On the Impact of Communication and Computing Delays on MEC-Enabled Cooperative Maneuvers

Michele Segata*†, Anish Shastri*, Andrea Piccin*, Ioan Gabriel Duta*, Paolo Casari*†
*DISI – University of Trento, Italy †CNIT, Italy

***Abstract*— Cooperative driving maneuvers can improve the efficiency of road transportation by actively coordinating vehicles to maximize traffic flow and minimize fuel consumption. Different from platooning, where direct vehicle-to-vehicle (V2V) communication might represent the best option, maneuvers are usually local and time-bound. In such situations, an approach exploiting multi-access edge computing (MEC) can yield the benefits of a centralized solution with reduced delay, as control and coordination traffic does not need to flow through the network core. Moreover, deploying maneuvers as MEC applications effectively realizes a new paradigm that we call cooperative driving-as-a-service (CDaaS), where the resources to manage and control maneuvers are flexibly hosted and managed by the same network that connects the vehicles. As the MEC is a shared computing infrastructure not dedicated to traffic optimization, its use raises concerns related to delays: excessive delays (e.g., due to server overload) could compromise the safety and the efficiency of the maneuvers. Moreover, to keep delay contained, applications may need to be migrated to different servers. In this work, we present the CDaaS concept and implement two maneuvers as MEC applications: a cooperative overtake and an intersection merge. Via an open-source simulator, we then study the impact of communication delays on their performance. Our results show that the impact of the delay is actually maneuver-dependent, hence migration decisions should depend on the maneuvers themselves, besides other factors such as the network delay and server load.**

***Index Terms*—Platooning, MEC, cooperative maneuvers, cooperative driving**

I. Introduction

Cooperative driving (CD) promises to improve the safety and the efficiency of future transportation systems by interconnecting vehicles for cooperation and coordination [1]. CD is a fundamental component of the broader cooperative, connected and automated mobility (CCAM), where vehicles make decisions not just based on data from their own on-board sensors, but rather compute the best actions in a cooperative manner by sharing intentions and sensed data [2].

The efforts towards CCAM come from different directions, which include both academia and industry, not to mention the efforts for integrating vehicular communications into cellular standards, such as 5G Cellular V2X (C-V2X) [3]. Realizing CCAM entails developing advanced cooperative maneuvers. Different from basic safety applications such as electronic emergency brake light (EEBL) [4], such maneuvers require complex protocols to properly coordinate vehicles, and still face a number of open challenges [2]. One of these challenges is how to involve the infrastructure, as the current literature mainly focuses on maneuvers enabled solely by direct vehicle-to-vehicle (V2V) communication.

A promising approach in this respect is the use of multi-access edge computing (MEC) [2], [5], [6]. The MEC paradigm can bring several benefits to cooperative maneuvers as well various vehicle-to-everything (V2X) use cases [7]. Besides the fact that centralizing the maneuver's intelligence can ease gathering data from vehicles and compute the best driving actions, MEC can also help implement collective perception (CP) and deal with mixed traffic.

In this work, we present the vision of SELF4COOP, an Italian government-funded project, which aims at exploring the possibilities offered by MEC for CD. In particular, SELF4COOP aims at developing cooperative maneuvers as MEC applications, making them available on demand to nearby vehicles. SELF4COOP defines this paradigm as cooperative driving-as-a-service (CDaaS), i.e., offering CD services as applications to vehicles that request them from the infrastructure.

This work presents the initial efforts of SELF4COOP towards the realization of a CDaaS architecture. The key contributions of this work are as follows: first, we describe the objectives of SELF4COOP in detail, describing the ideas proposed therein; second, we formally define two of the proposed cooperative driving maneuvers, namely *cooperative overtaking* and *cooperative intersection merging*. Such maneuvers have already been defined in the literature [2]: here, the goal is to propose a simple way to integrate them in the MEC infrastructure. We implement them as MEC applications, show the benefits that they can bring to road traffic, and study the impact of communication and computation delays. As an additional contribution, the paper implements the maneuvers by federating PLEXE and Simu5G. To do this, we implement new Simu5G features such as shared MEC applications, and release the software as open-source to foster further research work. Our results show the feasibility of MEC-based cooperative maneuver management, highlight some limitations, and pave the road towards a CDaaS-based road transportation system.

II. Background and Related Work

Most of the existing literature on cooperative driving maneuvers leverages V2V communications. For example, in [8], the authors propose a cooperative communication-based control strategy to implement bidirectional overtaking in a mixed-traffic setting involving both connected autonomous vehicles (CAVs) and human-driven vehicles. The authors of [9] propose a cooperative algorithm to control overtaking for platoons on freeways, without the need to disassemble or reassemble the platoons. Here, the control directives are computed and distributed by the platoon leader. A cooperative maneuver planning framework is explored in [10], where mixed vehicles (both sensor-rich vehicles and legacy) are clustered to form platoons depending on the routes at the signal-controlled intersections. From the literature, we can observe that cooperative driving applications that exploit V2V communications often offload the computation tasks to a single vehicle. This often leads to delay in inter-vehicular communication, especially when the tail-end vehicles of the platoon are very far from the platoon leader, thus triggering string instability scenario.

Recently, the MEC paradigm has emerged as front-runner technology for offloading such computational tasks to a centralized server [11]. Yet, MEC servers are shared infrastructure, and can be used safely only when low response times and high reliability are guaranteed. These conditions are explored in [12], where MEC responsiveness is contrasted to the stringent response time and reliability requirements of different CAV applications.

MEC-based centralized solutions have been considered for cooperative driving. For example, to regulate traffic at intersections, Farina *et al.* proposed to implement a first-in-first-scheduled (FIFS) algorithm on a MEC controller enabling vehicles to safely cross unregulated intersections [13]. In [14], the authors incorporate edge computing in their traffic flow optimization architecture for real-world assisted driving systems. The feasibility of the MEC paradigm for platoon control has been studied in [15]. In [5], the authors explore intelligent platooning controller deployed at the network edge and propose a context-aware Q-learning algorithm to dynamically migrate the controller among available MEC hosts. They also propose an asynchronous shared learning scheme to expedite the convergence of migration policies.

Nardini *et al.* developed a simulator that integrates the 5G New Radio (NR) ecosystem with the ETSI-standardized MEC systems [16], providing a complete end-to-end framework for 5G-based MEC. This simulator was later used to demonstrate the platooning-as-a-service (PlaaS) framework, providing services such as discovery and dynamic join/leave, in a multi-operator environment [6].

The above review suggests that the use of MEC in cooperative driving scenarios is promising, but has not been explored in sufficient depth. In particular, the impact of typical MEC-related issues such as time-varying processing delays and migration on cooperative maneuvers requires detailed characterization. In the following, we take some steps in this direction.

III. The CDaaS vision of SELF4COOP

We believe that future CAVs have the potential to not just reduce the issues of transportation and its footprint on the environment, but rather to turn the problem into a key instrument of its own solution. As CAVs will progressively take up the market and become pervasive, they can change mobility as we perceive it today, and transform into the nervous system of future connected cities. CAVs are connected together, with the roadside transportation infrastructure, and to the Internet.

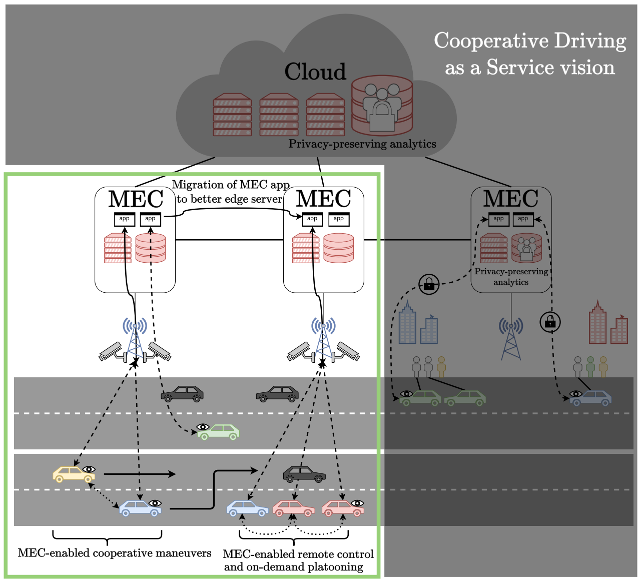


Figure 1. SELF4COOP vision of CDaaS.

These features enable traffic sensing and prediction as the cars travel, and safe traffic optimization through densely populated areas. By allowing a global management of sharing policies and incentives, CAVs will help reduce the actual number of vehicles in a city by up to 40 % [17]. A shared autonomous transportation infrastructure will make car ownership less necessary, reducing per capita costs. Moreover, we envision that CAVs will support a new mobility paradigm, which we term cooperative driving-as-a-service (CDaaS). By communicating with the CDaaS infrastructure, both users and vehicles will request mobility services provided by intelligence implemented at different scales, such as computing the optimal assignment of passengers to vehicles (global scale), while enabling vehicles to perform advanced cooperative maneuvers and optimize the traffic flow (local scale).

With reference to Fig. 1, SELF4COOP focuses on the development of fundamental building blocks that will be required to realize the CDaaS vision. These aspects are highlighted in the bright area of the picture, as opposed to the shaded area, which contains components whose development falls outside the scope of the project (including, e.g., securing communications, guaranteeing privacy, defining business models, etc.). In particular, SELF4COOP aims at tackling the challenges behind a subset of services provided by CDaaS, namely advanced cooperative driving maneuvers. These include the seamless integration of heterogeneous communication technologies, the optimal placement for MEC applications used to compute trajectories, the cooperative perception of the environment (including standard human-driven vehicles), and CAV control.

As anticipated in Sect. I, the core of the CDaaS paradigm is in the use of MEC applications to implement and offer cooperative maneuvers on demand. The MEC architecture naturally lends itself to some specific CD tasks. By being

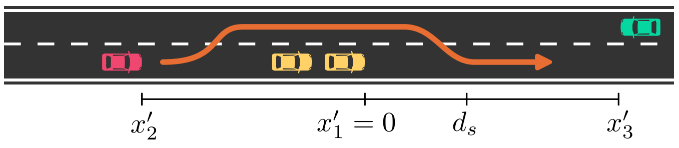


Figure 2. Overtake maneuver scenario.

centralized, it simplifies the collection of data, the computing tasks, and the redistribution of driving actions. However, different from a cloud system, a MEC is local: the application remains close to the users, thus minimizing communication delays and avoiding the uncertainties of data traffic routing through the core network or the Internet.

Unlike the work in [6], which presents open-source ETSI MEC framework and demonstrates basic platooning operations as MEC services, we provide a first design and implementation of MEC-based CD maneuvers without the aim of finding the best possible strategy, but rather with the idea of showing their feasibility as MEC applications and analyzing their potential limitations. In particular, we will consider a cooperative overtaking maneuver and a cooperative intersection merge maneuver, studying the impact that communication and computing delays have on them.

A. Cooperative overtaking maneuver

We start by describing the cooperative overtake maneuver. We consider the scenario in Fig. 2, i.e., a two-way road where a vehicle (red) wants to overtake a slower group of cars in front of it (yellow) with traffic approaching from the opposite direction (green). In most cases, the overtaking vehicle would decelerate to match the pace of the preceding vehicles and wait for the opposite traffic to clear the lane. This clearly ensures safety, but it can result in a waste of time and fuel due to unnecessary deceleration and acceleration. Instead, cooperation would make it possible to start overtaking when there is already sufficient room, or the necessary spacing can be created by slightly slowing down incoming traffic.

In what follows, we use P_1 to refer to the slow group of cars moving west-to-east, P_2 to refer to the overtaking vehicle, and P_3 to refer to the vehicle in the opposite direction. Moreover, we denote the position of the head of each vehicle or group of vehicles as x_i.

To compute the timings required for the maneuver, we first consider just the overtaking vehicle and the vehicles to be overtaken, and assume the latter to be the reference of our coordinate system. From now on, symbols with the "prime" superscript $'$ will refer to variables in a frame of reference relative to the overtaken vehicles, i.e., assuming the speeds to be $v_1' = 0\,\mathrm{m/s}$ and $v_2' = v_2 - v_1$. Finally, the origin of the longitudinal position is the front bumper of the first vehicle in P_1, corresponding to $x_1' = 0\,\mathrm{m}$.

The goal is to longitudinally cover the distance x_o' necessary to overtake the group of slower vehicles, including a proper safety margin d_s. To do so, the overtaking vehicle will start from its initial position $x_2' = d_s - x_o'$, with its initial speed v_2', and will accelerate with a certain rate a_2 until it reaches the desired overtaking speed v_o'. Depending on the initial speed and the distance to cover, the overtake might be completed before or after reaching v_o'. In the latter case, we assume the vehicle to complete the overtake at constant speed. Under these conditions, the time necessary to complete the overtake is

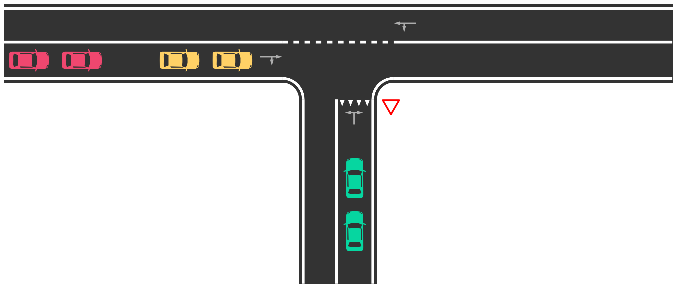


Figure 3. T intersection maneuver scenario.

$$t_o = \begin{cases} \frac{-v_2' - \sqrt{{v_o'}^2 + 2a_2 x_o'}}{a_2} & \text{if } x_o' < \frac{{v_o'}^2 - {v_2'}^2}{2a_2}, \\ \frac{v_o' - v_2'}{a_2} + \frac{x_o' - \frac{{v_o'}^2 - {v_2'}^2}{2a_2}}{v_o'} & \text{otherwise.} \end{cases} \tag{1}$$

In Eq. (1), $\frac{{v_o'}^2 - {v_2'}^2}{2a_2}$ is the space covered by the overtaking vehicle while accelerating to reach v_o'. Hence, the first line refers to the overtaking maneuver being completed before reaching v_o', whereas the second line covers the opposite case.

The incoming vehicle, instead, will start from its initial position x_3' traveling at a speed $v_3' = v_3 - v_1$ relative to P_1, with v_3 being a negative velocity. Under these conditions and assuming a constant speed, the final position of P_3 when the overtaking maneuver is completed is simply $x_3' + v_3' \cdot t_o$. To check whether there can be a collision, we can compare the position corresponding to the end of the overtake (d_s) and the final position of P_3, i.e.,

$$d_s + \epsilon \overset{?}{<} x_3' + v_3' \cdot t_o, \tag{2}$$

where ϵ is a quantity to account for measurement errors and driving comfort. If Eq. (2) is satisfied, then the maneuver can be safely completed without changes in the dynamics of vehicles.

In case Eq. (2) is not satisfied, we can still avoid slowing down P_2 to wait for P_3 to clear the overtaking lane. To achieve this, we can slow down P_3 until P_2 completes the maneuver. Formally, we simply consider a slower speed ρv_3, with $0 < \rho < 1$ being an acceptable slow-down factor. Assuming P_3 decelerates at a rate a_3, it will take a time $t_d = \frac{v_3(1-\rho)}{a_3}$ for it to decelerate down to ρv_3, so Eq. (2) becomes

$$d_s + \epsilon \overset{?}{<} x_3' + \underbrace{\frac{v_3 + \rho v_3 - 2v_1}{2} \cdot t_d}_{\text{deceleration space}} + \underbrace{(\rho v_3 - v_1) \cdot (t_o - t_d)}_{\text{space at } \rho v_3}. \tag{3}$$

We can then compute the value of ρ needed to satisfy Eq. (3) and, if the necessary slowdown is smaller than an acceptable threshold (e.g., $10\,\%$), we can then command P_3 to apply such a slowdown and P_2 to start the maneuver.

Clearly, this is not the only possible approach, as P_2 could speed up, P_2 could speed up while P_3 slows down, or even

P_1 could slow down. Any combination is in principle possible, but, as stated at the beginning of the paper, our goal is not to find the best overtaking strategy, but rather to study the feasibility of MEC-based maneuvers.

B. Intersection merge maneuver

With respect to the intersection merge maneuver, we consider a T-shaped intersection with the west-to-east road being the main one and the south-to-north road being a secondary leg, as depicted in Fig. 3. For coherence with the previous maneuver, we consider three platoons, namely platoon 1, 2, and 3 (P_1, P_2, and P_3). P_1 is the first in the west-to-east direction (yellow in Fig. 3), P_2 the second in the same direction (the red one in the figure), while P_3 is follows the south-to-north direction. We assume that P_3 needs to turn right at the intersection, and it is normally required to give the right of way. Using a coordinated approach, however, P_3 might not need to come to a complete stop, as it could potentially pass in front of P_1, or enter the intersection between P_1 and P_2. By means of communications, the vehicles on the main road can cooperate, potentially slowing down slightly to enable P_3 to smoothly enter the intersection.

To model the maneuver, for each platoon i, we define the quantities $t_{i,en}$, $t_{i,ex}$, and v_i which indicate the intersection enter and exit times, and the intersection approach speed. We also define d_i, l_i, and r_i as the distance from the intersection, the length, and the road of platoon i, respectively.

We define a quadratic optimization problem that computes a control action in terms of acceleration each platoon should apply to enter the intersection, minimizing the overall access time and acceleration. Here, we simply consider a simple three-platoon configuration as in Fig. 3, but the approach generalizes to more platoons as well. In particular, we want to optimize the intersection access times $\boldsymbol{t} = \{t_{i,en} \mid i = 1, 2, 3\}$. Given $t_{i,en}$ and considering a linear motion model, we can compute the acceleration a_i necessary to meet such target, i.e.,

$$a_i = 2\frac{-v_i t_{i,en} + d_i}{{t_{i,en}}^2}. \tag{4}$$

Starting from $t_{i,en}$ and Eq. (4), we can define the objective function as

$$f(\boldsymbol{t}) = w_t \sum_{i=1}^{|\boldsymbol{t}|} {t_{i,en}}^2 + w_a \sum_{i=1}^{|\boldsymbol{t}|} {a_i}^2, \tag{5}$$

where w_t and w_a represent two weight factors for the time and the acceleration components. We formulate the optimization problem as

$$\min_{\boldsymbol{t}} f(\boldsymbol{t}) \tag{6}$$

subject to the following constraints:

$$t_{i,en} < t_{j,en}, \quad \forall i, j : r_i < r_j \wedge d_i < d_j \tag{7}$$

$$t_{j,en} \geq t_{i,en} + \frac{l_i + d_s}{v_i}, \quad \forall i, j : t_{i,en} < t_{j,en} \tag{8}$$

$$v_i + a_i t_{i,en} \leq v_{max}, \quad \forall i \in 1, \dots, |\boldsymbol{t}| \tag{9}$$

$$a_i \leq a_{max}, \quad \forall i \in 1, \dots, |\boldsymbol{t}|. \tag{10}$$

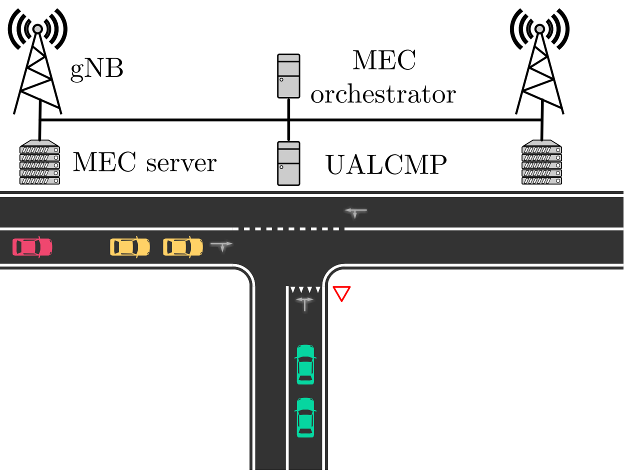


Figure 4. MEC architecture of the considered scenario.

Eq. (7) ensures that no overtaking between platoons on the same road is possible. Eq. (8) maintains the safety distance between each pair of platoons, i.e., if a platoon enters the intersection first, the second one needs to wait for the first to clear the intersection. Finally, Eqs. (9) and (10) ensure that vehicles do not violate speed and acceleration constraints.

IV. Performance evaluation

The simulation framework employed for the performance evaluation is derived from the Plexe ecosystem [18]. The framework has been coupled with Simu5G [16], which offers both a complete 5G stack and a MEC architecture. The federation of Plexe with Simu5G is an additional contribution of the work, which we release[1] to foster additional research on 5G and MEC-based CD. We equip vehicles in the simulation with a 5G communication interface which connects to the MEC infrastructure, as well as direct V2V using 5G sidelink Mode 1 (under gNodeB coverage).

Fig. 4 shows a general view of the simulation architecture, including vehicles and all the MEC components. Each vehicle runs two applications on top of the 5G networking stack, i.e., a device application and a User Equipment (UE) application, following the standard MEC architecture [19], [20]. The device application communicates with the User Application LifeCycle Management Proxy (UALCMP), enabling it to retrieve the list of resources the UE can access and requesting the instantiation of a specific MEC application. The UALCMP, in turn, communicates with the MEC orchestrator which, chooses the best location for instantiating the MEC application based on the current status of the MEC servers, on the resources required by the requested application, and on other policies. Once the MEC application is instantiated, the UALCMP notifies the device application, which in turn communicates the required IP/port pair to the UE application.

A MEC application handling a cooperative maneuver is clearly shared among the set of vehicle UEs participating in the maneuver, which is a feature that is considered in ETSI

1. https://github.com/michele-segata/plexe

MEC. When sending an application context creation to the UALCMP, the device application can request to instantiate or to join an existing application [20, Sect. 5.1.3]. This feature is currently not available in Simu5G, so we implemented it and we will release it together with the whole framework.

As the main goal of the paper is to evaluate the impact of MEC delay, we introduce artificial MEC to UE response delays which simulate the impact of communication and computation at the same time. This artificial response delay accounts for all potential MEC delay sources.

A. Cooperative overtaking maneuver

This section describes the results obtained for the cooperative overtaking maneuver. We start by describing the simulation parameters in Tab. I. The road is $1\,\mathrm{km}$ in length and the simulation terminates when vehicles travel the whole stretch of road. In case of unsuccessful maneuver, the simulation is terminated prematurely and the outcome is stored in the output. Failure can either be caused by a collision between the overtaking and the incoming vehicle or in cases when the algorithm decides to begin the overtake but later realizes that safety conditions are not met anymore. In this work, we do not implement any fallback, as we just want to show the impact of communication and computation delays on safety. In particular, the artificial MEC response delay is either null (to show the performance in perfect conditions) or uniformly distributed over different ranges, starting with $0.1\,\mathrm{s}$ to $0.2\,\mathrm{s}$, up to $1\,\mathrm{s}$ to $2\,\mathrm{s}$. Although computation and communication delays cannot be uniformly distributed in practice, we consider the response delay to be uniformly distributed only as a proof of concept. Additionally, we consider different slow-down factor ρ.

Table I
SIMULATION PARAMETERS FOR THE COOPERATIVE OVERTAKING MANEUVER.

Parameter	Value
Road length	$1\,\mathrm{km}$
a_2, a_3	$1.5\,\mathrm{m/s^2}$, $-1.5\,\mathrm{m/s^2}$
v_1, v_2, v_3	$50\,\mathrm{km/h}$, $70\,\mathrm{km/h}$, and $45\,\mathrm{km/h}$
x_1, x_2, x_3 (see footnote[2])	$150\,\mathrm{m}$, $50\,\mathrm{m}$, $100\,\mathrm{m}$ to $250\,\mathrm{m}$
ρ	0.8, 0.9
MEC response delay (s)	0, U(0.1, 0.2), U(0.2, 0.5), U(0.5, 1), U(1, 2)
d_s	$1.2\,\mathrm{s} \cdot v_1$
ϵ	$20\,\mathrm{m}$
Repetitions	10 per parameter set

The overtaken (P_1) and overtaking vehicles (P_2) always start in the same position, whereas the incoming one (P_3) starts in different positions to generate different situations. Their initial driving speeds are the same in all simulations, reproducing a classic scenario where a faster vehicle wants to overtake a slower one. Each set of parameters is repeated 10 times for statistical confidence. The results of the maneuver are compared against a baseline, where P_2 waits for P_3 to pass-by before overtaking P_1.

Concerning communication parameters, we do not make changes with respect to default values that come pre-set in Simu5G, so for the sake of brevity we omit them. The only important parameter to mention is that vehicles send their data to the MEC application ten times a second.

We start the analysis by looking at the travel times. In Fig. 5 we plot the average travel time as a function of the initial position of P_3. The travel time is computed as the average of both P_3 and P_2 over all simulation repetitions. We plot the average for the different delay distributions and values of ρ, and compare it against the baseline.

Starting with the baseline, this is decreasing simply because P_3 starts at different initial positions. For smaller initial positions, regardless of the value of ρ or the delay, the average travel time is shorter, thanks to the fact that P_2 does not need to wait for P_3 before performing the overtake. The reduction is limited, but significant considering that the road is only one kilometer long. The delay here does not play a role because P_3 is far enough to accommodate errors.

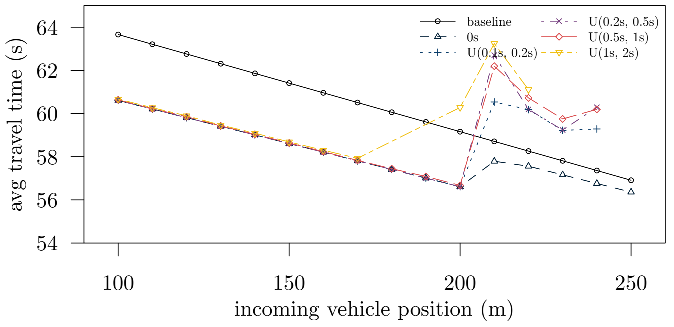
Considering the results with no delays for an initial position above $200\,\mathrm{m}$, we observe that the average travel time starts increasing, simply because the algorithm is slowing down P_3 to enable P_2 to perform the maneuver without having to wait. Still, the average travel time is shorter.

When considering communication and computing delays, however, we observe a negative impact on the maneuver's performance. For $\rho = 0.9$, the average travel time is larger, but still comparable to the baseline. For $\rho = 0.8$, instead, the average travel time becomes larger than the one of the baseline. This can be explained by the average speed of the vehicles. While the average speed of P_2 always matches the target one (Fig. 6a), the one of P_3 becomes much smaller, as the delayed response causes a delayed control action, requiring additional slow-downs for the maneuver to be completed. In addition, for extremely large delays, some simulation points are completely missing, indicating that all simulations resulted in a collision or in an unsafe situation.

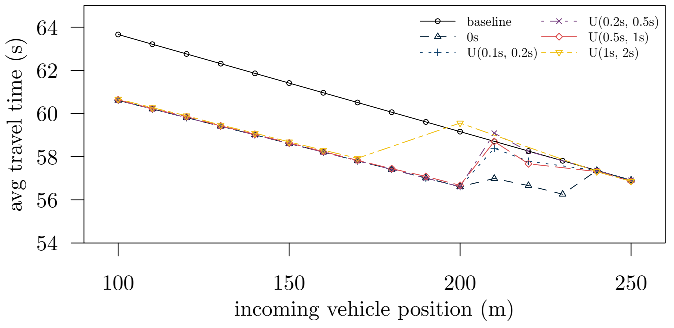
For a better understanding of the impact of delays on safety, Tab. II shows the percentage of simulations that resulted in a collision, for different delay distributions and values of ρ. Besides the intuitive outcome for very large delays, even delays between $0.5\,\mathrm{s}$ and $1\,\mathrm{s}$ can lead to collisions in $6\,\%$ to $8\,\%$ of the cases. Considering that these are head-on collisions, it is simply not acceptable.

Tab. II gives, however, a partial overview, as it does not account for cases in which the simulation has been terminated due to unsafe condition. To conclude the analysis, in Tab. III we show the split percentage of decisions taken by the algorithm, again for different values of ρ and delay. In the table, "No slowdown" indicates that the algorithm managed to implement the maneuver without slowing down P_3, "Slowdown" by slowing it down, and "Wait" by making P_2 wait for P_3 to pass-by. The "Unsafe" label refers to the case in which the algorithm initiates the overtaking maneuver but then, upon re-evaluating the state of the vehicles, decides that safety conditions are no longer met. In this latter case we also terminate the simulation

2. The initial position of P_3 is relative to its traveling direction, so starting from east.



(a) $\rho = 0.8$



(b) $\rho = 0.9$

Figure 5. Average travel times for overtaking and incoming vehicles.

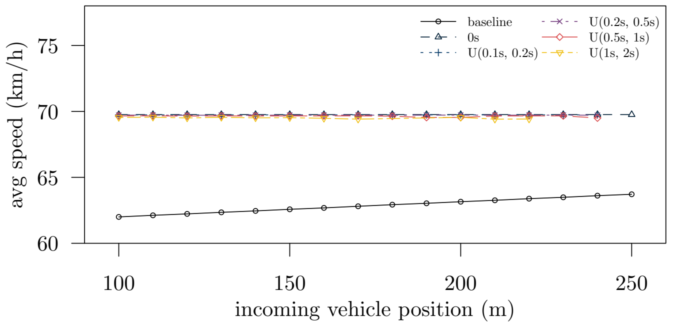
Table II
PERCENTAGE OF COLLISIONS FOR DIFFERENT VALUES OF ρ AND DELAY DISTRIBUTIONS.

	Delay (s)				
ρ	0	U(0.1, 0.2)	U(0.2, 0.5)	U(0.5s, 1s)	U(1s, 2s)
0.8	0	0	0	8.12	35.62
0.9	0	0	0	6.88	30

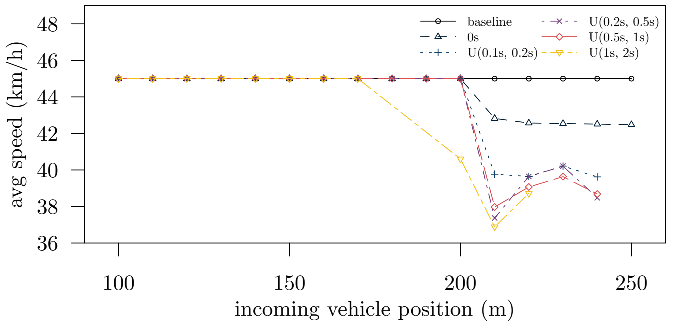
because, without a fallback mechanism, such situations would lead to a collision.

The interesting result from the table is exactly the last column ("Unsafe"). The percentage of unsafe cases even for small delays are substantial, i.e., always larger than 6 %. For $\rho = 0.9$, even with no delay, 3 % of the maneuvers still resulted in an unsafe situation. In depth analysis has shown that this is due to improper tracking of the slow-down (or speed-up) profiles of P_3 and P_2, which can be solved by considering larger safety margins or better tracking algorithms.

Overall, the results for the cooperative overtaking maneuver indicate that the precision necessary for safely performing this type of maneuver is very high and even moderate delays can negatively affect the safety of vehicles. This clearly will have an impact on MEC solutions, as to counteract such delays we might need to be able to predict them well in advance and potentially migrate the application. Even migration can cause additional delays, so such approaches must be carefully designed to meet the safety guarantees of cooperative driving.



(a) overtaking vehicle (P_2)



(b) incoming vehicle (P_3)

Figure 6. Average speeds for overtaking and incoming vehicles, $\rho = 0.8$.

Table III
PERCENTAGE OF SIMULATIONS USING SPECIFIC OVERTAKING DECISIONS FOR DIFFERENT VALUES OF ρ AND DELAY DISTRIBUTIONS.

		Overtake decision			
ρ	Delay (s)	No slowdown	Slowdown	Wait	Unsafe
0.8	0	68.75	30.63	0.00	0.62
0.8	U(0.1, 0.2)	68.75	25.00	0.00	6.25
0.8	U(0.2, 0.5)	68.75	25.00	0.00	6.25
0.8	U(0.5, 1)	70.75	22.45	0.00	6.80
0.8	U(1, 2)	66.99	13.59	0.00	19.42
0.9	0	68.75	15.00	12.50	3.75
0.9	U(0.1, 0.2)	68.75	12.50	12.50	6.25
0.9	U(0.2, 0.5)	68.75	12.50	12.50	6.25
0.9	U(0.5, 1)	69.80	10.07	13.42	6.71
0.9	U(1, 2)	61.61	0.89	17.86	19.64

B. *Cooperative intersection merge*

Considering the intersection merge maneuver, Tab. IV lists the simulation parameters. At the beginning of the simulations, each platoon has the same initial speed (50 km/h). We generate different initial conditions by using different distances from the intersection and different inter-platoon distances between P_2 and P_1. We have chosen a large set of initial conditions to maximize the number of possible outcomes and increase statistical confidence. However, not all such initial conditions lead to feasible solutions, so the results concerning invalid initial conditions have been removed during post-processing.

We test the maneuver for two different v_{max}, of which one enabling the platoons to go faster than their initial speeds (60 km/h). The choice of weights w_t and w_a does not follow

Table IV
SIMULATION PARAMETERS FOR THE COOPERATIVE INTERSECTION MERGE

Parameter	Value
Intersection size	$350\,\mathrm{m}$ (west/south legs) + $300\,\mathrm{m}$ (east leg)
Initial speed	$50\,\mathrm{km/h}$
Distance from intersection (P_1)	$200\,\mathrm{m}$ to $340\,\mathrm{m}$ every $20\,\mathrm{m}$
Distance P_2 – P_1	$20\,\mathrm{m}$, $40\,\mathrm{m}$, and $60\,\mathrm{m}$
Distance from intersection (P_3)	$200\,\mathrm{m}$ to $300\,\mathrm{m}$ every $20\,\mathrm{m}$
Platoon sizes (P_1, P_2, P_3)	3, 4, and 5
v_{max}	$50\,\mathrm{km/h}$ and $60\,\mathrm{km/h}$
a_{max}	$2.5\,\mathrm{m/s^2}$
w_t, w_a	0.0005, 16
MEC response delay (s)	0, U(0.1, 0.2), U(0.2, 0.5), U(0.5, 1), U(1, 2)

a strict procedure, as the goal here is not to find the parameters that maximize flow or minimize consumption. We have set $w_t << w_a$ to avoid that the optimization yields very large accelerations just to minimize travel times. To solve the optimization, we coupled PLEXE with Matlab, enabling the MEC application to solve the optimization problem every time the platoon leaders send their data to it, which is done once a second differently from the overtaking maneuver.

Fig. 7 shows the distribution of the minimum acceleration implemented by each platoon leader in all simulations, plotted using standard boxplots. The idea is to show whether delay has an impact on acceleration, which might result in sub-optimal tracking of the speed profile, limiting the actual efficiency of the maneuver. The graphs are horizontally split into three outcome scenarios, i.e., when the optimization algorithm makes P_3 enter the intersection first, in between P_1 and P_2, and as last. The results are then shown for the two values of v_{max} considered.

First of all, the plots show different deceleration patterns due to the different outcome scenarios. For example, for $v_{max} = 50\,\mathrm{km/h}$, scenario I, we see that basically only P_1 is decelerating to enable P_3 to enter before it, whereas in scenario III P_3 slows down to wait for P_1 and P_2 to cross the intersection.

With respect to the impact of delays, quite surprisingly the graphs show almost no impact, except a slight decreasing trend as function of the delay. This might be due to multiple reasons, e.g., that the scenario considered is rather ideal, with no interference from other vehicles. Moreover, given the length of the road, vehicles have plenty of time to correct their acceleration profiles, as also seen by the very small minimum accelerations. This is different from the cooperative overtaking, where both the overtaking car and the incoming car are approaching each other at high relative speed, so the impact of delay is definitely higher.

To actually appreciate the impact of delay, Fig. 8 shows the acceleration profile of the leader of P_2 in one specific simulation, i.e., when merging between P_1 and P_2 with v_{max} set to $60\,\mathrm{km/h}$. The profile displays the vehicle slowing down to get in the right position for merging and then, after entering the intersection, speeding up to cruising speed. During the deceleration phase, the vehicle experiences different profiles depending on the delay. The profile confirms the considerations made previously. Given the configuration of the scenario, the vehicles have enough time to compensate the error even if large delays occur. However, this comes at the cost of a stronger deceleration and a longer acceleration to get back to cruising speed, which in the end will translate into a larger fuel consumption. The absolute difference between the best and the worst case is just $0.25\,\mathrm{m/s^2}$, but in the best case the deceleration is between -0.4 and $-0.5\,\mathrm{m/s^2}$. For this specific case, the acceleration error is thus close to $50\,\%$.

Fig. 9 shows the average time required to complete the maneuver (computed over all simulations for $v_{max} = 50\,\mathrm{km/h}$), compared to the baseline scenario where P_1, P_2, and P_3 proceed along their path without acceleration. This results in either Scenario I or Scenario III, as the inter-platoon distance in Scenario II is less than the length of P_3. We clearly observe that MEC-enabled maneuvers reduce the average travel time by nearly $10\,\%$ in Scenario I and $26\,\%$ in Scenario III.

The takeaway here is that different applications can have completely different requirements and levels of "robustness". This suggests that, when designing an intelligent migration scheme for MEC-based cooperative driving applications, the application itself might need to be considered as a feature for the intelligent algorithm, and that a "one-size-fits-all" solution might actually lead to sub-optimal results.

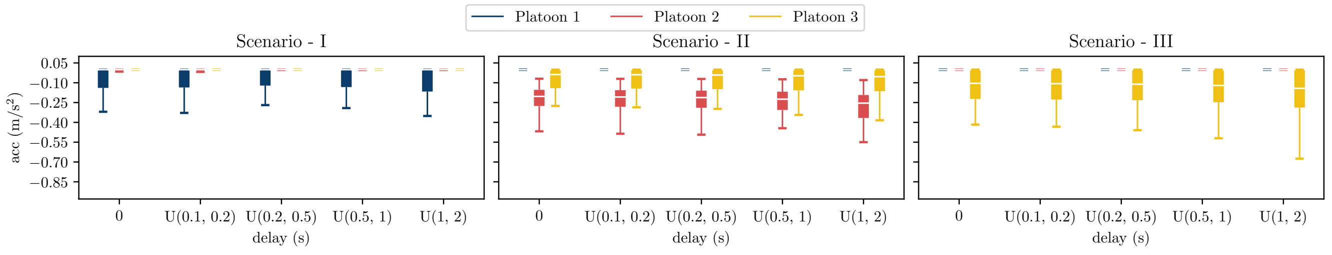
V. CONCLUSIONS AND FUTURE WORK

This work presented the idea of cooperative driving-as-a-service (CDaaS), which bases its foundation on offering cooperative driving maneuvers as services implemented as multi-access edge computing (MEC) applications, together with implementing two maneuvers and showing the impact of communication and computing delay on their efficiency. The maneuvers were implemented in an open-source simulator and tested under different parameters, initial conditions, and delays. The results showed that delay indeed has an impact on safety and efficiency, but it affects different maneuvers to different extents. This suggests that, when designing migration strategies for cooperative driving applications based on MEC, one should both consider the delay and the application itself.

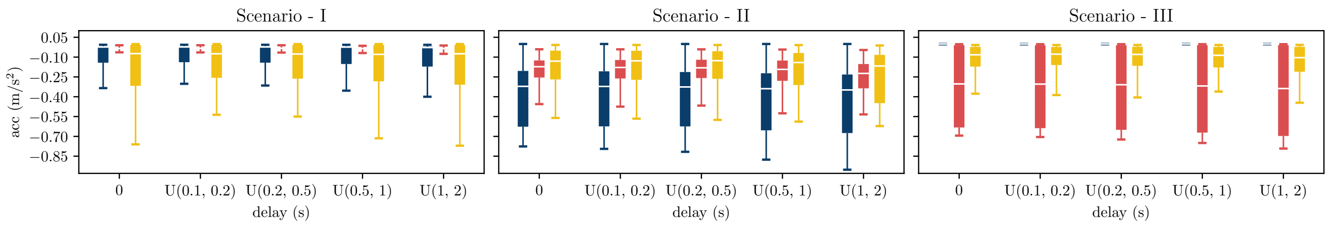
The outcome of this work suggests future directions, which include studying the impact on additional maneuvers, adaptive update frequencies, as well as on additional conditions such as fallback strategies in case of failures and realistic delay distributions. Furthermore, we will design and study different application-dependent migration strategies, to cope with delays and maintain cooperative driving maneuvers safe and efficient.

ACKNOWLEDGMENTS

This work was supported by Italy's PRIN, prot. 2022SKLZAY, funded by the European Union under Next Generation EU, Mission 4 Component 2 - CUP E53D23000920001 (SELF4COOP) and by the European Union – Next Generation EU under the Italian National Recovery and Resilience Plan (NRRP), Mission 4, Component 2, Investment 1.3, CUP B53C22003970001 (C2ASPER), partnership on "Telecommunications of the Future" (PE00000001 – program "RESTART"). Views and opinions expressed are those of the author(s) only and do not necessarily reflect those of the EU or the EU REA. Neither the EU nor the granting authority can be held responsible for them.



(a) v_{max} = 50 km/h



(b) v_{max} = 60 km/h

Figure 7. Statistical distribution of the minimum acceleration of each platoon during the maneuver over all possible simulations, for different values of v_{max}. Scenarios indicate, from left to right, when P_3 enters the intersection before P_1, after P_1 but before P_2, and after P_2.

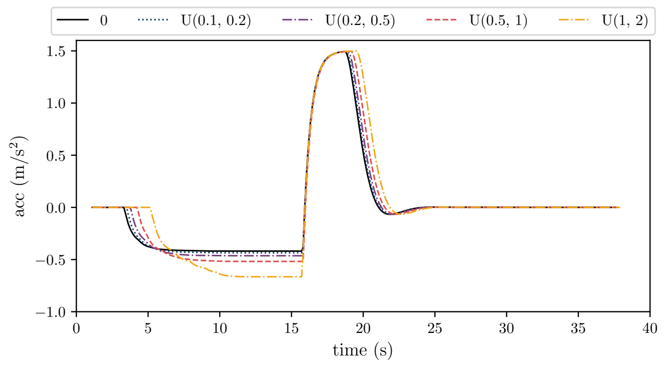


Figure 8. Impact of delay on the acceleration profile of the platoon leader of P_2 as a function of time. The profile is for a specific simulation of Scenario II (v_{max}= 60 km/h) where P_3 merges between P_1 and P_2.

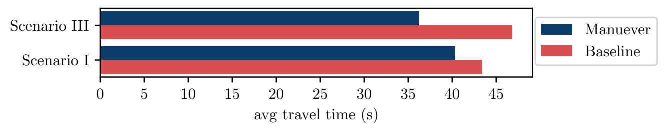


Figure 9. Average travel times of intersection merging maneuver when compared with baseline for different scenarios.

References

[1] R. Lo Cigno et al., "Cooperative driving: A comprehensive perspective, the role of communications, and its potential development," *Elsevier Computer Communications*, vol. 193, Sep. 2022.

[2] B. Hafner et al., "A Survey on Cooperative Architectures and Maneuvers for Connected and Automated Vehicles," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 1, 2022.

[3] M. H. C. Garcia et al., "A Tutorial on 5G NR V2X Communications," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 3, Jul. 2021.

[4] M. Segata et al., "Automatic Emergency Braking - Realistic Analysis of Car Dynamics and Network Performance," *IEEE Trans. on Vehicular Technology*, vol. 62, no. 9, Nov. 2013.

[5] C. Ayimba et al., "Driving Under Influence: Robust Controller Migration for MEC-Enabled Platooning," *Computer Communications*, vol. 194, Oct. 2022.

[6] G. Nardini et al., "Platooning-as-a-Service in a Multi-Operator ETSI MEC Environment," *IEEE Access*, vol. 11, 2023.

[7] J. Fonseca et al., "Exploring Multi-access Edge Computing Federation for V2X scenarios in the Route 25 project," in *Proc. IEEE MELECON*, 2024.

[8] F. M. Tariq et al., "Cooperative Bidirectional Mixed-Traffic Overtaking," in *Proc. IEEE ITSC*, 2022.

[9] M. Strunz et al., "CoOP: V2V-based Cooperative Overtaking for Platoons on Freeways," in *Proc. IEEE ITSC*, 2021.

[10] M. A. Guney et al., "Cooperative Maneuver Planning for Mixed Traffic at Signalized Intersections," in *Proc. IEEE ITSC*, 2021.

[11] R. Soua et al., "Multi-Access Edge Computing for Vehicular Networks: A Position Paper," in *Proc. IEEE GLOBECOM Workshops*, Abu Dhabi, UAE, Dec. 2018.

[12] W. Ouedraogo et al., "Can Edge Computing Fulfill the Requirements of Automated Vehicular Services Using 5G Network?" In *Proc. IEEE VTC*, 2024.

[13] L. Farina et al., "Maneuver Coordination Using V2I to Improve Traffic Efficiency at Intersections," in *Proc. IEEE VNC*, Kobe, Japan, May 2024.

[14] F. Malandrino et al., "Edge-powered Assisted Driving For Connected Cars," *IEEE Trans. on Mobile Computing*, vol. 22, no. 2, Feb. 2023.

[15] C. Quadri et al., "Edge-based platoon control," *Computer Communications*, vol. 181, 2022.

[16] G. Nardini et al., "Simu5G – An OMNeT++ Library for End-to-End Performance Evaluation of 5G Networks," *IEEE Access*, vol. 8, 2020.

[17] M. Vazifeh et al., "Addressing the minimum fleet problem in on-demand urban mobility," *Nature*, vol. 557, May 2018.

[18] M. Segata et al., "Multi-Technology Cooperative Driving: An Analysis Based on PLEXE," *IEEE Trans. on Mobile Computing*, vol. 22, no. 8, 2023.

[19] ETSI, "Multi-access Edge Computing (MEC); Framework and Reference Architecture," European Telecommunications Standards Institute, Sophia Antipolis, France, GS MEC 003 V2.2.1, Dec. 2020.

[20] ETSI, "Multi-access Edge Computing (MEC); UE application interface," European Telecommunications Standards Institute, Sophia Antipolis, France, GS MEC 016 V2.1.1, Apr. 2019.