Cooperative Intersection Crossing Over 5G

Citation for the original published paper (version of record):
Castiglione, L., Falcone, P., Petrillo, A. et al (2021)
Cooperative Intersection Crossing Over 5G
IEEE/ACM Transactions on Networking, 29(1): 303-317
http://dx.doi.org/10.1109/TNET.2020.3032652

Cooperative Intersection Crossing Over 5G

Luca Maria Castiglione, Paolo Falcone, *Member, IEEE*, Alberto Petrillo,
Simon Pietro Romano, *Member, IEEE*, and Stefania Santini, *Senior Member, IEEE*

***Abstract*— Autonomous driving is a safety critical application of sensing and decision-making technologies. Communication technologies extend the awareness capabilities of vehicles, beyond what is achievable with the on-board systems only. Nonetheless, issues typically related to wireless networking must be taken into account when designing safe and reliable autonomous systems. The aim of this work is to present a control algorithm and a communication paradigm over 5G networks for negotiating traffic junctions in urban areas. The proposed control framework has been shown to converge in a finite time and the supporting communication software has been designed with the objective of minimizing communication delays. At the same time, the underlying network guarantees reliability of the communication. The proposed framework has been successfully deployed and tested, in partnership with Ericsson AB, at the AstaZero proving ground in Goteborg, Sweden. In our experiments, three heterogeneous autonomous vehicles successfully drove through a 4-way intersection of 235 square meters in an urban scenario.**

***Index Terms*— Autonomous vehicles, distributed control and coordination, network-based communication, 5G networks, performance measurements.**

I. INTRODUCTION

AUTONOMOUS Driving (AD) is definitely one of the most challenging safety critical applications as it involves, among others, advanced sensing and control technologies. Furthermore, communication with other vehicles and/or the traffic infrastructure is expected to influence the development of AD technologies, as it allows to potentially improve the environment awareness beyond the range of the current sensing systems such as cameras, lidars and radars. When relying upon network-based coordination, issues typically related to wireless communication must be taken into account in order to design control algorithms and driving software that are guaranteed to be both safe and reliable. Structures such as buildings and walls that are commonly part of urban scenarios act as obstacles against the direct communication between two or more mobile nodes and may significantly degrade the communication performance, due to the so called *shadowing effect* [1]. Therefore, it is beneficial to use a communications technology that overcomes the shadowing and implements inter-node communication through an upstream centralized dispatching layer. In this article we propose to realize such a higher-level dispatching framework by leveraging 5G-enabled cloud-based inter-vehicle communication. With the proposed approach, every vehicle receives real time traffic updates from the cloud and is made aware of the presence of other nodes. A 5G-based communications solution overcomes the problems due to local obstacles since the cellular network, with the aid of the base station, can establish reliable communications among vehicles. Such a solution would be costly and cumbersome to deploy, if implemented with Wi-Fi 802.11p. This is due to the need of installing numerous access points to overcome shadowing and radio coverage issues. In addition, the cloud based technology powered by Ericsson[1] enables communication to meet the stringent time constraints requested by real-time distributed communication algorithms. The prerequisite for the applicability of fully distributed control architectures for cooperative driving in urban scenarios is the availability of a reliable network that supports 4G+ cloud based communications. The pre-5G proof-of-concept (PoC) at Astazero[2] uses LTE radio with a 5G EPC (*Evolved Packet Core*)[3] core and is designed to support low latency ultra-reliable communications. Hence, it is an ideal candidate for safety critical autonomous driving applications.

In this article, we address the challenging problem of coordinating connected self-driving cars at urban traffic junctions, where traffic efficiency has to be achieved while guaranteeing safety. In the proposed control paradigm, the cyber-physical traffic system is considered as a multi-agent system (MAS) composed of different dynamical agents, i.e., the vehicles that automatically control their dynamical behavior based on both local information and data shared by their neighbors via the communication network. The fully autonomous coordination of the self-driving cars at road intersection is solved by proposing a distributed nonlinear cooperative protocol based on the MAS abstraction. Note that MAS tools appear to be an alternative viable framework for controlling vehicles in a completely distributed fashion with a computational load compatible with real-life automotive applications. More notably, the effectiveness of the theoretical framework is experimentally tested by enabling communication through the pre-5*G* PoC network deployed by Ericsson at the *AstaZero* proving ground for autonomous vehicles. Experiments were

Manuscript received July 17, 2019; revised April 11, 2020, July 10, 2020, and August 30, 2020; accepted October 11, 2020; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor X. Liu. *(Corresponding author: Simon Pietro Romano.)*

Luca Maria Castiglione was with DIETI, University of Napoli Federico II, Naples, Italy. He is now with the Department of Computing, Imperial College London, London SW7 2AZ, U.K.

Paolo Falcone is with the Department of Electrical Engineering, Chalmers University of Technology, 412 96 Gothenburg, Sweden.

Alberto Petrillo, Simon Pietro Romano, and Stefania Santini are with the Department of Electrical Engineering and Information Technology (DIETI), University of Napoli Federico II, 80125 Naples, Italy (e-mail: spromano@unina.it).

This article has supplementary downloadable material available at https://ieeexplore.ieee.org, provided by the authors.

Digital Object Identifier 10.1109/TNET.2020.3032652

---

[1] http://www.ericsson.com/en

[2] http://www.astazero.com/

[3] Information about the development of the PoC is publicly available online at the following address http://www.astazero.com/wp-content/uploads/2016/06/ITS-for-ASTA-Zero-KEYNOTE-ERICSSON.pdf

carried out on two passenger vehicles, namely a Volvo $XC90$ and a Volvo $S90$, and one truck Volvo $FH16$. The main outcome of this research work shows that 4G+/5G networks will definitely play an important role in automotive applications, by allowing safe, real-time and reliable autonomous driving maneuvers.

The paper is organized as follows. Section II presents state of the art strategies for the coordination of autonomous vehicles over street junctions. Section III and IV introduce the preliminaries and the mathematical problem formulation, respectively. Then, in section V the adopted coordination strategy is presented. The application module developed to enable the communication between vehicles over cellular networks is introduced in section VI, while section VII presents a detailed description of the hardware instrumentation deployed on the autonomous vehicles and involved in the field tests. Experimental results are presented and validated in Section VIII, while the network performance during the experiments is discussed in section IX. Finally, conclusions are summarized in section X.

II. BACKGROUND

In the rich technical literature about connected autonomous vehicles, different techniques for safe intersection crossing have been mainly categorized as either *centralized* or *decentralized* (see [2] and references therein). In centralized approaches, an Intersection Coordination Unit (ICU) acts as a supervisor that coordinates vehicles' tasks in order to optimize some performance index while avoiding collisions [3]. However, when considering an intersection involving a large number of autonomous vehicles, such centralized architectures may result unsuitable because of both their limited capability to gather and process a large data set, and the difficulty arising from solving in real-time the consequent large-scale optimization problem [4]. On the other hand, in decentralized approaches each vehicle determines its dynamic behavior on the basis of only the information received by its neighbors. In particular, once the crossing time or order is scheduled, a control strategy locally provides the required acceleration/deceleration profile for each vehicle, based on the information received from its neighboring vehicles. Optimal control approaches are common to enforce the hard safety constraints necessary to avoid collisions, as for example in [5]–[7]. More notably, [6], [7] also carried out an experimental campaign by using Vehicle-to45‘ -Vehicle (V2V) over Wi-Fi (based on the IEEE 802.11p protocol) and provided the experimental validation of the proposed optimal control approach. In this case, experiments are performed in an extra-urban area where no structure, such as buildings and walls, are present. Therefore, some of the issues related to wireless communications in urban scenarios (see Table I) have not been considered. In fact, elements such as buildings, trees and walls constitute an obstacle to high frequency (Wi-Fi) communications by shadowing the signal. Indeed, 802.11p proves to be more suitable in modern cities than 802.11a. In this specification the cyclic prefix length is doubled by halving the bandwidth, which in turn gives more resilience to large delay spreads. However, 4G+/5G is preferable over Wi-Fi, in this context, as it has been proved to be more reliable, as well as more cost-effective and easier to deploy [8]. Cellular communications are preferred since 5G NR enables reliability through its support to Ultra Reliable Low Latency Communication (URLLC). Also, by leveraging the cellular network for inter-vehicle communication, there is no need to deploy and install further technical equipment on the ground. 4G+/5G is intrinsically less disturbed as cellular networks operate within a controlled and licensed spectrum. In view of the above considerations, this work explores the possible use of a pre-5G PoC, using LTE radio with 5G EPC technology, for the Cooperative Intersection Crossing (CIC). This entails that each vehicle autonomously makes decisions based on the information received from the pre-5G network, minimizing the computational delays at the road infrastructure side, that might significantly increase in dense scenarios. Connections between autonomous vehicles and infrastructure have been organized as shown in Fig. 1, where a real-time, 5G-enabled dispatcher component receives per-vehicle information (like, e.g., position, speed, acceleration, etc.) via unicast and then broadcasts collected data to all of the other vehicles involved in the cooperative intersection crossing.

TABLE I
LTE+ 5G EPC COMMUNICATION VS WI-FI IN URBAN AREAS

	Wi-Fi 802.11a	**Wi-Fi 802.11p**	**Cellular LTE**
Shadowing	Suffers	Suffers	Does not suffer
Available bands	2.4 GHz and 5 GHz	$5.85 - 5.925$ GHz	400MHz to 60 GHz
Authentication	Association per station	Originally absent at MAC Level	Association per handover
Reliability	Absent	Guaranteed by halving the bandwidth of 802.11a	Ultra Reliable Low Latency Communication (URLLC) supported in 5G NR
Cost Effectiveness and ease of deployment	New installations required	New installations required	Already deployed equipment usable

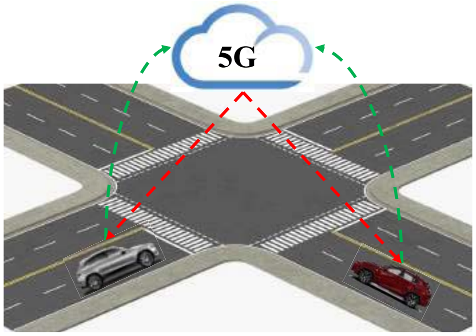


Fig. 1. Inter-vehicle communication over 5G infrastructure.

In the following, we theoretically and experimentally prove that the proposed approach is capable to meet hard-enough real-time constraints. The completely distributed nonlinear finite-time control strategy allows the cooperative negotiation of an intersection, while collisions are prevented by the achievement of the desired virtual formation in a finite time T before the first vehicle accesses the core of the intersection.

It is worth noting that, while a cross intersection is considered throughout this article, the proposed framework can be applied to any type of traffic junction.

III. NOMENCLATURE AND MATHEMATICAL PRELIMINARIES

The communication network established among vehicles can be modeled as a graph, where each vehicle is represented by a *node*, while the existence of a communication link between a pair of vehicles by an *edge*. Specifically, the communication topology of a group of N vehicles can be described by an undirected graph $\mathcal{G}_N = (\mathcal{V}_N, \mathcal{E}_N)$ of order N, with vertex set $\mathcal{V}_N = \{1, \ldots, N\}$ and edge set $\mathcal{E}_N \subset \mathcal{V}_N \times \mathcal{V}_N$, where the presence of the edge $(i, j) \in \mathcal{E}_N$ indicates that the vehicle i receives information from vehicle j, and viceversa. The topology of the graph is associated to the *binary adjacency matrix* $\mathcal{A}_N = [a_{ij}]_{N\times N}$ encoding vehicle communication relationship, where $a_{ij} = 1$ if $(i, j) \in \mathcal{E}_N$, and $a_{ij} = 0$ otherwise. Note that, $a_{ii} = 0$ since self-edges (i, i) are not considered. Therefore, each vehicle i receives the status of all vehicles that are members of its neighbors set $\mathcal{N}_i = \{j \in \mathcal{V}_N : (i, j) \in \mathcal{E}_N, j \neq i\}$. Moreover, a *path* in a graph is an ordered sequence of vertices such that any pair of consecutive vertices in the sequence is an edge of the graph. Here, according to the above definitions, the graph $\mathcal{G}_N$ that describes the communication topology of the cooperative vehicles is assumed to be connected, although not completely.

Note that, according to [9], although in a mixed traffic scenario there might be non-connected vehicles, in this work we assume that all vehicles, both autonomous and human-driven, are connected to the same network and participate in the coordination. Next, we introduce definitions and recall results from literature that will be exploited in the manuscript to establish our main results.

Definition 3.1: (Graph Connectivity) [10]. An undirected graph $\mathcal{G}_N$ is said to be *connected* if there exists a path between any two vertices. In addition, if there exists a path from any vertex to any other vertex, the $\mathcal{G}_N$ is said to be *completely connected*.

The main result in Section IV relies on the Sig Function defined next.

Definition 3.2: (Sig Function) [11]. Let

$$sig(x)^\alpha = sign(x)|x|^\alpha \tag{1}$$

where $\alpha > 0$, $x \in \mathbb{R}$ and $sign(\cdot)$ is the signum function.

Furthermore, for $\alpha > 0$ and $x \in \mathbb{R} \setminus \{0\}$ the following properties [12] hold:

$$\frac{\partial sig(x)^\alpha}{\partial x} = \alpha|x|^{\alpha-1} \; \frac{\partial |x|^\alpha}{\partial x} = \alpha sig(x)^{\alpha-1}, \tag{2}$$

Finally, we recall the finite-time Lyapunov Theorem [11].

Theorem 1: Consider the system $\dot{x} = f(x)$, where $x \in \mathbb{R}^n$, $f : U \to \mathbb{R}^n$ is a continuous function on an open neighborhood $U \subseteq \mathbb{R}^n$ of the origin and $f(0) = 0$. Suppose there exists a continuous positive definite $V(x) : U \to \mathbb{R}$, a real number $c > 0$ and $\alpha \in (0; 1)$ and an open neighborhood $U_0 \subset U$ of the origin such that $\dot{V}(x) + c(V(x))^\alpha \leq 0$, $x \in U_0 \setminus 0$. Then $V(x)$ approaches to 0 in finite time T with

$$T \leq \frac{(V(x(0)))^{1-\alpha}}{c(1-\alpha)}. \tag{3}$$

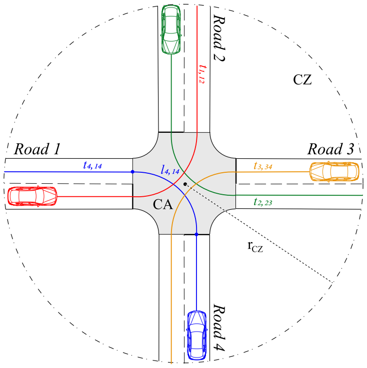
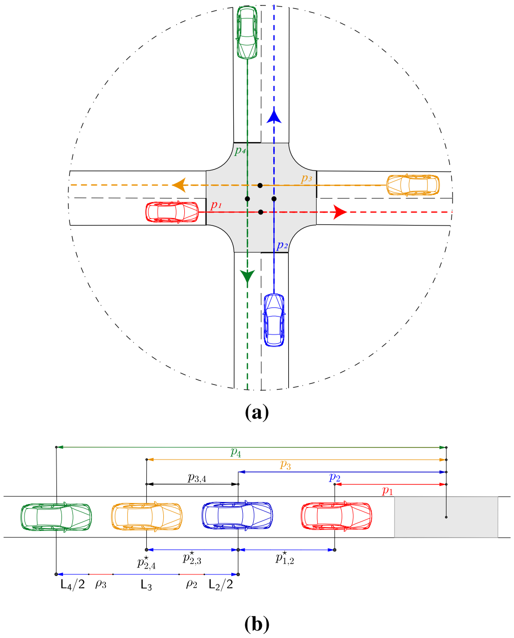


Fig. 2. A possible traffic junction scenario (μ = 4). Self-driving connected cars cooperate for crossing the Conflicting Area (CA). Once inside the Cooperative Zone (CZ), the vehicle i may choose one of the possible trajectories $t_{i,pq}$ starting from the road p where it is initially located.

Theorem 1 provides an upper bound to the time necessary for an autonomous dynamical system (e.g., a closed-loop system) to come to rest, starting from a neighborhood of the origin.

IV. FORMULATION OF THE COOPERATIVE INTERSECTION CROSSING PROBLEM

Consider N vehicles approaching a generic traffic junction from μ different two-lane roads, with no traffic lights or any other kind of signalling provided by an infrastructure acting as central arbiter. All vehicles have to overpass the intersection while avoiding collisions and minimizing the crossing time (virtually, even with no need for a stop). In Cooperative Intersection Crossing (CIC) problem, vehicles are also assumed to be connected via Vehicle-to-Vehicle (V2V) communication in order to share information about their own trajectory and their local state (e.g., see [13] and references therein). Hence, the practical implementation of a CIC strategy is heavily based on a reliable V2V communication network in the urban areas for guaranteeing the smooth and safe crossing of the vehicles through the intersection. Nowadays this is enabled by leveraging on-board modems that allow the vehicles to share their data across an urban cellular network (see Table I where the main cellular features are compared to ones of the vehicular Wi-Fi network based on the IEEE 802.11p protocol). Given a generic intersection, we define its central polygonal zone as the *Conflicting Area (CA)*, i.e., the part of the intersection where collisions could occur, while the larger circular zone around the CA, with radius r_{cz}, is referred to as the *Cooperation Zone (CZ)*, i.e., the zone where vehicles interact (see Fig. 2). The objective of the CIC is that each vehicle in the CZ autonomously regulates its motion, cooperating with its neighbors, to occupy the CA in a mutually exclusive fashion, without side and rear-end collisions [14]. Namely, at any time instant at most one vehicle is allowed to drive without stopping within the CA. Note that the traffic flow at the intersection may be continuous. However, for a specific time interval, we only need to consider a restricted group of N vehicles that are approaching the junction [15]. Under this assumption, as shown in Fig. 2, vehicles inside the CZ will be considered as the group that currently takes part in cooperative crossing, whereas vehicles outside the CZ will be postponed to the next negotiation slot.



(a)

(b)

Fig. 3. CIC. a): autonomous vehicles approaching the traffic junction b): recast into a virtual platoon problem (on the base of the position from the centre $p_i(t)$.

From a control perspective, it is assumed that the path-following is ensured by a lower-level path follower, while the vehicles velocity and the safe spacing among vehicles is automatically achieved via a cooperative control based on the *virtual platoon* concept [16], [17]. In other words, the two-dimensional intersection problem is simplified into a one-dimensional *virtual* platoon control problem (as shown in Fig. 3). The simplification of the coordination problem into a one-dimensional problem is motivated by the fact that intersections are strongly structured environments. As a matter of fact, the vehicles' motion at intersections is very much constrained within their own lanes, due to the very likely presence of other vehicles on adjacent lanes. Obstacles along the lanes are avoided by slowing down/accelerating, while large deviations from the path at the intersection are generally avoided in favor of full stops, if necessary. The crossing sequence is assigned to the connected vehicles based on their actual distance from the intersection center, which is mapped into a crossing order, i.e. the closest vehicle goes first. Since side and rear-end collision must be avoided, a desired spacing policy has to be imposed within the virtual formation, i.e., vehicles have to reach and maintain pre-fixed inter-vehicular gaps as they move with a common velocity. Specifically, the desired distances among virtual platoon members, say $p^\star_{ij}$ $(\forall(i,j) \in \mathcal{E}_N)$, have to be selected so to ensure that the vehicles access exclusively the CA, while the achievement of a common velocity guarantees that the desired formation will be preserved once reached. To this aim, the distributed cooperative algorithms guarantee the achievement of the desired virtual formation in a prescribed finite time T before the first vehicle enters. Hence, the exclusive access to the CA is ensured and collisions are avoided. In our framework, the control design is based on simple linear mathematical models of the agents, where the input longitudinal acceleration is integrated to obtain the vehicle velocity and position (double integrator). This modeling approach is fairly standard in the literature on interconnected vehicles (see, e. g., [18] for a recent survey).

Now the CIC problem can be stated as follows. Let each vehicle within the CZ be described by the following linear model:

$$\begin{aligned} \dot{p}_i(t) &= v_i(t) \\ \dot{v}_i(t) &= u_i(t), \end{aligned} \tag{4}$$

being $p_i(t)$ the position of each vehicle i, expressed as its distance from the center of its trajectory $t_{i,qg}$ (linking the road q, where the i-th vehicle is initially located, with the road g, where the i-th vehicle is heading to, as shown in Fig. 3), and $v_i(t)$ its velocity.

The cooperative control problem can be formulated as follows:

Problem 1: (*CIC – Cooperative Intersection Crossing – in finite time*). Given the virtual platoon, obtained by organizing the N vehicles within the CZ in ascending order of distances from the center of their trajectories $p_i(t)$ $(\forall i \in \mathcal{V}_N)$, find a distributed cooperative control protocol $u_i(t)$ such that $\forall(i,j) \in \mathcal{E}_N$ the achievement of the following desired formation is guaranteed in a finite-time T:

$$\begin{aligned} |p_i - p_j| &\rightarrow p^\star_{ij} \\ |v_i - v_j| &\rightarrow 0 \end{aligned} \tag{5}$$

being $p^\star_{ij} = r_{ij} + hv_i$ the safe virtual inter-vehicular gaps where r_{ij} is the stand-still distance between the vehicle i and the vehicle j, h is the headway time (i.e. the time each vehicle takes to arrive at the position of its predecessor), and v_i is the velocity of the i-th vehicle.

V. DESIGN OF THE FINITE-TIME DISTRIBUTED COOPERATIVE CONTROL FOR CIC

In order to solve Problem 1, here we design a distributed control strategy relying on the communication with the neighbouring vehicles. The choice of a distributed approach sharply reduces the computational load w.r.t. the centralized one and hence is more efficient from a computational point of view. Furthermore, a distributed algorithm generally well scales with the number of vehicles approaching the intersection. The distributed nonlinear finite-time homogeneous control law for each vehicle i is given as:

$$\begin{aligned} u_i(t) = &-\sum_{j=1}^{N} a_{ij} sig(p_i(t) - p_j(t) - p^\star_{ij})^{\frac{2\alpha}{1+\alpha}} \\ &-\sum_{j=1}^{N} a_{ij} sig(v_i(t) - v_j(t))^{\alpha}, \end{aligned} \tag{6}$$

where $\alpha \in (0;1)$ and $sig(\cdot)$ is defined as in definition III.2. Moreover, a_{ij} models the topology of the underlying connected communication graph $\mathcal{G}_N$, i.e., the presence/absence of a communication link between the i-th and j-th vehicle ($a_{ij} = 0$ $\forall j \notin \mathcal{N}_i$ as reported in section III). Note that the controller is distributed in the sense that each agent requires only relative position and velocity measurements of its neighboring agents. Note that, differently from discontinuous signed finite-time nonlinear protocols [19], [20], here we design a homogeneous continuous protocol aimed at avoiding undesirable chattering in the closed-loop trajectories arising from the existence of discontinuous control actions.

A. Finite-Time Stability Analysis of the Closed-Loop network

Given (4) and (6), the closed-loop dynamics for the i-th vehicle can be derived ($\forall i \in \mathcal{V}_N$) as:

$$\dot{p}_i(t) = v_i(t) \tag{7a}$$

$$\dot{v}_i(t) = -\sum_{j=1}^{N} a_{ij} sig(e_{ij}(t))^{\frac{2\alpha}{1+\alpha}} - \sum_{j=1}^{N} a_{ij} sig(v_i(t) - v_j(t))^{\alpha}, \tag{7b}$$

where $e_{ij}(t) = p_i(t) - p_j(t) - p^{\star}_{ij}$ is the distance error between vehicle i and vehicle j according to the desired spacing $p^{\star}_{ij}$.

We next establish a finite-time stability result.

Theorem 2: Consider N self-driving vehicles, sharing information via V2V communication, with closed-loop longitudinal dynamics as in (7). If the corresponding communication graph $\mathcal{G}_N$ is connected in the CZ, then the control strategy $u_i(t)$ in (6) solves Problem 1, i.e., it ensures that vehicles converge to the desired distance with common velocity in a finite time T.

Proof: In order to solve our specific crossing problem, we propose the following Lyapunov function candidate

$$V(e_{ij}(t), v_i(t)) = \sum_{i=1}^{N} V_i \tag{8}$$

where

$$V_i = \sum_{j=1}^{N} \int_0^{e_{ij}(t)} a_{ij} sig(s)^{\frac{2\alpha}{1+\alpha}} \mathrm{ds} + \frac{1}{2} v_i^2(t),$$

which is positive definite, w.r.t. $e_{ij}(t)$ and $v_i(t)$ $\forall i$, $j = 1, \cdots, N$, $i \neq j$. Note that this follows from properties (2). Differentiating the Lyapunov function along the trajectories $p_i(t)$ and $v_i(t)$, solutions of system (7), it follows

$$\begin{aligned}\dot{V}(e_{ij}(t), v_i(t)) = &\sum_{i=1}^{N}\sum_{j=1}^{N} a_{ij} sig(e_{ij}(t))^{\frac{2\alpha}{1+\alpha}} \dot{p}_i(t) \\ &+ \sum_{i=1}^{N} v_i(t)\dot{v}_i(t),\end{aligned} \tag{9}$$

and from (7)

$$\begin{aligned}&\dot{V}(e_{ij}(t), v_i(t)) \\ &= \sum_{i=1}^{N}\sum_{j=1}^{N} a_{ij} sig(e_{ij}(t))^{\frac{2\alpha}{1+\alpha}} v_i(t) \\ &\quad + \sum_{i=1}^{N} v_i(t) u_i(t) = \sum_{i=1}^{N}\sum_{j=1}^{N} a_{ij} sig(e_{ij}(t))^{\frac{2\alpha}{1+\alpha}} v_i(t) - \\ &\times \sum_{i=1}^{N} v_i(t) \Big(\sum_{j=1}^{N} a_{ij} sig(e_{ij}(t))^{\frac{2\alpha}{1+\alpha}} + \sum_{j=1}^{N} a_{ij} sig(v_i(t) - v_j(t))^{\alpha} \Big) \\ &= -\sum_{i=1}^{N}\sum_{j=1}^{N} v_i(t) a_{ij} sig(v_i(t) - v_j(t))^{\alpha}.\end{aligned} \tag{10}$$

Since $sig(\cdot)$ is an odd function, while the adjacency matrix A (defined in section III) is symmetric under the assumption of connected undirect graph $\mathcal{G}_N$, it follows that (10) can be rewritten as

$$\begin{aligned}&\dot{V}(e_{ij}(t), v_i(t)) \\ &\quad = \sum_{i=1}^{N}\sum_{j=1}^{N} v_i(t) a_{ij} sig(v_j(t) - v_i(t))^{\alpha} \\ &\quad = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} v_i(t) a_{ij} sig(v_j(t) - v_i(t))^{\alpha} \\ &\quad\quad + \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} v_j(t) a_{ij} sig(v_i(t) - v_j(t))^{\alpha} \\ &\quad = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} (v_i(t) - v_j(t)) a_{ij} sig(v_j(t) - v_i(t))^{\alpha} \\ &\quad = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} (v_i(t) - v_j(t)) a_{ij} sig(v_i(t) - v_j(t))^{\alpha} \\ &\quad = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} a_{ij} |v_i(t) - v_j(t)|^{1+\alpha}.\end{aligned} \tag{11}$$

Let us now introduce, for the sake of brevity, a more compact notation for the distance errors by indicating each couple of indices $(i, j) \in \mathcal{E}_N$ with a new index ρ. In so doing, errors are referred as elements of the following set $e_\rho(t) \in \{e_{ij}(t) : i, j = 1, \ldots, N; i \neq j\}$ for $\rho = 1, \ldots, m$, being $m = |\mathcal{E}_N|$, i.e., being m equal to the cardinality of the edge set (according to the nomenclature in section III).

Now it is possible to define the following distance error vector as $e(t) = [e_1(t), e_2(t), \cdots, e_m(t)]^{\top}$, while the velocity vector is $v(t) = [v_1(t), v_2(t), \cdots, v_N(t)]^{\top}$. Leveraging the above notation, from (11) one has that $\dot{V}(e(t), v(t)) \leq 0$ and, hence, that $V(e(t), v(t)) \leq V(e(0), v(0)) = V_0$, which indicates that $e(t)$ and $v(t)$ are bounded $\forall t \geq 0$. In addition, since the Lyapunov function $V(e(t), v(t))$ is radially unbounded [21] (see its structure in (8)) it follows that the invariant set Ω, defined as

$$\Omega = \{e(t) \in \mathbb{R}^m,\ v(t) \in \mathbb{R}^N\ : V(e(t), v(t)) \leq V_0\}, \tag{12}$$

is compact. Thus, from the LaSalle Invariance Principle [21] one has that all trajectories that start from Ω converge to the largest invariant set defined as

$$S = \{e(t) \in \mathbb{R}^m,\ v(t) \in \mathbb{R}^N\ : \dot{V}(e(t), v(t)) = 0\}. \tag{13}$$

Note that, since the underlying undirected communication graph is connected, $\dot{V}(e(t), v(t)) = 0$ implies that all vehicles velocities approach the average velocity ($i, j = 1, \ldots, N$, $\forall j \neq i$)

$$v_i(t) = v_j(t) = v^{\star} = \sum_{i \in \mathcal{V}_N} \frac{v_i}{N},$$

which in turn implies that at steady state $u_i(t) = u_j(t) = 0$.

From (6),

$$u_i(t) = -\sum_{i=1}^{N}\sum_{j=1}^{N} a_{ij} sig(p_i(t) - p_j(t) - p^{\star}_{ij})^{\frac{2\alpha}{1+\alpha}} = 0 \tag{14}$$

implies that $sig(p_i(t) - p_j(t) - p^{\star}_{ij})^{\frac{2\alpha}{1+\alpha}} = 0$, or equivalently that $p_i(t) - p_j(t) = p^{\star}_{ij}$. In so doing, it is proven that all vehicles asymptotically converge to the fixed desired formation configuration.

In the following, we will prove that the convergence of the velocity alignment, as well as the convergence of formation stabilization, is achieved in finite time. To this aim, we leverage the homogeneity property of the Lyapunov function according to [11], [12].

Given (8) and (11), for any $\mu > 0$ there holds

$$V(\mu^{\frac{\alpha+1}{\alpha}} e, \mu v) = \mu^2 V(e,v), \tag{15}$$

$$\dot{V}(\mu^{\frac{\alpha+1}{\alpha}} e, \mu v) = \mu^{1+\alpha} \dot{V}(e,v), \tag{16}$$

which verifies the homogeneity properties of $V(e,v)$ and $\dot{V}(e,v)$. Note that for the sake of simplicity, the time dependence has been omitted.

From (16), with $\mu = [V(e,v)]^{-\frac{1}{2}}$ we have

$$\begin{aligned}\frac{\dot{V}(e,v)}{V(e,v)^{\frac{1+\alpha}{2}}} &= \dot{V}(V(e,v)^{-\frac{\alpha+1}{2\alpha}} e, V(e,v)^{-\frac{1}{2}} v)\\ &\leq \max_{(e,v)\in\Upsilon} \dot{V}(e,v)\end{aligned} \tag{17}$$

where

$$\begin{aligned}\Upsilon = \Big\{& e \in \mathbb{R}^m,\ v \in \mathbb{R}^N \setminus \{(0^T, 0^T)^T\} :\\ & V(e,v) = V\Big(V(e,v)^{-\frac{\alpha+1}{2\alpha}} e, V(e,v)^{-\frac{1}{2}} v\Big)\Big\}.\end{aligned} \tag{18}$$

From homogeneity property in (15), it follows

$$V\Big(V(e,v)^{-\frac{\alpha+1}{2\alpha}} e, V(e,v)^{-\frac{1}{2}} v\Big) = \Big(V(e,v)^{-\frac{1}{2}}\Big)^2 V(e,v) = 1. \tag{19}$$

Therefore, $\Upsilon = \{e \in \mathbb{R}^m,\ v \in \mathbb{R}^N : V(e,v) = 1\}$ is a compact set due to the radially unbounded property of $V(e,v)$. Since $\dot{V}(e,v)$ is continuous and non-positive on the compact set Υ, we have

$$\frac{\dot{V}(e,v)}{V(e,v)^{\frac{1+\alpha}{2}}} \leq \max_{(e,v)\in\Upsilon} \dot{V}(e,v) = -c \tag{20}$$

where $c \geq 0$. Furthermore, by the fact that

$$\{e(t) \in \mathbb{R}^m,\ v(t) \in \mathbb{R}^N\ : \dot{V}(e(t), v(t)) = 0\} = \{(0^\top, 0^\top)^\top\}, \tag{21}$$

one obtains $c > 0$. Therefore, condition (20) implies that

$$\dot{V}(e,v) \leq -cV(e,v)^{\frac{1+\alpha}{2}}. \tag{22}$$

Since $\frac{1+\alpha}{2} \in (0;1)$, from Theorem 1 it follows that the closed-loop system is finite time stable with settling time T such that

$$T \leq \frac{2}{c(1-\alpha)} V(e(0), v(0))^{\frac{1-\alpha}{2}}. \tag{23}$$

This completes the proof. ■

Theorem 2 proves that the control law (6) leads to the convergence of the vehicle formation to the desired inter-vehicle distances and a common velocity, under the assumption that, in the CZ, the communication graph $\mathcal{G}_N$ is connected in the sense of Definition 3.1. While this is clearly a strong assumption in a mixed environment, where non-connected road users might be present, in strongly controlled environments (highly automated intersections, factories, construction sites), all the moving agents are connected to the same network and participate in the coordination [9]. However, although this is not the aim of our work, our approach can accommodate the presence of non-connected vehicles, as they can be detected by on-board sensors (e.g., radar and/or lidar) and accounted for in the coordination algorithms. Indeed, a link can be added to the graph topology for each on-board sensor and, accordingly, the controller weights information obtained via both the communication and sensing links [22]–[24].

Remark 1: The settling time estimation can be obtained by computing the Lyapunov function value at the initial point (see e.g., [25]). Note that, According to (23), it is possible to tune the control gain α to select a proper upper-bound for the convergence time. This gain tuning procedure allows considering the convergence rate of the virtual platoon to the desired spacing and speed from the very beginning of the control design phase. Indeed, the analytical derivation provides a control gain-dependent estimation of the settling time T that has to be imposed. This is crucial for our safety critical application where there is the need of selecting and guaranteeing a prescribed settling time before the first vehicle accesses the CA in order to avoid collisions.

VI. COMMUNICATION SOFTWARE: THE HERMES MODULE

In this section we discuss the design and implementation of a generalised, real-time, low-latency and reliable message exchanging system that we called *Hermes*. Hermes acts as an application-level communication infrastructure to support the control algorithm discussed in Section V. Thanks to the level of abstraction that has been used in the architecture design, *Hermes* can provide a generic road user with traffic information, withstanding the differences intrinsic in the specific configuration of each vehicle.

Moreover, even though it is reasonable to assume that each involved vehicle is able to take autonomous decisions based on the information received from the traffic controller, the latter should also provide recipients with additional information (also called *control-side information* in this context) that can bias the final control decision (i.e., the one initially taken locally on the vehicle). An example scenario where such control-side information comes in handy, could be the need to prioritize a vehicle. In fact, the traffic controller might be willing to force the order of vehicles that are about to cross an intersection because a high priority vehicle (e.g., an ambulance) is approaching. To this extent, the communication software is highly decoupled from the adopted control strategy. In order to be reliable, the system must be aware of the status of the connection it has established with any user. In this sense, it is possible to identify two main classes of listeners: 1) *vehicles*, either autonomous or human-driven, for which reliability must be guaranteed; 2) *monitors*, with no specific reliability requirements. Indeed, while vehicles are supposed to proactively leverage the data they get from the traffic controller, monitors are just passively listening to control data in order to, e.g., assess the performance of the overall system. They hence demand for less stringent requirements in terms of reliability. The entire communication system has been designed to be easily deployable and highly scalable, so to seamlessly cope with an increasing number of vehicles. As to the *traffic manager*, it has been developed

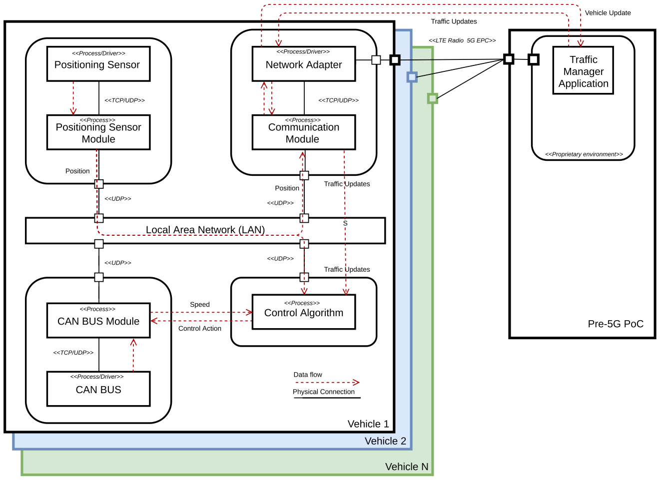


Fig. 4. Hermes software architecture.

using state of the art software and well-known programming methodologies.

A. System Domain Model

Hermes is able to orchestrate traffic in the presence of several road users. In order to do that, the system monitors the status of mobile nodes within a desired area, constantly receiving status messages and providing clients with network updates (see Fig. 9 for a complete example of message exchange). The core element of the system is the *Mobile Node*, a virtual representation of a subscribed road user. Each *Mobile Node* element, which is uniquely identified by a combination of *id* and *name*, stores status information such as *position, speed* and *acceleration* of the corresponding physical node in the *Status* fields. Eventually, the *Timestamp* field is updated with the time value of the last received message. This proves useful when detecting possible communication and vehicular anomalies. Hence, the traffic manager stores the virtual representation of each subscribed node in a subscription list, along with control information such as a *Sequence Number* and a *Global Timestamp*. Finally, the software architecture of the system is sketched in Fig. 4. Positioning and Speed data are sensed by dedicated system modules and sent to the local controller for the computation of the control output. This latter also uses information coming from other vehicles (and dispatched through the Traffic Manager Application) for its computations. Position and Speed are also sent to the Traffic Manager, through the Network Module, as Traffic Updates.

B. System Dynamics

When a node is activated, it is requested to send a subscription request to the traffic manager in order to have its status tracked over the network. Once the subscription has been submitted and approved by the manager, the node in question can start sharing status updates by sending information such as current position, speed and acceleration, along with the above mentioned control data (such as a timestamp and a local sequence number). The traffic manager, in turn, periodically sends the latest updates via multicast to connected nodes, making each of them aware of the status of all vehicles passing through the covered area. Thus, from a high-level point of view, the traffic manager acts as a mean to share 'all to all' information among road users. It is important to highlight a difference between *local* and *global* sequence number. The former is computed locally at each vehicle and belongs to the vehicle's status data structure. It is used to distinguish between two packets originating from the same mobile node. The latter originates at the server and is associated with a *traffic update* packet sent by the traffic manager. Two packets with different global sequence number have both origin in the traffic manager server and are born in two different moments. In particular, the packet with the highest sequence number is the newest one. In order to exchange their own status, road users leverage a dedicated class of messages, referred to as a *to5GPoC* message in this work. Since the designed architecture envisages the presence of different types of vehicles, each such vehicle firstly has to declare its own nature (type of vehicle and type of aid in guide) by properly filling in the *Vehicle Type* field of a *to5GPoC* message. Also, this architecture uses the field *Vehicle Name* to distinguish between two different vehicles. In a real scenario, this field has been thought to be filled in with unique information such as plate number or VIN (Vehicle Identification Number). The following data are also included in this kind of message: a) *GNSS Coordinates*: current Position of the vehicle according to the WGS84 standard [26]; b) *GNSS Heading*: heading angle of the running vehicle; c) *Vehicle Speed*: current Speed of the vehicle. This field is composed of two values, namely the *latitudinal speed* and the *longitudinal speed*; d) *Proximity*: here the vehicle declares its distance in meters from the intersection it is approaching and that is supposed to be autonomously negotiated. If the traffic manager has to handle more than one intersection, it can mix this information with data coming from the GNSS to figure out which specific intersection the vehicle is approaching; e) *Connection Status*: this field is used to keep track of the connection status of the vehicle. The assumed value should be binary: *Active* or *Inactive*; f) *Latency*: the latency measured across the connection between the node and the traffic manager, based on the last message received; g) *Local Timestamp*: time at the mobile node when information encapsulation has taken place at application level; h) *Local Sequence*: sequence number of the packet. This counter is managed by the local node and will be reset to 0 when the connection is restarted.

On the other side of the communication, the traffic manager is listening for updates from subscribed nodes, keeping its vehicle list up-to-date with received information. It then periodically shares the stored traffic data with road users. Messages sent from the server are classified as *Traffic Update* messages and contain the following data: a) *Connected Nodes*: the number of mobile nodes currently subscribed to the network. b) *Global Sequence Number*: the sequence number of the current message. It is computed at the traffic manager and keeps track of the messages sent. This counter only resets when the server is shut down. c) *Global Timestamp*: time value extracted on the server machine when a *Traffic Update* packet is encapsulated at application level. d) *Control Side Information*: additional control data that may prove useful when there is a need to bias local decisions. This field can be used, for instance, if the manager wants one vehicle (i.e., ambulance) to be prioritized against the others, as well as in the case that stakeholders decide to switch to a more centralized control strategy.

C. Hermes Architecture

Hermes has been built by combining the simplicity of the *Client-Server* pattern with the efficiency of the *Multicast* communication paradigm. As already mentioned, the traffic manager plays a role which is of paramount importance in the overall architecture. It acts as a server, with the mobile nodes representing the clients. The *Traffic Manager* server receives messages from clients, elaborates them and eventually shares the results among vehicles through a multicast session. In order to add an additional level of reliability to the communication, messages between clients and traffic manager are exchanged over TCP rather than UDP. With this choice we actually traded slightly decreased network responsiveness for improved communication reliability and this is justified by the critical nature of the application. In fact, even though in 5G networks the reliability of the communication is guaranteed, at the physical layer, by URLLC [27], the further layer of reliability added by TCP does make it possible to safely use the *Hermes Traffic Manager* software even in areas where 5G coverage is not ensured and the connections are downgraded to standard LTE. It can also be noticed that, whenever 5G coverage is already available, the overhead introduced by TCP is minimal [28] (since a fault will cause a re-transmission at the physical layer and stay transparent to TCP) and it can be considered a fairly low price to be paid, which allows to gain portability toward classical LTE networks. A fully-fledged version of the *Hermes Traffic Manager* (that we called *HermesJS*) has been implemented to carry out the experiments at AstaZero proving ground. The implementation of the service uses WebSockets [29] as a means to exchange messages among involved entities. If the communication happens over a public network, the connection can be easily upgraded to secure WebSockets. In a nutshell, WebSockets represent an advanced technology that makes it possible to open an interactive, event-driven communication session between client and server with no need for polling to receive a reply. On the server side, the *socket.io* implementation of WebSockets has been used, within the context of a *Node.JS* environment. The HermesJS server waits for incoming HTTP connections and upgrades them to the WebSocket protocol if they are supposed to interact with the traffic manager. This choice comes as a result of a trade-off between availability, reliability and easy prototyping. While the reliability of *Node.JS* is not proved, there are several studies (such as [30]) on its availability and security attributes. However, it has proved to be reliable during our trials.

The deployed proof of concept slightly diverges from the discussed design, particularly in relation to the way multicast is implemented. Indeed, according to the standard patterns, two different connections should be used by each client to send the status and receive road-traffic information. In this sense, outgoing information should travel towards the traffic manager across a dedicated *client-server* route, while incoming data are supposed to be dispatched via multicast at network level. What actually happens in the discussed implementation is that the multicast paradigm is implemented at the application level. Communication between a node and the traffic manager actually happens, for a single TCP flow, via bidirectional unicast. Consequently, from a network perspective each vehicle opens a TCP connection towards the traffic manager and uses this stream both to send and receive messages.

In this scenario, each client negotiates a session with the *Traffic Manager*, by setting up a websocket connection towards it. Once the connection has been established, the node engages in a subscription operation. During this phase, it sends a subscription message along with an identifier. If such an identifier has not been taken yet, the traffic manager notifies the occurred subscription by sending a positive acknowledgement. Otherwise, the client will be disconnected. After a successfully completed subscription, mobile nodes are able to communicate updates about their status. In parallel, the *Traffic Manager* broadcasts received information, along with optionally computed control side data, at a frequency of 20 Hz. The 20 Hz update frequency has been chosen to strike a balance between the need for minimizing network traffic overhead on one side and that of maximizing the effectiveness of the communication on the other. The entire message exchange sequence is sketched in Figure 9.

D. Mobile Nodes

In order to enable test cars to communicate over 5G, an additional module has been designed and developed, in accordance with the mobile node specifications. This software is highly asynchronous and is written in low level C++ code in order to allow for maximum performance. It is logically divided in two main components running in parallel in different threads: a) *Remote Sender*: a time-triggered asynchronous thread that periodically[4] sends data about current state of the vehicle towards the 5G PoC; b) *Remote Receiver*: an asynchronous thread triggered by an incoming message. The main purpose of this method is the extraction of traffic data from the websocket data format and the initialization of an internal traffic data structure coherent with the car software and components.

VII. TEST CARS SOFTWARE ARCHITECTURE AND EXPERIMENTAL SETUP

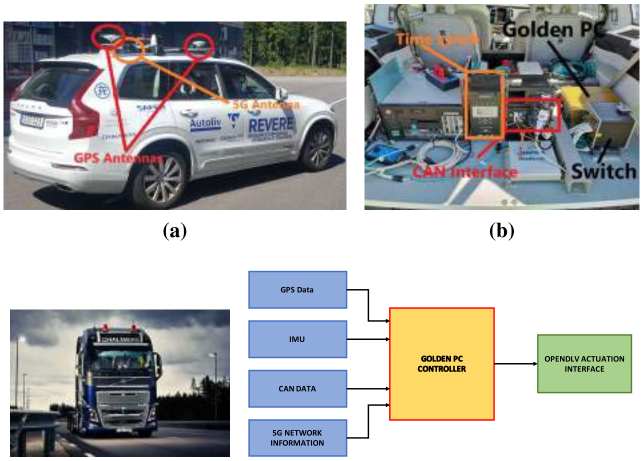
The experimental setup consists in three vehicles (namely two cars, Volvo XC90 and Volvo S90, and one Truck, Volvo FH16) that exchange information via the pre-5G communication test network provided by Ericsson. Vehicles are heterogeneous in their masses, power-trains and on-board systems. Namely, the Volvo Car XC90 and the Volvo Truck FH16 are equipped with the open-source driving system OpenDLV [31], [32], while the Volvo Car S90, provided by the proving ground AstaZero, is equipped with an ADB Pedal Robot [33] that controls the longitudinal vehicle motion by acting on its throttle/brake pedals. The robot can be controlled through a proprietary interface that, in this context, has been accessed with the *Matlab Realtime* tool. In the following we detail the main on-board hardware devices and software components.

A. Volvo Car XC90 and Volvo Truck FH16

The XC90 is equipped as follows: a) Applanix GNSS/INSS unit providing the car position data in GPS coordinates. This

---

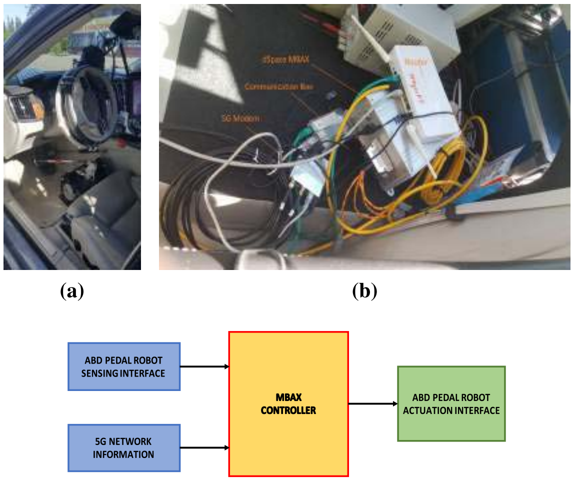
[4] A frequency of 20Hz has been used in the experiments



(a) (b)

(c) (d)

Fig. 5. Experimental Setup: a) Outside Equipment of the Volvo XC90; b) Inside Equipment of the Volvo XC90; c) Picture of the Volvo Truck FH16; d) Schematic overview of the software architecture executed on OpenDLV.



(a) (b)

(c)

Fig. 6. Equipment of the Volvo S90: a) Detail of the ADB Pedal Robot; b) Details of the on-board equipment; c) Schematic overview of the software architecture executed on the dSpace MicroAutobox.

sensor is combined with a Radio modem to gain Real-Time Kinematic (RTK) corrections, thus achieving a precision up to centimeters in data position [34]. b) Inertial Movement Unit (IMU) providing the current vehicle acceleration. Velocity measurements are obtained from the on-board commercial Electronic Control Unit (ECU) via the Controller Area Network (CAN) Interface. c) Pre-5G Telit Modem LTE+, a 5^{th} generation modem establishing the radio communication with the Ericsson test network. d) Roof antennas for sharing information over the Ericsson test network. e) A PC running the OpenDLV (see Section VII-C) software, under a GNU/Linux based operating system (ArchLinux) processing the sensors measurements and implementing the control law in equation (6). All the on-board sensors and actuators are connected, through a Local Area Netowrk (LAN), to the PC and exchange data through a UDP Multicast session. The Actuation Interface on OpenDLV provides the appropriate commands to the powertrain controller, that finally actuates the throttle and/or brake system of the XC90 (see details of the hardware configuration in Figs. 5a and 5b; the software architecture, executed on OpenDLV, is instead depicted in Fig. 5d). With respect to the FH16 in Fig. 5c, the on-board equipment and the software architecture executed on OpenDLV are similar to the one of the XC90 (see Figs. 5b and 5d). Indeed, the only difference is in the GNSS/INSS unit providing position data GPS coordinates, that for the truck is the Oxford OXTS GNSS (again combined with a Radio modem for RTK corrections).

B. Volvo Car S90

The S90 is equipped with an ADB Pedal Robot (shown in Fig. 6a) for actuating the control action calculated by the cooperative protocol. The controller action is on-board computed via the dSpace Micro Autobox (MBAX), a real-time platform interconnected with the vehicle and the on-board equipment for cooperative driving via the CAN and the Local Area Network, respectively. Namely, the ADB Pedal Robot drives the acceleration and braking systems of the vehicle (through mechanical actuators on the pedals) tracking the driving profile provided by the cooperative strategy. Moreover, the robot has a direct connection to the vehicle GNSS unit, IMU and CAN and communicates with the dSpace MBAX. The cellular communication is guaranteed by the pre-5G Telit Modem and Roof Antenna. Additional specific on-board devices also include: a) Communication Box, implemented on a Raspberry PI that is opportunely programmed and deployed to receive and convert data from the Ericsson test network so that they are readable from the dSpace MBAX. b) On-board Switch for providing the in-vehicle LAN. Details of both hardware configuration and software architecture, executed on MBAX, are shown in Figs. 6b and 6c, respectively. Specifically, the MBAX Controller has been prepared to run a Matlab/Simulink scheme whose main aim is to gather information about the state of vehicles, merge it with traffic data coming from the cloud and compute a control output for the *ADB*. The operation of the dSpace MBAX can be hence summarized as follows: i) receiving current states of Volvo S90 from the *ADB*; ii) gathering traffic information from the communication box; iii) computing the control input and sending actuation signals to the *ADB*; iv) communicating current known states, through the communication box, to the cloud.

C. OpenDLV Communication Module

OpenDLV is a modern open source software environment to support the development and testing of self-driving vehicles. It has been implemented using high quality and modern C++14 with a strong focus on code clarity, portability, and performance. In addition, it is entirely based on micro-services, usually run in separate docker [35] containers. For a more comprehensive treatment, we refer the interested reader to the specific literature ([31], [32]). In order to extend the communication abilities of our test cars to 5G, an additional module has been designed and developed, in accordance with the OpenDLV specification. The *OpenDLV Standard Message Set* has been hence expanded to account for sensing information

Fig. 7. The city area at AstaZero.

from other vehicles. Among the added message properties, we find the number of connected nodes (i.e., the number of current active nodes at the intersection), the sequence number of the packet and a so-called *whoami* field that each vehicle uses to identify itself within a fleet. The sequence number is used to discard packets received out of sequence. The OpenDLV receiver keeps track of received packets: if the current received packet has a lower sequence number than the last packet received, it will be discarded and won't be replayed in the UDP-based OpenDLV session. The OpenDLV communication module has been thought to run in a container within an OpenDLV session. In particular, this module is able to exchange data with other OpenDLV components such as the *proxy interface* to the car CAN bus and the *Applanix GPS*.

D. Hermes Traffic Monitor

A web interface has been built that allows a user to monitor the status of the traffic, by providing details about all connected vehicles. Thanks to the generalized structure of the communication software and the standardization of most of the *traffic management* system, the development of this component took a minimal amount of effort and time.

VIII. EXPERIMENTAL VALIDATION

A. Illustrative Driving Scenario

The tests have been executed at the City Area[5] of the AstaZero[6] (near Gothenburg, Sweden). The City area consists of an intersection with streets, of varying widths and lanes, equipped with bus stops, pavements, street lighting, and building backdrops. The road system, allowing different kinds of driving tests, includes roundabouts, T-junctions and return-loops. Connections to a rural road occur in two places. The area has a relatively flat surface with dummy blocks that resemble buildings and host some technical aids such as radars (see Fig. 7). One of the blocks also contains space for a control room and a warehouse for dummies. The map of the City Area exploited for the tests is reported in Fig. 8. Here, the Cooperative Zone (CZ) of interest is marked with a blue circle, while the Conflicting Area (CA) (at the intersection center) is marked with a red circle. Different experimental runs were performed in different driving conditions.

In what follows we will first describe how we performed preliminary trials in a so-called *multilane* scenario allowing us to safely simulate a real-world intersection. We will then move to the actual street junction scenario, for which we will report some of the experimental results related to the case when the vehicles, initially located as in Fig. 8, access the CZ with initial velocities and relative positions that would lead to collision without any control action. This exemplar scenario also considers mixed traffic. Namely, the Volvo XC90 and S90 are fully automated, while the Volvo Truck F16 is human-driven, yet connected, i.e., it shares information about its actual position and speed.

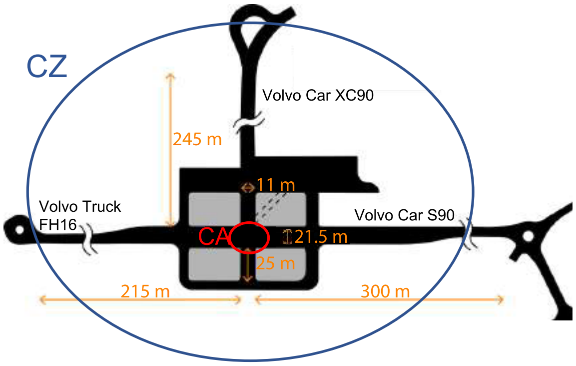


Fig. 8. Map of the city area at AstaZero.

B. Experimental Campaign

Our coordination system has been experimentally validated in both multi-lane and street junction experiments. A multi-lane experiment is a safe real emulation of a street intersection where roads leading to the intersection are projected parallel to each other. The involved vehicles will drive in parallel while approaching a designated area, resembling the intersection, which they must access according to a mutual exclusion policy. This scenario is of the utmost importance since it allows to test control algorithms in a highly realistic situation, taking account of the actual delay introduced by centralized communication hardware and software, without the risk that vehicles will collide. Once results have been validated with the multilane experiments, they can easily be replicated on a real intersection without risking collisions. It must be highlighted that buildings placed at the corners of the intersection constitute an obstacle against both the human driver's eye and a virtual direct *Vehicle-to-Vehicle* communication. Under this assumption, in fact, each vehicle would not be able to see hidden mobile nodes approaching the crossroad until the very last moment, without being connected to the 5G PoC.

C. Outcomes

In both classes of experiments, multilane and intersection, the overall system has successfully demonstrated its capacity of managing the negotiation of street junctions over pre-5G PoC, using LTE radio with 5G EPC. In the following sections, the results obtained in the real intersection scenario will be illustrated and discussed. We discuss the results for just one class of experiments since the two classes are identical from a scientific point of view. The choice of executing the multilane set of trials before moving to the real intersection scenario just depends on reasons related to the safety of people inside the cars.

---

[5] http://www.astazero.com/the-test-site/test-environments/city-area/

[6] http://www.astazero.com

IX. NETWORK PERFORMANCE

The proposed model and framework (Sections IV and V) have been successfully implemented, deployed and tested on three vehicles at the AstaZero (AZ) proving ground. Communication among entities has been enabled by the pre-5G PoC test network provided by Ericsson. The pre-5G PoC, using LTE radio with 5G EPC, offers a full radio coverage of the City Area (CA) of AstaZero, as outlined in Fig. 8. In order to reduce control and management times, the distributed cloud network is installed within the boundaries of the proving ground itself. Experiments discussed in this work date back to March 2018 and the Ericsson test network has evolved meanwhile towards 5G NR. The aim of this section is to provide the experimental evidence that the key functional and performance requirements of the communication network, necessary in cooperative driving and intersection crossing services, are satisfied [36], [37].

A. Preliminary Latency Analysis

In early March 2018, performance of the network has been measured in terms of delay and latency. Early measurements have been carried out with the aim of understanding the impact of the network on the overall communication performance. The analysis enabled us to design a communication software that could meet the safety timing constraints required by the involved control algorithms. The network flow between two laptops connected to the test network showed an average TCP Round Trip Time (RTT) of $24ms$ with Standard Deviation (STD) of 0.028. The RTT has been measured using the TCP Acknowledgement segment. In addition, the measurements of the time between two consecutive frames showed an inter-frame latency of $24ms$, along with 0.027 standard deviation. We need to point out that measurements of our interest have been taken at software level, and that the air-interface latency is a small part of the RTT we measured. Finally, a sequence number analysis has been conducted via a Stevens Graph (omitted for the sake of brevity) which shows the progression of packets sequence numbers versus time. This suggests that there are negligible timeout events in the communication. From this analysis it is reasonable to assume that the performance offered by the network, including delay, meets control constraints and potentially enables a safe and reliable communication between a set of mobile nodes and a remote endpoint on the ground.

B. Delay Components

Three kinds of delays have been taken into account when measuring the performance offered, through the pre-5G PoC, by the communication software designed for our vehicles. Firstly, since the conceived *Traffic Management* protocol has been designed to rely on TCP, we wanted to measure the impact of the chosen transport protocol on the overall communication. Moving up along the ISO/OSI stack, two additional delay contributions have been considered, namely the *Application Layer ACK* (WS ACK) and the so-called *State RTT*, as outlined in Figure 9. The former is the acknowledgement time of a websocket packet at the application level. The latter measures the interval between the time a vehicle sends its state and the time it receives the same state reflected from the traffic manager, in the form of a *traffic update* message. The *State RTT* is of utmost importance, since it represents the delay that is actually impacting the control algorithm. Also in this case, we want to remark that measurements of our interest have been taken at software level, and that the air-interface latency is a small part of the RTT we measured.

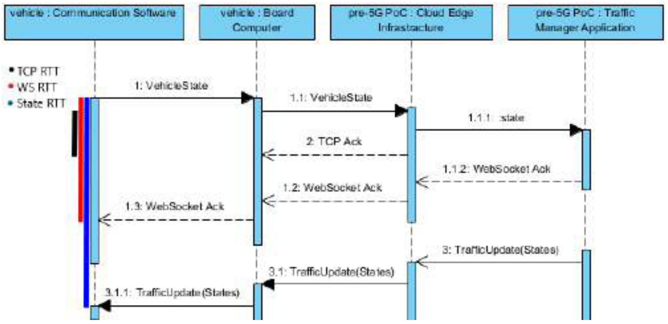


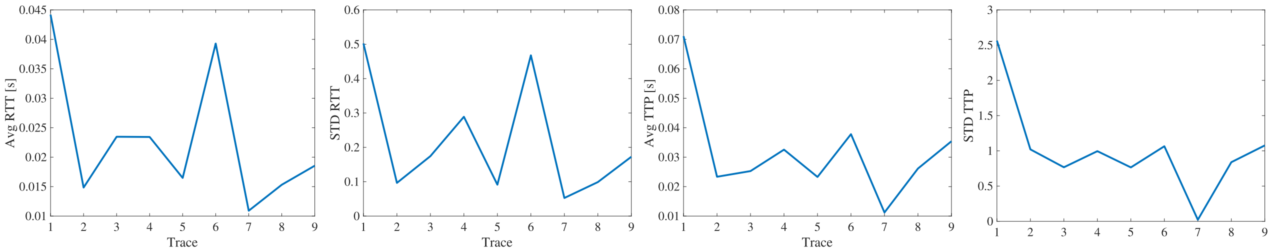
Fig. 9. Contributions to measured delays.

C. TCP Analysis

During the experiments, nine TCP traces have been collected. As envisaged, their analysis shows very low delays along the whole experimental session. In particular, two features have been considered at this stage: Round Trip Time (RTT) and Time From Previous Frame (TTP), whose aggregated behavior during the entire duration of the experiments is reported in Fig. 10. A more detailed diagram has been reported for Traces 1 and 8 in Fig. 11. With respect to the *RTT*, the average value oscillates between 15 and 45 ms. Furthermore, the average time between two subsequent TCP frames received oscillates between 10 and 40 ms with a peak of 70 ms on the first trace. It is important to point out that the nine traces differ between one another in number of packets. In particular, trace 1 stores a number of packets whose order of magnitude is 10^5, while the others store a number of packets with orders of magnitude between 10^3 and 10^5. A noticeable improvement has been observed since the preliminary tests, due to hardware and software improvements that have been applied to the test network. Outliers in the diagrams are negligible as their number is infinitesimal compared to the overall number of transmitted packets. Reliability of the system is hence not impacted. To this extent, the Stevens Diagram for Trace 1, omitted for the sake of brevity, does not show any major issues related to communication.

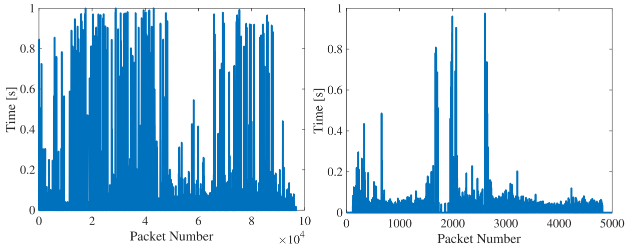
D. Communication Software Performance

As mentioned above, one parameter for classifying the performance perceived at application level is the RTT ACK. Measurements report a value which is bounded within the interval $[0 - 50]$ ms. As a matter of fact, these data are quantitative and cannot be considered 100% reliable (0 seconds delay is not realistic). However, they provide coarse-grained information about the order of magnitude of the measured indicator. Such lack of precision can be ascribed to two main factors: a) *Software*: The acknowledgement time has been measured, on the vehicles, via the `ack-callback` provided by the C Websocket Library, which is known not to be triggered in a strictly real-time fashion; b) *Operating*



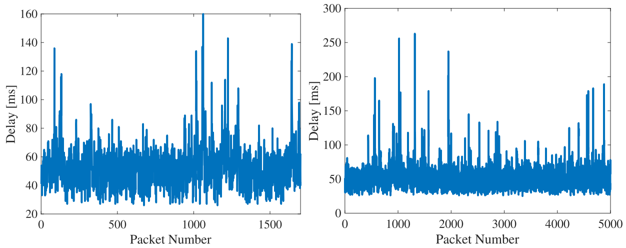
(a) (b) (c) (d)

Fig. 10. Nine TCP traces recorded during the experiments: a) average TCP Round Trip Time (RTT); b) RTT Standard Deviation normalized to $N_{sample} - 1$; c) average Time From Previous Frame (TTP); d) TTP Standard Deviation normalized to $N_{sample} - 1$.



(a) (b)

Fig. 11. Round Trip Time TCP Delay during two of the nine TCP traces we have recorded: a) Trace 1; b)Trace 8.



(a) (b)

Fig. 12. State RTT: a) for one experiment; b) for a series of experiments conducted in the real intersection scenario.

System: The acknowledgement is measured at a really high level in the operating system. With the above considerations in mind, we can nonetheless safely state that the application Level RTT ACK stays below the envisaged design threshold. On the other hand, the parameter used to measure the actual delay impacting on the control algorithm is the *State RTT*. It is indeed, for a vehicle, the window between a *status message* sent to the traffic manager and the very same message received as *traffic information* from the traffic controller. This delay is measured at application level and it is the ultimate, composed, delay that might affect the control algorithm. The control action is indeed estimated using, as input, the latest neighbours' state received. To this extent, the safety of the control algorithm must be proved against this value. For this kind of delay, our measurements have reported an average value of $70ms$. This ensures that the system behaved in a reliable manner for the entire duration of the experiment. Fig. 12a reports the state RTT delay curve in the case of a single experiment. Fig. 12b, on the other hand, aggregates the results of all of the experiments we conducted in the real intersection scenario. In both cases, the outliers can be ignored since, by design, we imposed the rule to discard so-called *late predecessor* packets, i.e., packets with lower sequence numbers arriving at a node that has already successfully received a packet belonging to the same stream and carrying a higher sequence number. Therefore, the experimental results prove that the pre-5G network is capable of supporting fully-distributed control protocols, based on the cyber-physical interactions of vehicles, for the safe intersection crossing in the very challenging use case when vehicles are driving in the absence of any signalizing system or predetermined traffic rule. More specifically, according to [36], [37] the following requirements are satisfied: i) end-to-end latency within the range $[3; 100]ms$; ii) reliability of 99%; iii) data rate within the range $[10; 5000]\ kbps$; iv) short to medium communication range.

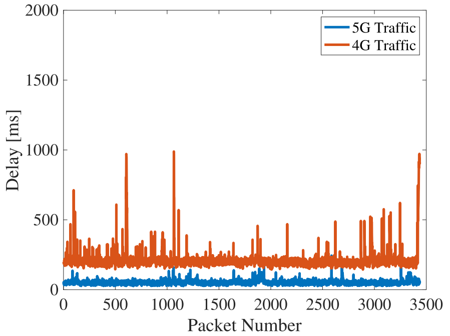
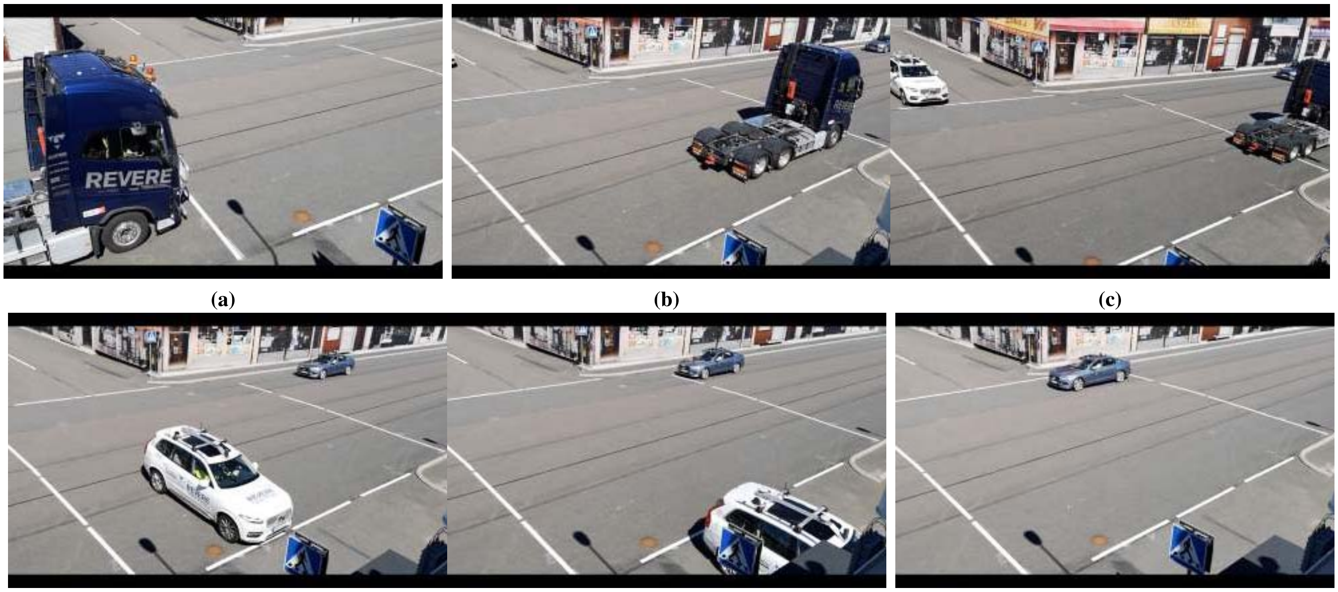


Fig. 13. Comparing results with 4G Public cloud: In red it is plotted the trend of delays registered on a public cloud against the timing we collected on our network (blue).

E. Further Considerations

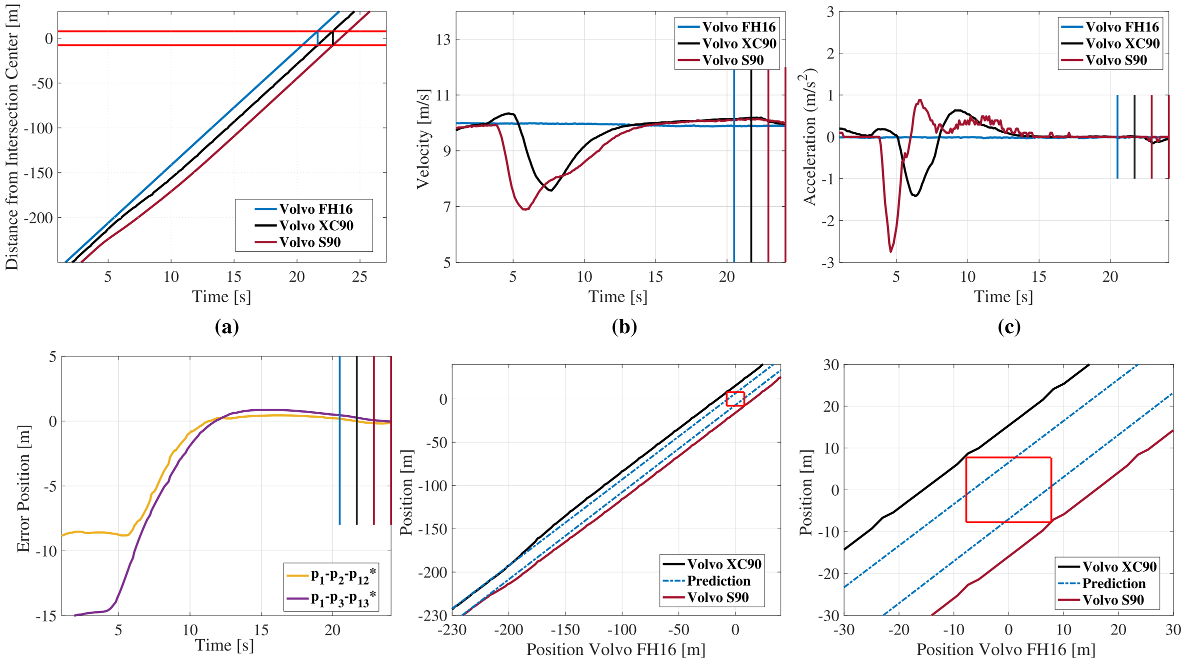
As part of our trials, we were also interested in investigating the performance increase deriving from the adoption of a 5G-enabled network using Edge cloud technology. All other things being equal, we ran the same experiments over a publicly reachable LTE infrastructure. Results are reported in Fig. 13 and clearly show that the measured delay, in case of the public infrastructure, has more than doubled, with an average value that is close to $200ms$. While the observable difference in performance is mostly due to the difference between an application server deployed at the edge and a more distant one, an interesting direction to further explore is to analyze the performance of our framework over a non dedicated network supported by state of the art technology. For the sake of completeness, we remark that we did not carry out a statistically reliable set of comparative trials. Since we did not get enough LTE data to claim statistically relevant results, the presented graph should be taken with a grain of salt. It nonetheless gives an idea as of the qualitative performance trend in the presence of the two mentioned communication



(a) (b) (c)

(d) (e) (f)

Fig. 14. Snapshots of autonomous crossing over 5G.



(a) (b) (c)

(d) (e) (f)

Fig. 15. Experimental Results: a) Time history of the position (beginning and end of the CA: solid horizontal lines); b) Time history of the vehicle velocity; c) Time history of the vehicle acceleration; d) Time history of the position errors w.r.t. desired inter-vehicle gaps; e) Position of the 2^{nd} vehicle and the 3^{rd} vehicle, vs position of the 1^{st} vehicle (colliding area: rectangular area; theoretical trajectories without the control correction: dashed line); f) Position of the 2^{nd} vehicle and the 3^{rd} vehicle, vs position of the 1^{st} vehicle: Zoom on the colliding area. In figures a), b), c) and d) the blue vertical line indicates the time instant at which the 1st vehicle enters the CA; the black vertical line indicates the time instant at which the 1^{st} vehicle exits and 2^{nd} enters the CA; the bordeaux vertical line indicates the time instant at which the 2^{nd} vehicle exits and 3^{rd} enters the CA; the second bordeaux vertical line indicates the time at which the 3^{rd} vehicle exits the CA.

infrastructures. As a final remark, we mentioned that the Traffic Manager dispatcher is deployed on a server cluster at the edge of the cellular network, geographically close to the path-way. Such proximity aims to minimise round trip time and latency in general. When scaling towards a production scenario, we made the implicit assumption that additional delays introduced by inter-operator communications are negligible. However we recognise that, in some scenarios, this issue can become relevant. In such cases, Software Defined Networks (SDN) and Network Functions Virtualization (NFV) can be effectively exploited to enhance inter-operator communications and minimise end to end delays [38].

TABLE II

EXPERIMENTAL SCENARIO PARAMETERS

Parameters	**Values**
Positions initial condition $[m]$ $[p_1(0), p_2(0), p_3(0)]^\top$	$[-220, -235, -250]^\top$
Velocities initial condition $[m/s]$ $[v_1(0), v_2(0), v_3(0)]^\top$	$[10, 9.7, 9.8]^\top$
Control gain α	0.1
Vehicle length L_i $[m]$	$L_1 = 7.8$ $L_2 = L_3 = 4.6$
Headway time h $[s]$	0.8
Distance at standstill r_{ij} $[m]$	10

F. Results From Cooperative Crossing Driving Tests

Results in fig. 15 show the effectiveness of the finite-time cooperative control protocol in guaranteeing the safe crossing at an intersection, with communication happening over 5G (parameters values characterizing the experimental scenarios are summarized in Table II). Full video of the experiments can be found at *https://www.youtube.com/watch?v=rmjJkIlFMJ4*, while some snapshots are in Fig. 14. Specifically, the time histories of positions, velocities, and accelerations reported respectively in fig. 15a, fig. 15b, and fig. 15c, confirm that the cooperative control protocol guarantees the exclusive vehicles access into the CA, whose boundaries are indicated with solid red horizontal lines in fig. 15a. Indeed, only when the first vehicle has exited the CA, the second vehicle is just ready to enter the intersection (as highlighted by the vertical solid line). The achievement of the required inter-vehicle spacing correctly matches with the reaching of a common velocity (see fig. 15b). According to the theoretical derivation, the time history of the position error converges to zero with a finite settling time T of about 20 $[s]$ (see fig. 15d). Hence, the cooperative algorithm safely converges in a finite time before the first vehicle accesses the CA, hence ensuring collision avoidance at the junction. To better appreciate the effectiveness of the proposed control approach in avoiding collisions, in fig. 15e and fig. 15f the position of both the second and the third vehicle, i.e. $p_2(t)$ and $p_3(t)$, are plotted against the position of the first vehicle, i.e. $p_1(t)$. Here the red square area represents the set of positions for which collision occurs and, as it can be easily observed, both trajectories (solid lines) tangentially touch the critical colliding area, indicated by the red square, so it follows that, as soon as a vehicle exits the CA, the next one is just ready to enter. Conversely, the ideal trajectories (dashed-dotted line), i.e. the unsafe trajectories that vehicles would have followed if their initial velocities would have been held without any correction, uncover the occurrence of collisions in the absence of control.

X. CONCLUSION

In this article we have successfully modelled, implemented, deployed and tested an autonomous driving system operating in proximity of a street intersection over a pre-5G test network. Experiments have been carried out in urban scenarios where buildings and objects of different shapes and materials act as obstacles against most used sensors, and point to point communication paradigms. The control algorithm has been thought and designed in a distributed way, with the objective of minimizing the impact of transmissions and the computational load on remote devices. Network performance, along with control results, has been analyzed and discussed.

ACKNOWLEDGMENT

Experiments for this project have been realized in a tight cooperation with Ericsson AB and REVERE Lab (Chalmers). Special thanks to Keerthi Kumar Nagalapur, Henrik Sahlin, Magnus Castell and Anders Aronsson, from the Ericsson Team, for the great ideas, the constant help, and the frequent discussions. Also, they would like to thank Christian Berger, Arpit Karsolia, and Fredrik von Corswant from Revere LAB and Niklas Lundin from AstaZero. Without their contribution and passion, their experiments would have never been possible.

REFERENCES

[1] B. Liu, B. Otis, S. Challa, P. Axon, C. Chou, and S. Jha, "On the fading and shadowing effects for wireless sensor networks," in *Proc. IEEE Int. Conf. Mobile Ad Hoc Sensor Syst.*, Oct. 2006, pp. 51–60.

[2] J. Rios-Torres and A. A. Malikopoulos, "A survey on the coordination of connected and automated vehicles at intersections and merging at highway on-ramps," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 5, pp. 1066–1077, May 2017.

[3] M. A. S. Kamal, J.-I. Imura, T. Hayakawa, A. Ohata, and K. Aihara, "A vehicle-intersection coordination scheme for smooth flows of traffic without using traffic lights," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 3, pp. 1136–1147, Jun. 2015.

[4] Y. Zheng, S. E. Li, K. Li, F. Borrelli, and J. K. Hedrick, "Distributed model predictive control for heterogeneous vehicle platoons under unidirectional topologies," *IEEE Trans. Control Syst. Technol.*, vol. 25, no. 3, pp. 899–910, May 2017.

[5] C. Liu, C.-W. Lin, S. Shiraishi, and M. Tomizuka, "Distributed conflict resolution for connected autonomous vehicles," *IEEE Trans. Intell. Vehicles*, vol. 3, no. 1, pp. 18–29, Mar. 2018.

[6] M. Zanon, S. Gros, H. Wymeersch, and P. Falcone, "An asynchronous algorithm for optimal vehicle coordination at traffic intersections," in *Proc. Conf. IFAC World Congr.*, Aug. 2017, pp. 12008–12014. [Online]. Available: https://www.youtube.com/watch?v=fzkv5beS4uk

[7] R. Hult, M. Zanon, S. Gros, and P. Falcone, "Primal decomposition of the optimal coordination of vehicles at traffic intersections," in *Proc. IEEE 55th Conf. Decis. Control (CDC)*, Dec. 2016, pp. 2567–2573. [Online]. Available: https://www.youtube.com/watch?v=fzkv5beS4uk

[8] S. A. A. Shah, E. Ahmed, M. Imran, and S. Zeadally, "5G for vehicular communications," *IEEE Commun. Mag.*, vol. 56, no. 1, pp. 111–117, Jan. 2018.

[9] L. Chen and C. Englund, "Cooperative intersection management: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 2, pp. 570–586, Feb. 2016.

[10] G. Chartrand, *Introduction to Graph Theory*. New York, NY, USA: McGraw-Hill, 2006.

[11] S. P. Bhat and D. S. Bernstein, "Finite-time stability of continuous autonomous systems," *SIAM J. Control Optim.*, vol. 38, no. 3, pp. 751–766, Jan. 2000.

[12] Z. Sun, S. Mou, M. Deghat, and B. D. O. Anderson, "Finite time distributed distance-constrained shape stabilization and flocking control for d-dimensional undirected rigid formations," *Int. J. Robust Nonlinear Control*, vol. 26, no. 13, pp. 2824–2844, Sep. 2016.

[13] E. Coelingh and S. Solyom, "All aboard the robotic road train," *IEEE Spectr.*, vol. 49, no. 11, pp. 34–39, Nov. 2012.

[14] W. Wu, J. Zhang, A. Luo, and J. Cao, "Distributed mutual exclusion algorithms for intersection traffic control," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 1, pp. 65–74, Jan. 2015.

[15] L. Li and F.-Y. Wang, "Cooperative driving at blind crossings using intervehicle communication," *IEEE Trans. Veh. Technol.*, vol. 55, no. 6, pp. 1712–1724, Nov. 2006.

[16] A. I. M. Medina, N. van de Wouw, and H. Nijmeijer, "Cooperative intersection control based on virtual platooning," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 6, pp. 1727–1740, Jun. 2018.

[17] M. D. Vaio, P. Falcone, R. Hult, A. Petrillo, A. Salvi, and S. Santini, "Design and experimental validation of a distributed interaction protocol for connected autonomous vehicles at a road intersection," *IEEE Trans. Veh. Technol.*, vol. 68, no. 10, pp. 9451–9465, Oct. 2019.

[18] S. Feng, Y. Zhang, S. E. Li, Z. Cao, H. X. Liu, and L. Li, “String stability for vehicular platoon control: Definitions and analysis methods,” *Annu. Rev. Control*, vol. 47, pp. 81–97, Jan. 2019.

[19] Z. Zuo, Q.-L. Han, B. Ning, X. Ge, and X.-M. Zhang, “An overview of recent advances in fixed-time cooperative control of multiagent systems,” *IEEE Trans. Ind. Informat.*, vol. 14, no. 6, pp. 2322–2334, Jun. 2018.

[20] Y. Li and C. Tan, “A survey of the consensus for multi-agent systems,” *Syst. Sci. Control Eng.*, vol. 7, no. 1, pp. 468–482, 2019.

[21] J.-J. E. Slotine *et al.*, *Applied Nonlinear Control*, vol. 199, no. 1. Englewood Cliffs, NJ, USA: Prentice-Hall, 1991.

[22] A. Petrillo, A. Pescapé, and S. Santini, “A collaborative approach for improving the security of vehicular scenarios: The case of platooning,” *Comput. Commun.*, vol. 122, pp. 59–75, Jun. 2018.

[23] G. Fiengo, D. G. Lui, A. Petrillo, S. Santini, and M. Tufo, “Distributed robust PID control for leader tracking in uncertain connected ground vehicles with V2 V communication delay,” *IEEE/ASME Trans. Mechatronics*, vol. 24, no. 3, pp. 1153–1165, Jun. 2019.

[24] A. Petrillo, A. Pescape, and S. Santini, “A secure adaptive control for cooperative driving of autonomous connected vehicles in the presence of heterogeneous communication delays and cyberattacks,” *IEEE Trans. Cybern.*, early access, Jan. 2020, doi: 10.1109/tcyb.2019.2962601.

[25] Y. Zhao, Z. Duan, G. Wen, and G. Chen, “Distributed finite-time tracking of multiple non-identical second-order nonlinear systems with settling time estimation,” *Automatica*, vol. 64, pp. 86–93, Feb. 2016.

[26] NGA World Geodetic System Website of the NGA. (Apr. 2012). *National Geospatial-Intelligence Agency*. Accessed: Apr. 2, 2012. [Online]. Available: https://web.archive.org/web/20120402143802/, https://www1.nga.mil/productsservices/geodesyandgeophysics/, and worldgeodeticsystem/pages/default.aspx

[27] P. Popovski *et al.*, “Wireless access for ultra-reliable low-latency communication (URLLC): Principles and building blocks,” 2017, *arXiv:1708.07862*. [Online]. Available: http://arxiv.org/abs/1708.07862

[28] G. Xylomenos and G. C. Polyzos, “TCP and UDP performance over a wireless LAN,” in *Proc. INFOCOM*, 1999, pp. 439–446.

[29] *IETF*. Accessed: Jun. 24, 2019. [Online]. Available: https://tools.ietf.org/html/rfc6455rfc6455

[30] Node.js Monitoring. *Alerting and Reliability 101 Ebook*. [Online]. Available: https://risingstack.com/resources/node-js-monitoring-ebook

[31] C. Berger, B. Nguyen, and O. Benderius, “Containerized development and microservices for self-driving vehicles: Experiences & best practices,” in *Proc. IEEE Int. Conf. Softw. Archit. Workshops (ICSAW)*, Apr. 2017, pp. 7–12.

[32] O. Benderius, C. Berger, and V. M. Lundgren, “The best rated human–machine interface design for autonomous vehicles in the 2016 grand cooperative driving challenge,” *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 4, pp. 1302–1307, Apr. 2018.

[33] *Vehicle Pedal Robots*. [Online]. Available: https://www.abdynamics.com/resources/files/SP6031-i6-Pedal-Robots-Outline.pdf

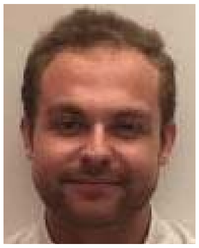
[34] L. Wanninger, “Introduction to network RTK,” *IAG Work. Group*, vol. 4, no. 1, pp. 2003–2007, 2004.

[35] D. Merkel, “Docker: Lightweight Linux containers for consistent development and deployment,” *Linux J.*, vol. 2014, no. 239, p. 2, 2014.

[36] M. Boban, A. Kousaridas, K. Manolakis, J. Eichinger, and W. Xu, “Use cases, requirements, and design considerations for 5G V2X,” 2017, *arXiv:1712.01754*. [Online]. Available: http://arxiv.org/abs/1712.01754

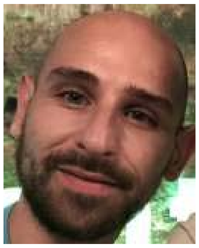
[37] *Technical Specificationgroup Services and System Aspects. Enhancement of 3GPP Support for V2X Scenarios*, document v16.0.0., Release 15, Sep. 2018.

[38] G. Biczok, M. Dramitinos, L. Toka, P. E. Heegaard, and H. Lonsethagen, “Manufactured by software: SDN-enabled multi-operator composite services with the 5G exchange,” *IEEE Commun. Mag.*, vol. 55, no. 4, pp. 80–86, Apr. 2017.



Luca Maria Castiglione received the M.Sc. degree in computer science and engineering from the University of Napoli Federico II. He is currently pursuing the Ph.D. degree with the Resilient Information Systems Security (RISS) Group, Imperial College London. His work is at the intersection between safety and security of cyber physical systems. In particular, he investigates techniques to ensure the continuous delivery of critical functionalities when systems are under attack and have been partially compromised.

Paolo Falcone (Member, IEEE) received the Ph.D. degree in information technology from the University of Sannio, Benevento, Italy, in 2007. He is currently an Associate Professor with the Department of Signals and Systems, Chalmers University of Technology, Sweden. His research interests include constrained optimal control and verification methods, applied to autonomous and semi-autonomous mobile systems, cooperative driving, and intelligent vehicles. He is involved in a several projects, in cooperation with industry, focusing on autonomous driving, cooperative driving, and vehicle dynamics control.



Alberto Petrillo received the Ph.D. degree in control systems engineering from the University of Naples Federico II in 2019. He is currently a Research Fellow with the University of Naples Federico II. His research interests include distributed, synchronization-based, control of networked control systems in the presence of communication impairments and security vulnerabilities, with application to automotive field.

Simon Pietro Romano (Member, IEEE) is currently an Associate Professor with the Department of Electrical Engineering and Information Technology, University of Napoli Federico II. He teaches computer networks, computer architectures, network security, and telematics applications. He is also the Co-Founder of Meetecho, a startup and university spin-off dealing with scalable video streaming and WebRTC-based unified collaboration, as well as of SECurity Solutions for Innovation (SECSI) that focuses on network security. He actively participates in Internet Engineering Task Force (IETF) standardization activities, mainly in the applications and real time (ART) area.

Stefania Santini (Senior Member, IEEE) is currently an Associate Professor of automatic control with the University of Napoli Federico II. She is involved in many projects with industry, including small- and medium-sized enterprises operating in the automotive field. Her research interests include analysis and control of nonlinear systems with applications to automotive engineering, transportation technologies, and, more recently, computational biology.