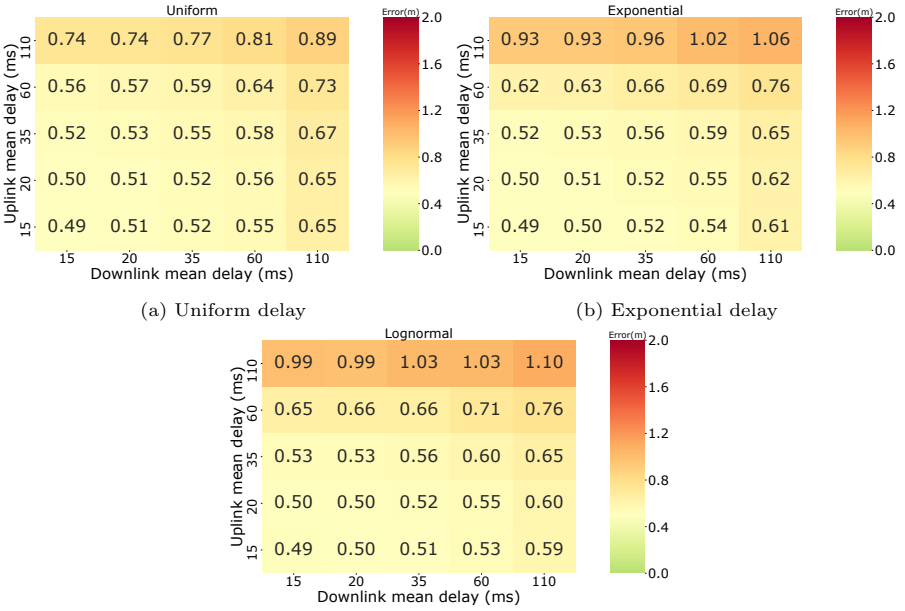


(b) 99th percentile



(c) Maximum

Figure 7: Absolute distance error for each RTT scenario with a sinusoidal mobility pattern, for an ideal vehicle's engine with actuation lag, without packet loss. Each point represents a run of simulation, the bars show 95% confidence intervals with respect to the average.



(a) Uniform delay

(b) Exponential delay

(c) Lognormal delay

Figure 8: Heatmap of the 99th percentile of the absolute distance errors for unbalanced latency scenarios with sinusoidal pattern, for an ideal vehicle's engine with actuation lag and without packet loss (mean over all simulation runs).

presents the maximum absolute distance error for various cases, with up to 2% of packet loss in uplink and downlink, which is unrealistically high in today's cellular networks. Losses considered in the experiment are uncorrelated over time and population, i.e., each packet has the same probability to be lost. For reference, results for the case without errors are included in Figure 10, and correspond to what shown in Figure 7c. Robustness to packet loss is not surprising in light of the high rate adopted for sensor data gathering, and because of the

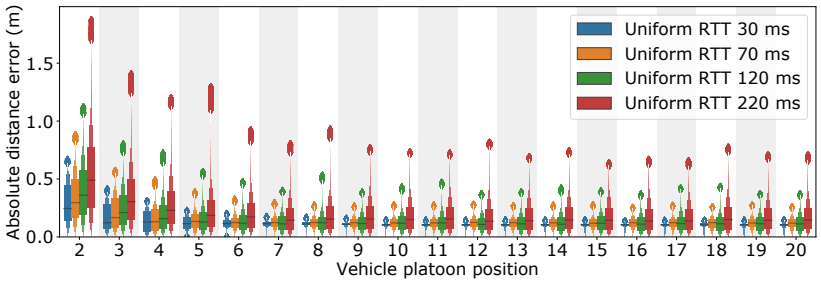


Figure 9: Boxplot of the distribution of the absolute distance error under sinusoidal mobility pattern for different vehicle positions in a platoon with 20 elements (for ideal vehicle's engines with actuation lag and without packet loss).

presence of a significant actuation lag.

To further assess the impact of losses, we consider the case of correlated losses, which could occur upon a radio handover, during which a vehicles could fail to send a few updates and miss CACC directives. Figure 11 depicts the 95th percentiles of the absolute distance error for handover duration of up to 1 s (on average, and with exponential distribution). Small values of the average handover duration have negligible impact, while long handovers can cause large error degradation, up to one order of magnitude with respect to the case without handovers (cf. Figure 7a). These results tell that the MEC-based platoon control is sufficiently robust also to correlated losses occurring in time windows of a few hundreds of milliseconds.

7.1.3. Impact of actuation lag

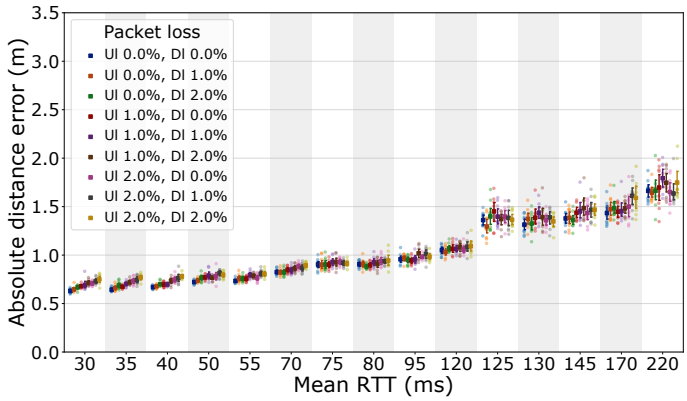
The fact that vehicle actuation lags have time constants higher than the whole chain comprising uplink and downlink packet transmission and computation at the MEC, raises a question about the relative impact of communication/computing delays and actuation lag. We thus finally explore the impact of the 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 absolute distance error, with uniform delay distribution. For ease of comparison, the figure also includes results for the case in which we do not consider the actuation lag (same as in Figure 7). We can see that the impact of lag is quite relevant, especially in the intermediate interval of RTT values, which are the ones most likely in practice. The actuation lag is detrimental for performance, although it is not critical, as its presence only adds up a few tens of centimeters to the absolute error. Furthermore, in Figure 13 we can observe that delay in the uplink has higher impact than on the downlink. The heatmaps of the figure only show the 99th percentile of the absolute distance error for uniform distribution of the delay. However, results for other distributions and for other percentiles show the same trend.

7.2. Real-trace mobility

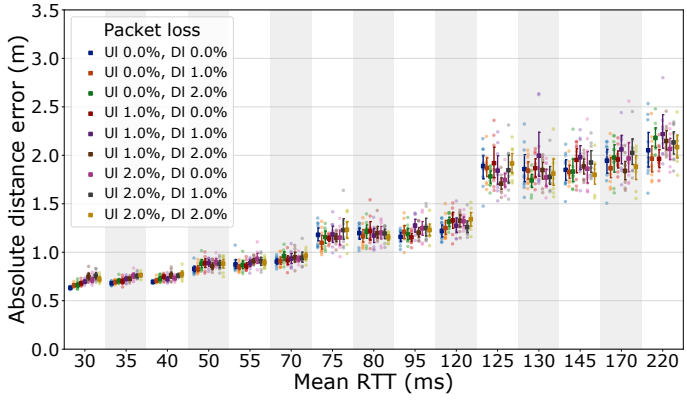
In this subsection we consider the realistically smooth platoon leader's speed trace that is shown in Figure 6 and that provides a less challenging environment with respect to the sinusoidal mobility that we considered so far. We also introduce and evaluate the impact of two realistic factors: the inertia of real engines and the possibility to encounter radio coverage holes.

7.2.1. Distance error

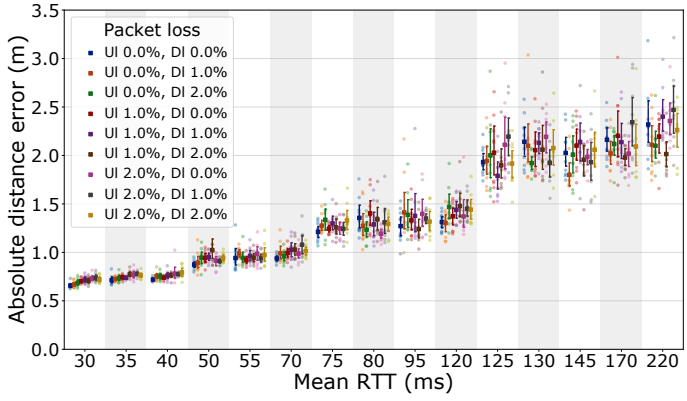
In Figure 14 we report the 99th percentile of the absolute distance error for variable RTT. Each point represents a run of simulation, and bars show 95% confidence intervals for average values. In this case we can observe distance error values up to about 30 cm (with very high RTT value), while in the sinusoidal mobility case we had about five times as much (1 m vs 20 cm, with RTT equal to 220 ms). Results for the 95th percentile of the error and for its maximum



(a) Uniform delay

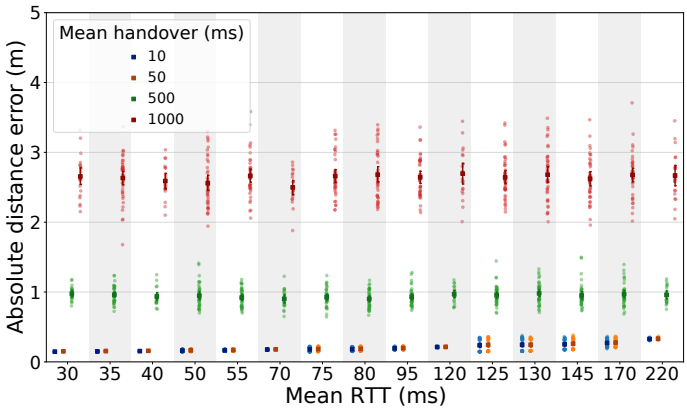


(b) Exponential delay

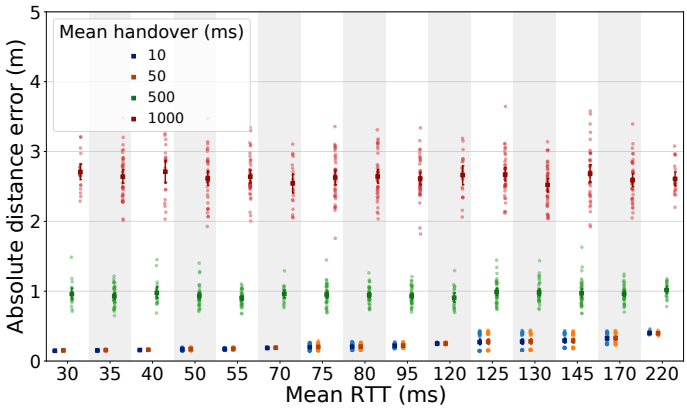


(c) Lognormal delay

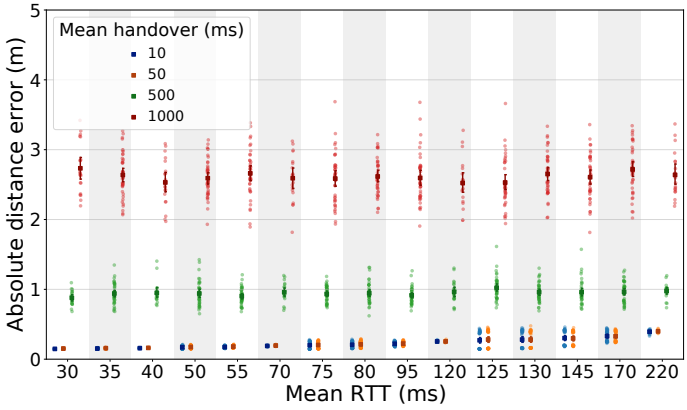
Figure 10: Maximum absolute distance error (average over all runs) versus RTT for equal average delay on uplink and downlink under sinusoidal mobility pattern, for an ideal vehicle's engine with actuation lag, with nonzero packet loss probability.



(a) Uniform delay

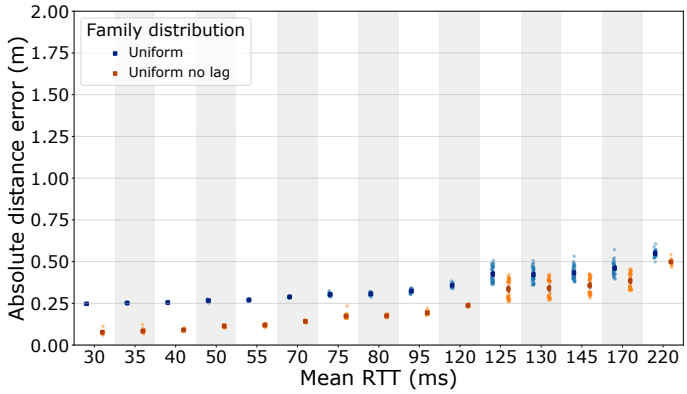


(b) Exponential delay

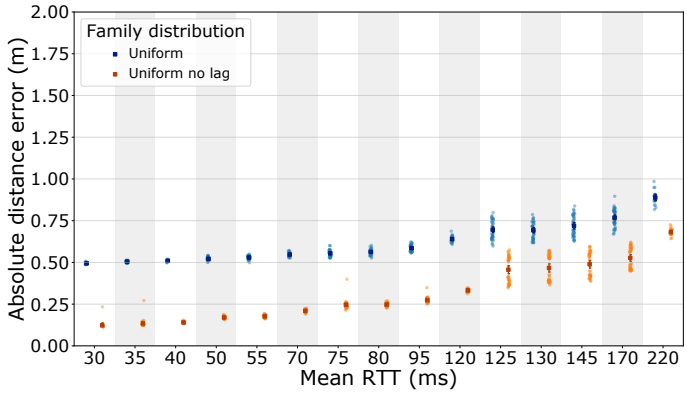


(c) Lognormal delay

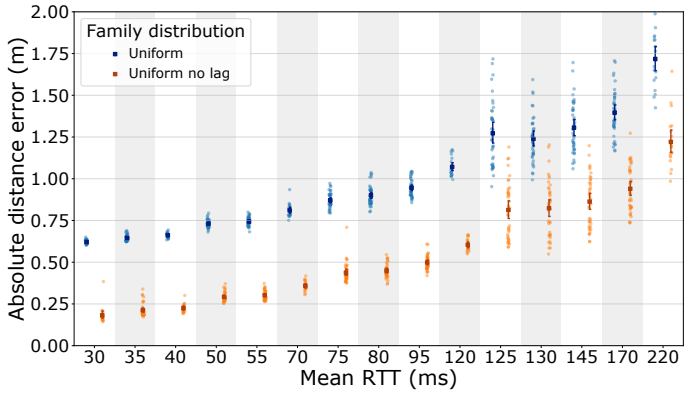
Figure 11: Impact of the handover delay on the 95th percentile of the absolute distance errors for unbalanced latency scenarios with sinusoidal pattern, for ideal vehicle's engines with actuation lag, and without packet loss (mean over all simulation runs).



(a) 95th percentile

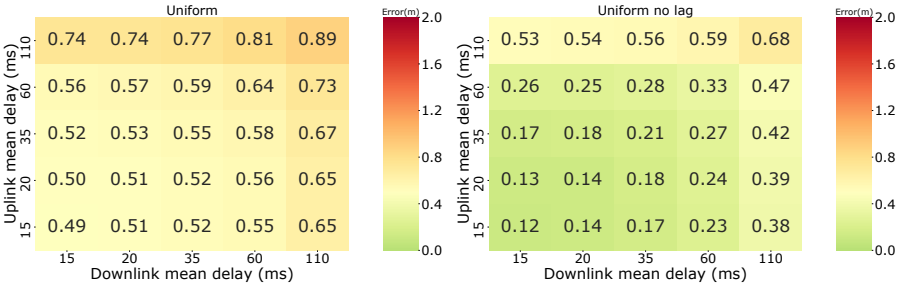


(b) 99th percentile



(c) Maximum

Figure 12: Impact of the actuation lag on the 95th and 99th percentile of the absolute distance error, and on its maximum, for each RTT scenario with sinusoidal pattern, for an ideal vehicle's engine and without packet loss. 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 sinusoidal mobility pattern, for ideal vehicle's engines, uniform RTT, and no packet loss (mean over all simulation runs).

are omitted here because they lead to similar conclusions: our results confirm that a realistically smooth speed profile yields much lower control errors than under sinusoidal mobility conditions.

7.2.2. Realistic engine

When the performance of realistic engines of cars, vans and heavy trucks, are considered together with their relevant parameters (see Table 1), the vehicle ability to react to platoon controller instructions may be highly conditioned with respect to the ideal case. In Figure 15 we report the 95th percentile of the distance error versus the RTT value, assuming a uniform distribution of delay and realistic mobility trace. We can first of all observe that in the ideal case the average distance error is significantly smaller than with sinusoidal mobility (cf. Figure 7). We can also see that the impact of the engine performance is much more important than the one of the RTT value, even if the RTT values grow very high. Finally, we can observe that cars and vans behave very similarly, and even a heavy truck engine does not lead to distance errors significantly larger than 1 m.

7.2.3. Blackouts

In Figure 16, for a 8-vehicle platoon, we report the inter-vehicle distance in meters when a short RAN blackout interrupts the data transfer from and to vehicles. We consider a short RTT value, equal to 30 ms, and we look at three different positions of the blackout event within the trace. In Figure 16(a)-(b) we select a period of quasi-constant speed (see inset), looking at blackout duration equal to 200 and 500 m, respectively. The gray area corresponds to the blackout, and the horizontal colored bands indicate the times of start and end of the blackout for vehicles in different positions in the platoon. Lines of the same colors report the distance of vehicles from the preceding one. The gray line reports the speed of the platoon leader. In Figure 16(c)-(d) we select a period of speed reduction, and in Figure 16(e)-(f) we consider a portion of the

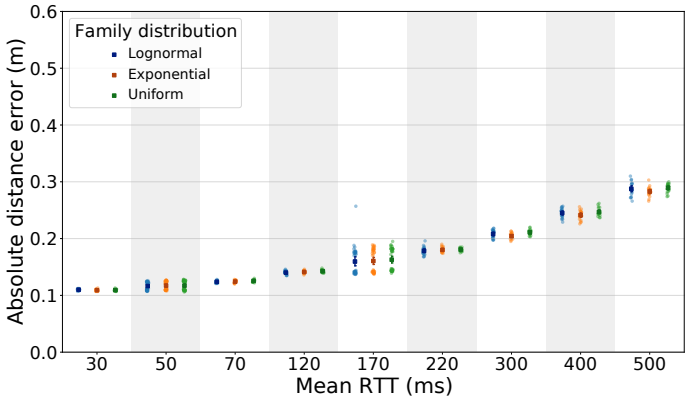


Figure 14: 99th percentile of the absolute distance error for each RTT scenario with real-trace mobility pattern for an ideal vehicle's engine with actuation lag, and without packet loss. Each point reports the result of one simulation run, while bars show 95% confidence intervals.

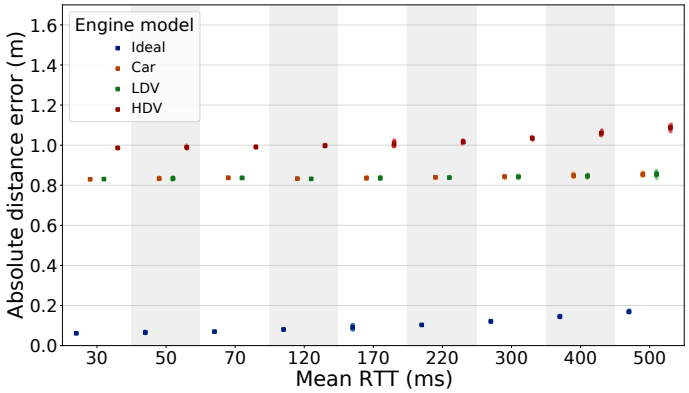
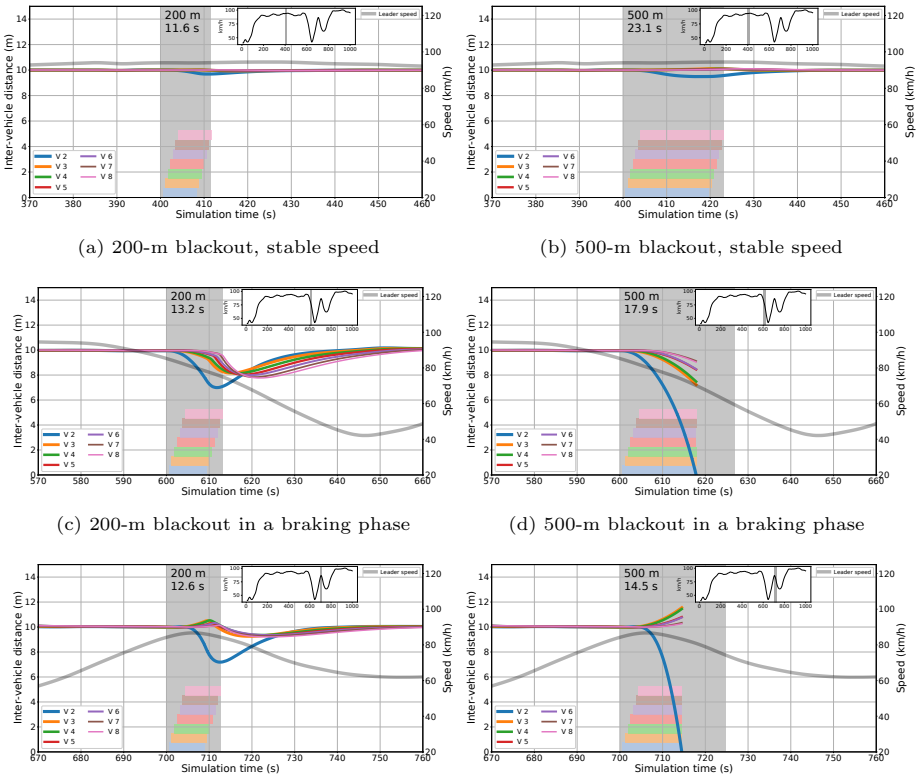


Figure 15: 95th percentile of the absolute distance error versus RTT with real-trace mobility pattern and different types of engine with actuation lag, under uniform RTT and no packet loss. Each point reports the result of one simulation run, while (barely visible) bars show 95% confidence intervals.

trace in which, before the blackout start, the platoon leader is accelerating, but during the blackout it starts reducing its speed.

Quite remarkably, we see that in the three portions of the trace, a short blackout (200 m, corresponding to little more than 10 s) is well survived by the platoon, with significant distance errors (about 3 m) experienced by the second car of the platoon when the leader's speed varies sensibly. On the contrary, a longer blackout (500 m, i.e., about 15-20 s) can only be tolerated when the platoon leader speed is almost constant (as easily forecast), but otherwise leads to collisions between the platoon leader and the second vehicle of the platoon during the blackout event (where the blue line goes to zero).

Figure 17 reports a detail for the acceleration profile of the platoon leader and two followers at the head and the tail of the platoon, respectively. The figure also reports the value of the acceleration indicated in CACC directives



(a) 200-m blackout, stable speed

(b) 500-m blackout, stable speed

(c) 200-m blackout in a braking phase

(d) 500-m blackout in a braking phase

(e) 200-m blackout, acceleration inversion

(f) 500-m blackout, acceleration inversion

Figure 16: Inter-vehicle distance variation over time with connectivity blackout (ideal engine with actuation lag, uniform RTT with average 30 ms, packet losses only occur during the blackout). The platoon leader's speed profile is indicated via a grey line, while the grey area spans from the moment the leader enters the coverage hole to the instant when the last platoon member re-connects to the network. Individual disconnection periods are indicated with colored bands. The inset figure indicates where in the real-trace the disconnection happens. Acceleration curves are interrupted when a collision of vehicles occurs.

received by the vehicles (represented as dots in the plot). The figure focuses on the case of 200-m blackout during a braking maneuver, which corresponds to the distance plots of Figure 16(c). While the leader constantly decelerates with a variable pattern, the two followers implement the last received directive, which is a flat deceleration profile during their blackout. When they eventually recover connectivity to the controller, the new received directives impose harsh braking, which the vehicles cannot actually implement immediately due to the actuation lag.

These results expose the main limitation of the MEC-based platooning approach, which lies in the need for RAN coverage, as we will discuss in a later section While short blackouts are well tolerated thanks to the redundancy in the transmission of control messages, long blackouts deprive vehicles of the in-

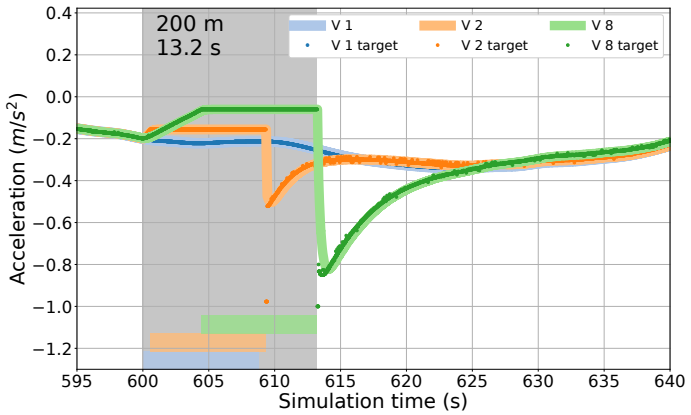


Figure 17: Acceleration profile for leader and first and last followers of the experiment of Figure 16(c). Soft-colored lines indicate vehicle acceleration, while dots indicate target accelerations (CACC directives). Blackout intervals are indicated with colored bands, while the grey area includes the interval during which at least a vehicle is not connected.

formation necessary for inter-vehicle distance control, and lead to crashes.

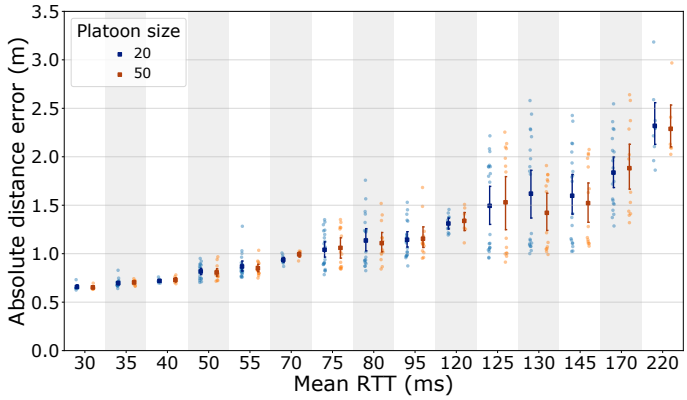
7.3. Scalability of MEC-based platooning

7.3.1. Platoon size

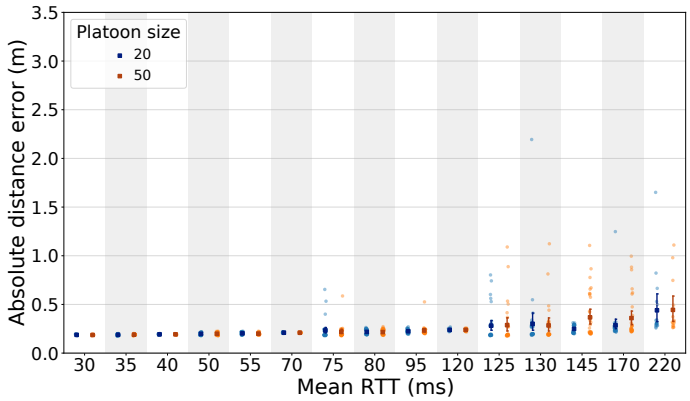
Scalability is quite a strong point of MEC-based platooning. Indeed, a PaaS approach allows the control of large platoons with no significant problem. For example, in Figure 18 we show the average maximum absolute distance error versus RTT for platoons with size 20 and 50, in the cases of lognormal delay 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 19 we report the boxplots of the distribution of the absolute distance error under sinusoidal mobility pattern and uniform RTT distribution with average 70 ms, 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 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.

7.3.2. Data rate and computation

As regards the base station bandwidth consumption, we recall that each vehicle transmits to the base station 10 messages of 200 bytes each, per second. 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



(a) Sinusoidal



(b) Real-trace pattern

Figure 18: Maximum absolute distance error (average over all runs) versus RTT for different platoon sizes, with lognormal delay distribution and ideal engine with actuation lag and no packet loss: (a) sinusoidal mobility pattern, (b) real-trace mobility pattern.

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 1 km 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

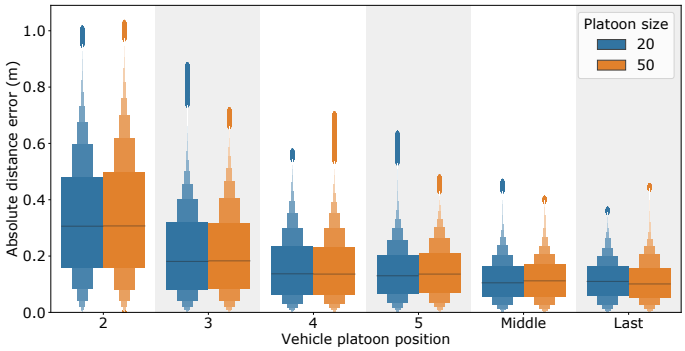


Figure 19: Boxplot of the distribution of the absolute distance error under sinusoidal mobility pattern and uniform RTT (70 ms average latency), with ideal engine with actuation lag and no network packet loss, for different platoon sizes.

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 3 GHz 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 150,000 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.

8. Discussion

In most of the described experiments, we observed no vehicle collision events, and the maximum inter-vehicle distance error was always below 4 m, except when we considered connection blackouts lasting over 10 seconds. We showed that vehicle collision events may only occur under the extreme (and rather unlikely) conditions which combine long coverage holes, and leader's speed variations. In addition, we did not account for the emergency procedures that a OBU can implement, including the obvious fallback option in which the control of the vehicle goes back to the human driver or to a simple emergency V2V control upon network disconnection. We showed that in normal conditions edge-assisted platoons can operate extremely well under the guidance of a controller located near the platoon, or even with a less close controller, up to a few hundreds of

milliseconds away from the platoon, in RTT terms. MEC-based platoons can work well with various types of vehicles and under variable speed profiles—from realistically smooth to frantic sinusoidal patterns—and with either ideal or realistic engines, lags, and platoon sizes. It can also survive critical conditions, such as those generated by short communication interruptions, due for instance to cell handovers or exceedingly high RTT values. The only critical (and rather obvious) limitation that we found is that radio coverage must be guaranteed over the whole path, with the possible exception of small blind spots.

Since the dynamics of communication between vehicles and MEC do not depend on the inter-vehicular distance, the results reported for the 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. Indeed, 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 [35], 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 guaranteed if the platoon controller runs in the cloud, since no less than 150 ms RTT can be achieved as of today, both due to the distance of cloud resources from vehicles and to the processing time required on remote shared servers [9].

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 distances. 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 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). 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 [36]. 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, under the harsher driving conditions we have tested, which is in line with 3GPP recommendations as per QCI-75 and QCI-79. With smoother realistic cases, much higher RTT values can be however tolerated as well. We remark that delays compatible with safety in the most demanding driving circumstances and with real engine's limitations are only feasible in case of running the platoon controller either on the platoon itself or at the MEC, while cloud resources might be 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, without requiring 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 [37] (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).

Finally, it is important to observe that transmitting CACC messages at 10 Hz frequency is a key feature to obtain robustness to transmission errors and to survive short disconnection periods or variations in the link quality due to fading or reduced SINR (signal to interference and noise ratio) close to cell borders. Indeed, assuming a message loss probability equal to p and independent losses, the probability of receiving no message for one second equals p^{10}, which remains below 10^{-3} (resp. 10^{-6}) even for quite unrealistic message loss probability values, up to 0.5 (resp., 0.25). We observed that a 1 s interval with no updates is well tolerated by the system in Fig. 11 when discussing handovers. As a consequence, update frequencies close to the highest value specified in [30] (equal to 10 Hz) seem quite a wise choice to incorporate redundancy in the system and achieve robustness to different system conditions.

Variations in the data rate due to changing SINR, hence to the use of different modulation formats, are not a critical issue, since the amount of data to be exchanged for platoon control is small, although the data rates affect the size and number of platoons that can be controlled in parallel from the network. Some attention to this aspect should however be paid in case of very long platoons spanning multiple cells or multiple platoons coordinating with each other, as studied in [37].

As we already observed, the main weakness of the MEC-based platoon control is in the possibility of loss of RAN coverage, either due to coverage holes,

or because of power blackouts or other events that produce periods of outage of portions of the RAN. We have observed that in this case the platoon remains reasonably stable if the leader speed remains constant, but few tens of seconds of outage can be sufficient to produce a crash when the platoon leader accelerates before the outage, but starts braking during the outage. In order to survive these conditions, an emergency V2V system can be very useful to at least control the distance from the preceding vehicle in case of failure of the MEC-based control. This leads to the consideration that optimal safety could be achieved by a hybrid system integrating MEC and V2V in the platoon control.

9. Conclusions

In this article we have investigated the feasibility of a MEC-based centralized control of platoons of vehicles, exploring the impact of the RAN latency distributions, of time-uncorrelated and correlated packet loss probability introduced by the uplink and downlink transmissions, as well as of engine characteristics and actuation lags, and ideal vs realistic mobility profiles.

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. A centralized controller running at the edge of the cellular network is actually very robust to latency and packet losses, and its complexity scales very well. The approach can rely 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. However, as we have shown in the article, considering realistic characteristics of vehicles and mobility is paramount to correctly design the service.

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] M. Williams, Prometheus-the european research programme for optimising the road transport system in europe, in: IEE Colloquium on Driver Information, IET, 1988, pp. 1–1.

[2] D. Jia, K. Lu, J. Wang, X. Zhang, X. Shen, A survey on platoon-based vehicular cyber-physical systems, IEEE Communications Surveys Tutorials 18 (1) (2016) 263–284. doi:10.1109/COMST.2015.2410831.

[3] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, D. Sabella, 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.

[4] Y. Zheng, S. Eben Li, J. Wang, D. Cao, K. Li, Stability and scalability of homogeneous vehicular platoon: Study on the influence of information flow topologies, IEEE Transactions on Intelligent Transportation Systems 17 (1) (2016) 14–26. doi:10.1109/TITS.2015.2402153.

[5] T. Zeng, O. Semiari, W. Saad, M. Bennis, Joint communication and control for wireless autonomous vehicular platoon systems, IEEE Trans. on Communications 67 (11) (2019) 7907–7922. doi:10.1109/TCOMM.2019.2931583.

[6] G. Cecchini, A. Bazzi, B. M. Masini, A. Zanella, 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), 2017, pp. 109–114. doi:10.1109/VNC.2017.8275637.

[7] F. Dressler, F. Klingler, M. Segata, R. Lo Cigno, Cooperative driving and the tactile internet, Proceedings of the IEEE 107 (2) (2019) 436–446. doi:10.1109/JPROC.2018.2863026.

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

[9] B. P. Rimal, D. Pham Van, M. Maier, Mobile-edge computing versus centralized cloud computing over a converged fiwi access network, IEEE Transactions on Network and Service Management 14 (3) (2017) 498–513. doi:10.1109/TNSM.2017.2706085.

[10] A. Virdis, G. Nardini, G. Stea, A framework for MEC-enabled platooning, in: IEEE Wireless Communications and Networking Conference Workshop (WCNCW), IEEE, 2019, pp. 1–6.

[11] X. Fan, T. Cui, C. Cao, Q. Chen, K. S. Kwak, Minimum-cost offloading for collaborative task execution of MEC-assisted platooning, Sensors 19 (4) (2019) 847.

[12] Y. Hu, T. Cui, X. Huang, Q. Chen, Task offloading based on Lyapunov optimization for MEC-assisted platooning, in: International Conference on Wireless Communications and Signal Processing (WCSP), 2019, pp. 1–5. doi:10.1109/WCSP.2019.8928035.

[13] S. Dabbene, C. Lehmann, C. Campolo, A. Molinaro, F. H. P. Fitzek, A mec-assisted vehicle platooning control through docker containers, in: 2020 IEEE 3rd Connected and Automated Vehicles Symposium (CAVS), 2020, pp. 1–6. doi:10.1109/CAVS51000.2020.9334658.

[14] A. Lekidis, F. Bouali, C-v2x network slicing framework for 5g-enabled vehicle platooning applications, in: VTC2021-Spring Workshops, IEEE, United States, 2021, pp. (In–Press), 93rd Vehicular Technology Conference, VTC2021-Spring ; Conference date: 25-04-2021 Through 28-04-2021.

[15] C. Chen, J. Jiang, N. Lv, S. Li, An intelligent path planning scheme of autonomous vehicles platoon using deep reinforcement learning on network edge, IEEE Access 8 (2020) 99059–99069. doi:10.1109/ACCESS.2020.2998015.

[16] R. Rajamani, Vehicle dynamics and control, Springer, 2012, Ch. 7.

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

[18] R. Rajamani, Han-Shue Tan, Boon Kait Law, Wei-Bin Zhang, Demonstration of integrated longitudinal and lateral control for the operation of automated vehicles in platoons, IEEE Transactions on Control Systems Technology 8 (4) (2000) 695–708. doi:10.1109/87.852914.

[19] J. Ploeg, B. Scheepers, E. van Nunen, N. van de Wouw, H. Nijmeijer, Design and experimental evaluation of cooperative adaptive cruise control (2011) 260–265.

[20] S. Santini, A. Salvi, A. S. Valente, A. Pescapé, M. Segata, R. Lo Cigno, A consensus-based approach for platooning with intervehicular communications and its validation in realistic scenarios, IEEE Trans. on Vehicular Technology 66 (3) (2017) 1985–1999. doi:10.1109/TVT.2016.2585018.

[21] Y. Ma, Z. Li, R. Malekian, R. Zhang, X. Song, M. A. Sotelo, Hierarchical fuzzy logic-based variable structure control for vehicles platooning, IEEE Transactions on Intelligent Transportation Systems 20 (4) (2019) 1329–1340. doi:10.1109/TITS.2018.2846198.

[22] F. Navas, V. Milanés, C. Flores, F. Nashashibi, Multi-model adaptive control for cacc applications, IEEE Transactions on Intelligent Transportation Systems 22 (2) (2021) 1206–1216. doi:10.1109/TITS.2020.2964320.

[23] V. Vegamoor, S. Yan, S. Rathinam, S. Darbha, Mobility and safety benefits of connectivity in cacc vehicle strings, in: 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC), 2020, pp. 1–6. doi:10.1109/ITSC45102.2020.9294203.

[24] W. Hao, L. Liu, X. Yang, Y. Li, Y. J. Byon, Reducing cacc platoon disturbances caused by state jitters by combining two stages driving state recognition with multiple platoons' strategies and risk prediction, IEEE Transactions on Intelligent Transportation Systems (2020) 1–11doi:10.1109/TITS.2020.3033436.

[25] B. Tian, G. Wang, Z. Xu, Y. Zhang, X. Zhao, Communication delay compensation for string stability of cacc system using lstm prediction, Vehicular Communications 29 (2021) 100333. doi:https://doi.org/10.1016/j.vehcom.2021.100333.

[26] S. Sadraddini, S. Sivaranjani, V. Gupta, C. Belta, Provably safe cruise control of vehicular platoons, IEEE Control Systems Letters 1 (2) (2017) 262–267. doi:10.1109/LCSYS.2017.2713772.

[27] N. Chen, M. Wang, T. Alkim, B. van Arem, A robust longitudinal control strategy of platoons under model uncertainties and time delays, Journal of Advanced Transportation 2018 (2018).

[28] O. E. Gungor, I. L. Al-Qadi, All for one: Centralized optimization of truck platoons to improve roadway infrastructure sustainability, Transportation Research Part C: Emerging Technologies 114 (2020) 84–98. doi:https://doi.org/10.1016/j.trc.2020.02.002.

[29] C. Quadri, V. Mancuso, M. Ajmone Marsan, G. P. Rossi, Platooning on the edge, in: ACM MSWiM, 2020. doi:10.1145/3416010.3423220.
URL `https://doi.org/10.1145/3416010.3423220`

[30] Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 2: Specification of Cooperative Awareness Basic Service, Tech. rep., ETSI Std. EN 302 637-2 V1.3.2 (Oct. 2014).

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

[32] J. Martín-Pérez, L. Cominardi, C. J. Bernardos, A. de la Oliva, A. Azcorra, Modeling mobile edge computing deployments for low latency multimedia services, IEEE Transactions on Broadcasting 65 (2) (2019) 464–474. doi:10.1109/TBC.2019.2901406.

[33] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, E. Wießner, Microscopic traffic simulation using SUMO, in: IEEE International Conference on Intelligent Transportation Systems, 2018.

[34] S. Moon, I. Moon, K. Yi, Design, tuning, and evaluation of a full-range adaptive cruise control system with collision avoidance, Control Engineering Practice 17 (4) (2009) 442–455. doi:https://doi.org/10.1016/j.conengprac.2008.09.006.

[35] TS 23.203 (Rel-15); Policy and charging control architecture, Tech. rep., 3GPP (October 2019).

[36] C. Bonnet, H. Fritz, Fuel consumption reduction in a platoon: Experimental results with two electronically coupled trucks at close spacing, Tech. rep., SAE Technical Paper (2000).

[37] C. Quadri, V. Mancuso, V. Cislaghi, M. Ajmone Marsan, G. P. Rossi, From PLATO to platoons, in: IEEE MedComNet, 2021.

Appendix A. Actuation lag

We consider two types of vehicle: (i) an ideal vehicle with fixed actuation lag which is able of performing any acceleration instruction independently of the current speed, and (ii) a vehicle equipped with a realistic engine/braking system, and affected by air drag inertia due to the mass. For the first type of vehicle, we adopt the approach in [16, 17] and we model the actuation lag as a first order low-pass filter:

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

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

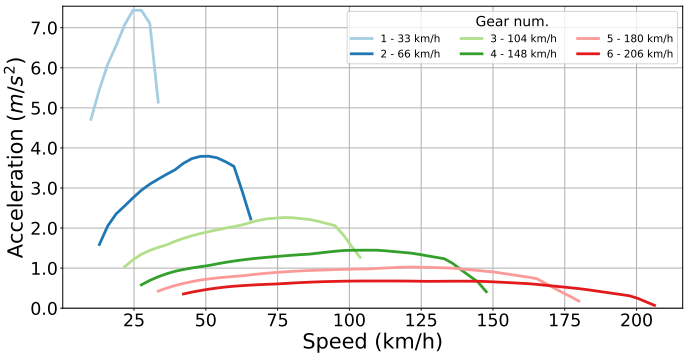
where $\ddot{x}[n+1]$ is the next acceleration at the $n+1$-st simulation step which depends on the desired acceleration $\ddot{x}_{des}[n+1]$, computed by the OBU's coordination layer, and on the current vehicle acceleration $\ddot{x}[n]$. Δ_t represents the time-step of the discrete event system operations,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

As for the realistic engine we use the approach in Santini *et al.* [20] (Sec. V), which takes into account the different capabilities of the engine at different speeds, i.e., the maximum possible acceleration as a function of speed and gear. By applying this equation to the three vehicles considered in this work, we obtain the maximum acceleration values versus speed/gear which are shown in Figure A.20. These results are obtained by modelling the actuation lag according to equation (46) in [20], which considers the engine rotation speed and the number of cylinders to determine the actual value of τ to be used in (A.1).

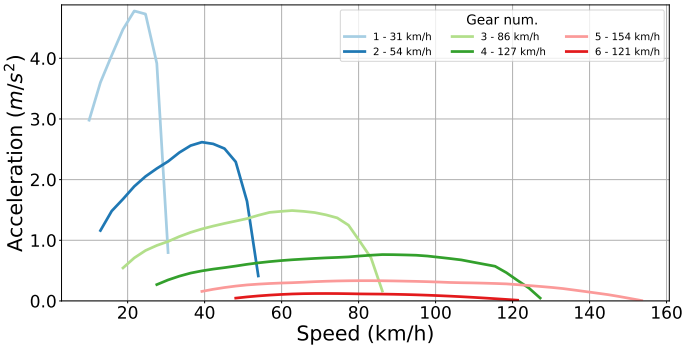
Appendix B. String stability and CACC

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 [16]. Formally, the string stability is defined as follows:

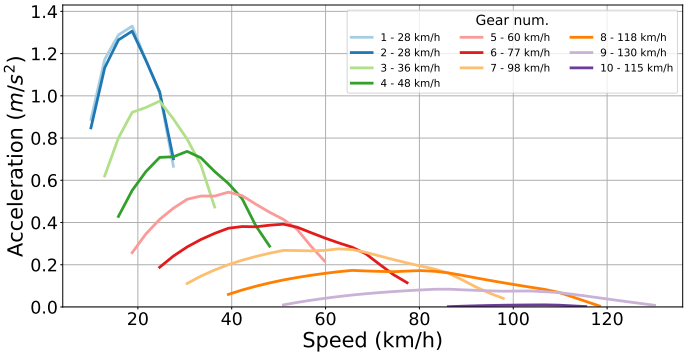
$$\left\| \frac{e_i(t)}{e_{i-1}(t)} \right\|_\infty \leq 1 \quad \forall i = 2 \cdots n \tag{B.1}$$



(a) Car (Citroen C5)



(b) LDV (Fiat Ducato L4H2)



(c) HDV (Volvo FM)

Figure A.20: Maximum acceleration as function of speed and gear.

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 [16]) is

$$\ddot{x}_{i_des} = \alpha_1 \ddot{x}_{i-1} + \alpha_2 \ddot{x}_0 + \alpha_3 \dot{\varepsilon}_i + \alpha_4 (\dot{x}_i - \dot{x}_0) + \alpha_5 \varepsilon_i \tag{B.2}$$

$$\dot{\varepsilon}_i = \dot{x}_i - \dot{x}_{i-1} \tag{B.3}$$

$$\varepsilon_i = x_i - x_{i-1} + l_{i-1} + d_{des} \tag{B.4}$$

$$\alpha_1 = 1 - C_1 \tag{B.5}$$

$$\alpha_2 = C_1 \tag{B.6}$$

$$\alpha_3 = -\left(2\xi - C_1\left(\xi + \sqrt{\xi^2 - 1}\right)\right)\omega_n \tag{B.7}$$

$$\alpha_4 = -C_1\left(\xi + \sqrt{\xi^2 - 1}\right)\omega_n \tag{B.8}$$

$$\alpha_5 = -\omega_n^2 \tag{B.9}$$

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 with respect to 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.