Transportation Research Part C 124 (2021) 102905

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

Multi-layer multi-rate model predictive control for vehicle platooning under IEEE 802.11p

Amr Ibrahim [a,*], Dip Goswami [a], Hong Li [b], Iñaki Martín Soroa [a], Twan Basten [a,c]

[a] Electrical Engineering, Eindhoven University of Technology, Eindhoven, the Netherlands
[b] Car Infotainment & Driving Assistance, NXP Semiconductors, Eindhoven, the Netherlands
[c] ESI, TNO, Eindhoven, the Netherlands

ARTICLE INFO

Keywords:
V2V communication
Vehicle platooning
Multi-layer control
Multi-rate control
Model predictive control
Embedded implementation

ABSTRACT

Vehicle platooning has gained attention for its potential to increase road capacity and safety, and higher fuel efficiency. Platoon controls are implemented over Vehicle-to-Vehicle (V2V) wireless communication, in-vehicle networks and Electronic Control Units (ECUs). V2V communication has a low message rate imposed by the V2V standard compared to the rate of modern in-vehicle networks and ECUs. The platoon control strategy should take into account such *multi-rate* nature of the implementation architecture for higher performance. Current literature does not explicitly consider such real-life constraints. We propose a two-layered control framework for vehicle platoons wirelessly communicating complying with the industrial standard IEEE 802.11p. In the upper-layer, vehicles receive state information from the immediate preceding vehicle over a control channel at 10 Hz under the IEEE 802.11p standard with occasional packet drops. Using such information and the vehicle state information, the engine control system, i.e. the lower-layer, realizes the desired vehicle state (e.g., acceleration) over the fast and reliable in-vehicle networks (e.g., FlexRay, Ethernet). In this work, a distributed model predictive control framework is proposed for the upper-layer targeting a Predecessor-Follower (PF) topology. A state-feedback control scheme is proposed for realizing the desired vehicle states for the lower-layer. Our framework minimizes the inter-vehicle distance and the tracking error enforcing collision avoidance and robustness against packet drops at the upper-layer. We validate our algorithm in simulation using our co-simulation framework CReTS and on an embedded platform, developed by Cohda Wireless and NXP, running in real time and communicating through the IEEE 802.11p standard. With extensive simulations and experiments, we evaluate the performance and feasibility of the proposed framework under a number of practical constraints. Our approach is a step towards the implementation of platoon control in reality.

1. Introduction

Automated and cooperative driving are key technologies to improve safety, fuel efficiency, and reduce emissions (Kato et al., 2002; Lee, 1976). A group of autonomous vehicles that are closely following each other and maintaining a safe inter-vehicle distance is called a *platoon*. Cooperative Adaptive Cruise Control (CACC) which is an extension to Adaptive Cruise Control (ACC) is an enabling

* Corresponding author.
E-mail addresses: a.ibrahim@tue.nl (A. Ibrahim), d.goswami@tue.nl (D. Goswami), hong.r.li@nxp.com (H. Li), i.martin.soroa@gmail.com (I.M. Soroa), a.a.basten@tue.nl (T. Basten).

https://doi.org/10.1016/j.trc.2020.102905
Received 4 November 2019; Received in revised form 21 October 2020; Accepted 27 November 2020
Available online 17 January 2021

technology for vehicle platooning. In ACC, a vehicle uses on-board sensors e.g., radar or lidar, to measure the distance to the preceding vehicle and therefore keep a safe distance to it (Marsden et al., 2001). CACC enhances the functionality of ACC by integrating Vehicle-to-Vehicle (V2V) wireless communication between vehicles along with other sensors which enables significant reduction in *headway* time (i.e. the time needed by the follower vehicle to reach the position of the preceding vehicle; or in other words, the geometric distance divided by velocity). Wireless communication enables vehicles to share a richer set of information such as acceleration, position, velocity, road intersection and traffic flow status e.g. existence of moving or stationary obstacles. The design and functionality of CACC have been extensively studied in literature e.g. (Kato et al., 2002; Naus et al., 2009; Tiganasu et al., 2017; Zeng et al., 2019; Ploeg et al., 2011; Naus et al., 2010). Some studies have shown that the capacity of highways improves by increasing CACC market penetration (Van Arem et al., 2006).

A platoon control is realized by a set of software tasks running on the electronic control units (ECUs – automotive processors) communicating over in-vehicle networks and wireless V2V communication. Overall, it involves computation at the ECUs, in-vehicle and V2V communication. The ECUs run under automotive real-time operating systems (RTOSs) such as OSEK which are pre-configured and offer only a specified set of periods (e.g. 1 ms, 2 ms, 5 ms, 10 ms, etc. (OSEK Consortium, 2005)). Multiple ECUs are connected via high-speed and reliable networks such as FlexRay, CAN or Ethernet (Makowitz and Temple, 2006; Tuohy et al., 2014). In-vehicle architecture (ECUs and buses) generally supports a high sampling rate for the platoon controllers.

On the other hand, the platooned vehicles exchange periodic Cooperative Awareness Messages (CAMs) over V2V communication. As per IEEE 802.11p under the European Telecommunications Standards Institute (ETSI) (Jiang and Delgrossi, 2008), there are two types of channels – one control channel (CCH) and six service channels (SCHs). The SCHs allow for higher message rates e.g., 25 Hz, 50 Hz, 100 Hz. However, since SCHs are meant to be shared by multiple safety and infotainment related services, they may be congested and introduce packet drops and large delays when shared by a large number of vehicles. On the other hand, the CCH is dedicated to safety-critical applications like platoons. The CCH allows for a message rate of 10 Hz for CAMs when the channel load is less than 70% and it can be lowered to 1 Hz in case of high network congestion controlled by a so called Decentralized Congestion Control (DCC) algorithm (ETSI, 2014b). Data types contained in all generated CAMs shall include all fast-changing (dynamic) status information along with their measurement accuracy level (ETSI, 2011; ETSI, 2014a). Due to a higher message rate in SCHs, current CACC trials such as (Öncü et al., 2014; van Nunen et al., 2017) and the Grand Cooperative Driving Challenge (GCDC) (van de Sluis et al., 2015) used a dedicated SCH for V2V communication at 25 Hz message rate using a specialized transceiver. Additionally, recent works such as (Naus et al., 2010; Öncü et al., 2014; van Nunen et al., 2017; Dolk et al., 2017) consider the desired vehicle status to be sent over V2V communication instead of transmitting the actual vehicle status. Clearly, such special arrangement will not be accommodated in real-life platoons. We consider the CCH at 10 Hz for exchanging the actual vehicle status via CAMs assuming a channel load no more than 70% or DCC algorithm is not implemented. The challenge is to design the controller with this restricted message rate.

We adopt a multi-layered control scheme where the platoon control is divided into an upper-layer and a lower-layer. Having two separate control layers adds flexibility in platoon control design since each layer has a different role and therefore can be designed separately. Moreover, each layer uses a different communication standard with different operating frequency. The upper-layer in each vehicle receives CAM messages from its preceding vehicle (i.e., under Predecessor-Follower (PF) topology) at 10 Hz frequency. It calculates the vehicle's desired acceleration while ensuring safety (e.g. avoiding rear-end collision), fuel efficiency, driving comfort, tracking capability and better road capacity. Model Predictive Control (MPC) (Maciejowski, 2002) is the control method chosen for the upper-layer for its ability to handle different constraints on input and states. The desired acceleration is then passed over to the high-speed in-vehicle network e.g., FlexRay to be realized by the lower-layer controller. The role of the lower-layer is to reach the desired acceleration within a certain time by opening the throttle plate by specific values. We design the lower-layer controller using the state-feedback control method.

Current literature in dealing with a multi-layered platooning scheme is divided into two approaches: (i) merging the two layers into one layer where vehicle dynamics is totally ignored or highly simplified and therefore a perfect lower-layer controller is assumed e.g. (Kianfar et al., 2014; Zheng et al., 2016; Han et al., 2013; Dunbar and Caveney, 2011; Maxim et al., 2016; Zhou et al., 2012; Sancar et al., 2014; Xu et al., 2014; Liu et al., 2017) (ii) considering two layers with identical sampling rate e.g. (Li et al., 2010; Wang et al., 2015; Tiganasu et al., 2017; Zeng et al., 2019; Orosz, 2016; Turri et al., 2016; Öncü et al., 2014; Saxena et al., 2016; Öncü et al., 2012; Naus et al., 2010; Ploeg et al., 2011; van Nunen et al., 2017). Ignoring or simplifying vehicle dynamics as in approach (i) leads to a non-realistic platoon performance since the desired acceleration is assumed to be immediately realized. With approach (ii), the impact of the different in-vehicle and V2V communication standards is not taken into account. We develop a multi-rate multi-layered control scheme for vehicle platoons that realizes the proper role of the two layers and adheres to the real-life communication standards. To ensure realistic platoon performance, we consider a realistic yet simple vehicle model. Hence, we introduce a platoon model which realizes the fast vehicle dynamic model and the slow inter-vehicle dynamics. We use our co-simulation framework CReTS (ContRol, nEtwork and Traffic Simulator) (Ibrahim et al., 2018) to evaluate our platoon control framework under realistic network behavior. CReTS is a framework composed of network simulator ns-3, traffic simulator SUMO and Matlab. ns-3 provides packet reception ratios and delays to the controller implemented in Matlab. Different network congestion levels can be simulated by adjusting the number of communicating vehicles using SUMO and the corresponding control performance is evaluated.

Most theoretical works do not consider the computational constraints and real-time aspects that arise from the implementation of different control strategies on embedded platforms. The embedded implementation of MPC for vehicle platooning is challenging due to the computational requirements of MPC. We implemented the proposed controller on an embedded platform, developed by Cohda Wireless and NXP, with the aim to validate that the theoretical performance of our approach also translates to a good performance in a realistic scenario. We have created an evaluation setup with four devices communicating wirelessly using the IEEE 802.11p protocol. Each device emulates a different vehicle and runs in real time. This setup emulates the movement of the vehicle using a model and

emulates the measurement devices by adding noise to the predecessor's reported position. Our work shows that the actual implementation of our scheme is feasible.

Current literature makes several simplifying assumptions which hinder applicability of the presented techniques in real-life scenarios (without special safety arrangements). To this end, the contributions of our paper are:

1. We propose a multi-rate multi-layered control scheme for vehicle platooning while considering different communication standards with different sampling rates. To this end, we introduce a unified platoon model under PF topology that considers different sampling rates between V2V communication standard IEEE 802.11p and the in-vehicle network.
2. We consider 10 Hz as the message rate of the CAM messages in the control channel in compliance with the IEEE 802.11p standard. This makes our proposed method suitable to adopt in a real-life scenario.
3. Based on the overall platoon model, we develop a distributed MPC (DMPC) for the upper layer to achieve fuel efficiency, better road capacity and to ensure safety. We consider packet drops under realistic network behavior.
4. We perform a realistic evaluation and feasibility study using the CReTS framework and a Cohda embedded platform. We evaluate our controller under realistic wireless network behavior using both the network simulator ns-3 and the embedded devices that wirelessly communicate via the IEEE 802.11p.

This paper is organized as follows. Section 2 presents the related work. Section 3 gives an overview of the multi-layered control scheme. Section 4 introduces the vehicle model and the control design using state-feedback control. Section 6 introduces the V2V IEEE 802.11p communication standard and the platoon modelling under PF topology. Discretization of the platoon model is shown in Section 7 considering the multi-rate concept between layers. In Section 8 we present the MPC control design. In Section 9 we evaluate our approach and we show the control and network performance. Section 10 introduces the embedded implementation and the evaluation of our approach on the embedded platform. In Section 11 we conclude our paper.

2. Related work

Vehicle platooning is an example of a cyber-physical-system that integrates communication, control and computing technologies. In the following we discuss related work that is relevant to our control approach.

Spacing policy: The spacing policy for vehicle platooning can be either considered to be (i) a constant spacing headway where fixed distance between vehicles has to be maintained or (ii) a velocity-dependent headway where the gap between vehicles varies depending on velocity (Swaroop et al., 1994; Yanakiev and Kanellakopoulos, 1995). In this work we consider a velocity dependent headway, because this allows safety, higher fuel savings and better road usage.

Multi-layer control: Some earlier work combined upper and lower-layer control into a one-layered control scheme where the vehicle dynamics and lower-layer controller were ignored e.g. (Kianfar et al., 2014; Zhou et al., 2012; Zheng et al., 2016; Dunbar and Caveney, 2011; Maxim et al., 2016; Sancar et al., 2014; Han et al., 2013; Xu et al., 2014; Liu et al., 2017). The upper-layer controller (MPC for example) computes a vehicle's desired acceleration which is assumed to be immediately realized by the virtual vehicle's engine system (lower-layer). For example in Zhou et al. (2012), it is assumed that the lower-layer controller is already built and usable with a constant mechanical latency. A more realistic platoon behavior can be obtained by considering vehicle dynamics. A multi-layered control scheme is adopted in some works e.g. (Li et al., 2010; Wang et al., 2015; Turri et al., 2016). In Wang et al. (2015), the lower-layer controller for CACC is designed using a PID (proportional-integral-derivative) controller while the upper-layer is designed using PD (proportional-derivative) controller and MPC controller with a target to maintain a specified time gap between vehicles and to ensure safety and comfort. In Li et al. (2010) the lower layer controller compensates for the nonlinear vehicle dynamics and it tracks the desired acceleration while the upper-layer considers multiple objectives. In Turri et al. (2016), a two-layered control architecture is proposed where the upper and lower layers are combined into one layer (vehicle control layer). Another higher layer is proposed (centralized platoon coordinator), which computes the optimal fuel-efficient speed trajectory for the platoon vehicles which is then delivered to the vehicle control layer. In all these examples, the V2V communication between vehicles is assumed to be as fast as the in-vehicle control loop. In our work, we develop a multi-rate scheme that is compliant with the IEEE 802.11p standard.

Multi-rate control: There is extensive literature in control theory concerning multi-rate sampling control but not in the context of vehicle platooning; see for example (Goswami et al., 2013; Mizuochi et al., 2007). The work in Goswami et al. (2013), due to the limited set of sampling periods offered by the OS on an ECU, proposed a multi-rate switching control scheme which switches between different sampling periods offered by the OS in order to satisfy the required control objectives and reduce the processor load. The work in Mizuochi et al. (2007) considers the different sampling periods between sensing devices and actuating devices, and how to ensure stability of the overall system. None of these existing concepts can trivially be extended to the multi-rate concept for the control of vehicle platooning.

The current platooning literature deals with the rate difference arising from the different rates between the V2V network and the in-vehicle network by using a zero-order-hold (ZOH) as in Öncü et al. (2014). So the status of the preceding vehicle is repeated several times to compensate for the different communication rates. Another approach is proposed in van Nunen et al. (2017) where a buffer is designed to set the communication frequency (25 Hz) equal to the control-platform frequency (100 Hz) by using the same vehicle's status four times. Other approaches assume that upper and lower layers run with the same sampling rate as in Li et al. (2010), Wang et al. (2015), Turri et al. (2016), already discussed above. This assumption ignores the fact that the upper-layer runs much slower than the fast in-vehicle network and therefore, the system operates at slow sampling rate potentially compromising performance.

In this work, we propose, for the first time, a unified platoon model that takes into account the multi-rate nature arising from the

different in-vehicle sampling rates and the IEEE 802.11p V2V communication frequency.

Model predictive control: MPC for vehicle platooning has been widely studied in literature for ACC (Li et al., 2010; Eben Li et al., 2013; Luo et al., 2010; Zhao et al., 2017) and CACC (van Nunen et al., 2017; Kianfar et al., 2014; Zhou et al., 2012; Zheng et al., 2016; Wang et al., 2015; Dunbar and Caveney, 2011; Maxim et al., 2016; Sancar et al., 2014; Han et al., 2013; Turri et al., 2016). Among others, Distributed MPC (DMPC) is discussed in van Nunen et al. (2017), Kianfar et al., 2014, Maxim et al. (2016), whereas stability of DMPC was the focus of other studies e.g. (Zheng et al., 2016; Dunbar and Caveney, 2011). Multi-objective MPC is studied in literature for ACC (Li et al., 2010; Eben Li et al., 2013; Zhao et al., 2017) and CACC (Han et al., 2013).

In Maxim et al. (2016), a DMPC algorithm is implemented to achieve a constant spacing policy between platoon vehicles; vehicle dynamics are modeled as a double integrator. In Kianfar et al. (2014), van Nunen et al. (2017), DMPC is designed to satisfy certain constraints for ensuring safety, string stability and for not exceeding actuator limitations. Moreover in van Nunen et al. (2017), the future predictions of the intended acceleration computed by MPC in each vehicle are shared via V2V communication between vehicles.

In Li et al. (2010), Eben Li et al. (2013), Zhao et al. (2017), Han et al. (2013) multi-objective MPC is implemented in order to satisfy different objectives such as minimal fuel consumption, tracking capability and desired driver response; these objectives are contradictory. In other words, to have less fuel consumption, acceleration should be minimized; but minimizing the acceleration increases the gap between vehicles which in turn increases the tracking error. MPC is the control framework that can handle the contradiction between these objectives. This was implemented in Li et al. (2010), Eben Li et al. (2013), Zhao et al. (2017), Han et al. (2013) by minimizing a cost function with respect to certain constraints on inputs and states.

In this work, we extend the state-of-the-art multi-objective MPC approach for platooning in a distibuted manner by considering: (i) a realistic vehicle dynamics implying that a more descriptive platoon model is obtained; (ii) the multi-rate concept in the design of the upper-layer and the lower-layer controllers; (iii) realistic network behavior with packet loss.

In Zheng et al. (2016), Dunbar and Caveney (2011) the stability of DMPC for vehicle platooning was proved by choosing a terminal cost function and terminal constraint set. In Dunbar and Caveney (2011) sufficient conditions for guaranteeing stability are derived for a platoon of vehicles with decoupled nonlinear dynamics and varying platoon size. In Zheng et al. (2016), an equality-based terminal constraint is proposed to ensure stability for a dynamically decoupled system under unidirectional communication topology. For a time-varying system with coupled dynamics as in our approach, computing a cost function and terminal constraint set every time step requires huge computational resources. As we aim at low computational costs, we followed the proposed theorem in Limón et al. (2006) which states that stability of MPC can be obtained when considering a zero terminal constraint set.

Complying with the standards: Some platooning work uses custom wireless technologies such as WI-FI (IEEE 802.11a/g) (Ploeg et al., 2011; Naus et al., 2010) with a message rate of 10 Hz. Using WI-FI technology may cause interference with other signals especially in urban areas which weakens the wireless signal. Similarly, Ultra Wide Band (UWB) communication technology (IEEE 802.15.3a) was adopted in Wang et al. (2015). UWB provides a high speed UWB PHY enhancement amendment for applications involving streaming multimedia and massive data (Zhang et al., 2017). Other work, e.g. (Öncü et al., 2014; van Nunen et al., 2017; van de Sluis et al., 2015), uses the V2V communication standard IEEE 802.11p where a dedicated SCH is used with a special transceiver for sharing CAM messages at a 25 Hz message rate. In this work we adhere to the standards by using IEEE 802.11p with 10 Hz message rate over CCH.

Communication imperfections – delay/packet loss: Some work studied the effect of communication imperfections in terms of transmission delay and packet drops on platoon string stability, see for example (Öncü et al., 2014; Zhou et al., 2012; van Nunen et al., 2017; Ploeg et al., 2014; Gong et al., 2018). The authors in Ploeg et al. (2014) proposed a control strategy that switches from CACC to ACC in case of persistent packet loss. In case of increased communication delay, but not yet persistent packet loss, CACC switches to a degraded CACC where the preceding vehicle's acceleration is estimated using on-board sensors. In our work, we focus on the situation where packet losses are not persistent and can be handled in CACC. It can be complemented with an approach to switch to ACC when needed.

Communication delay is either constant or time-varying. Constant communication delay is considered in Öncü et al. (2014), Liu et al. (2001), Gao et al. (2016) whereas time-varying communication delay is considered in Qin et al. (2016), Fiengo et al. (2019), Di Bernardo et al. (2014). The authors in Liu et al. (2001) studied the effects of communication delay on string stability. A control law is designed that uses the information received from the preceding vehicle and the lead vehicle. In Gao et al. (2016), an H-infinity control method is proposed for heterogeneous platoons with uniform communication delay and uncertain dynamics. In Qin et al. (2016), the authors investigated the effects of stochastic communication delay on the stability of connected cruise control. In Di Bernardo et al. (2014), the authors considered the platooning problem as a problem of achieving consensus in a network of dynamical systems under time-varying heterogeneous communication delay. In this work, we do not consider communication delay in the control design. However, using the CReTS simulation framework, we measure the communication delay experienced between vehicles for the scenarios under consideration. It can be concluded that delay is small in the scenarios low or medium congestion, and that control performance does not suffer in such cases.

On the other hand, we evaluate our approach considering packet drops under realistic network behavior using the CReTS simulation framework and the Cohda embedded platform. Packet drops were assumed to be either constant for all sender-receiver pairs over time in Zhou et al. (2012) or artificially accomplished by holding the transmission for a short period of time in van Nunen et al. (2017). In Zhou et al. (2012) (we discussed (Öncü et al., 2014; van Nunen et al., 2017) earlier), DMPC is designed for the upper-layer to handle latency in harsh communication environments. The message loss rate is assumed to be constant for all sender-receiver pairs and constant over time which is not a realistic assumption.

Experimental studies and embedded implementation: One of the first platooning projects that was founded in 1986 is the *California Partners for Advanced Transit and Highways* (PATH) project (Swaroop, 1997) in the US. This was followed by SARTRE

(Bergenhem et al., 2010) and GCDC (van de Sluis et al., 2015) in Europe. The goals from SARTRE are to achieve comfort, ensure safety, decrease congestion, and increase energy savings, whereas the aim from PATH is to increase throughput per lane and energy savings. In GCDC, where multiple teams tested their CACC systems and evaluated them against other teams, the goal is to promote deployment of cooperative driving systems based on the integration of state-of-the-art sensors, V2V communication, and control (Bergenhem et al., 2012). Some recent work e.g. (Naus et al., 2010; Ploeg et al., 2011; Milanés et al., 2013) reports experimental studies based on built-in ACC. When packet drops occur, ACC functionality is active. The implementations in the mentioned projects and studies are performed in powerful prototypes which are unlikely to be available in regular cars. The concept of connected cruise control (CCC) which is introduced in Orosz (2016), Qin et al. (2016), is experimentally validated in Jin et al. (2018). In CCC, automated vehicles that are equipped with V2V devices respond to the motion of human-driven vehicles ahead that are equipped with V2V devices but not necessarily equipped with range sensors (e.g. radar). Typically, the CCC technology is suitable and targeted for lesser level of autonomy (e.g., level-3 autonomy) while we target for a higher level of autonomy (e.g., level-5 autonomy (SAE international, 2016)).

On the other hand, MPC has been implemented for embedded platforms for several applications (Bernardini, 2018; Zometa et al., 2012). A number of works propose using specialized hardware such as FPGAs or ASICs (Bleris et al., 2006; Wills et al., 2011). Applications using conventional embedded platforms are generally limited to controlling small systems with slow sampling rates (Zometa et al., 2012). In this work we show the implementation of the proposed approach on an embedded platform targeting the automotive domain developed by Cohda Wireless and NXP. The authors proved in Soroa et al. (2019) that embedded MPC for vehicle platooning using conventional hardware is feasible.

Modeling: Longitudinal vehicle dynamics is nonlinear in nature. Nonlinearity comes from the engine torque maps, tire forces, gear positions, aerodynamic drag force, and so on (Ulsoy et al., 2012; Rajamani, 2011). Some work considers nonlinear models in MPC design for platooning, e.g. (Zheng et al., 2016; Dunbar and Caveney, 2011). In reality, control methods might not be easily implemented using such complicated models which closely describe the dynamics of the vehicle. Often in literature, simplified linear models are obtained with reasonable assumptions (Li et al., 2010; Zheng et al., 2014); for example double integrator models are used in Maxim et al. (2016), Liu et al. (2017)). Choosing a realistic model can successfully approximate the dynamics of the vehicle and hence advanced control methods can be applied easily.

In this work, we adopt the vehicle model proposed in Ulsoy et al. (2012), Tsujii et al. (1990) where the vehicle model combines the linearized longitudinal dynamics of the vehicle with the dynamics of the engine system. The throttle actuator which adjusts the throttle angle is modeled as a DC-servo motor. Interested readers are referred to (Li et al., 2015) and references therein for more details about state-of-the-art vehicle models used for platoon applications.

3. Multi-layer control scheme

The role of the upper-layer is to receive information from the other vehicles and from the environment. Based on this information, the upper-layer control calculates the desired acceleration (a^i_{des}) of the i^{th} vehicle and provides it to the lower-layer controller (see Fig. 1). As mentioned earlier, we use PF topology, where the upper-layer controller of the i^{th} vehicle receives information from its preceding vehicle i.e., the $(i-1)^{th}$ vehicle. It computes a^i_{des}based on q^{i-1}, v^{i-1} and a^{i-1} i.e. position, velocity and acceleration of the $(i-1)^{th}$ vehicle received over the IEEE 802.11p wireless network and the current status of the i^{th} vehicle i.e. q^i, v^i, a^i. Thus, by exchanging information over IEEE 802.11p, each platoon member computes its desired acceleration (i.e. upper-layer) which is then provided to the lower-layer via the in-vehicle network to realize it. As already stated, the upper-layer controller runs at 10 Hz as per the

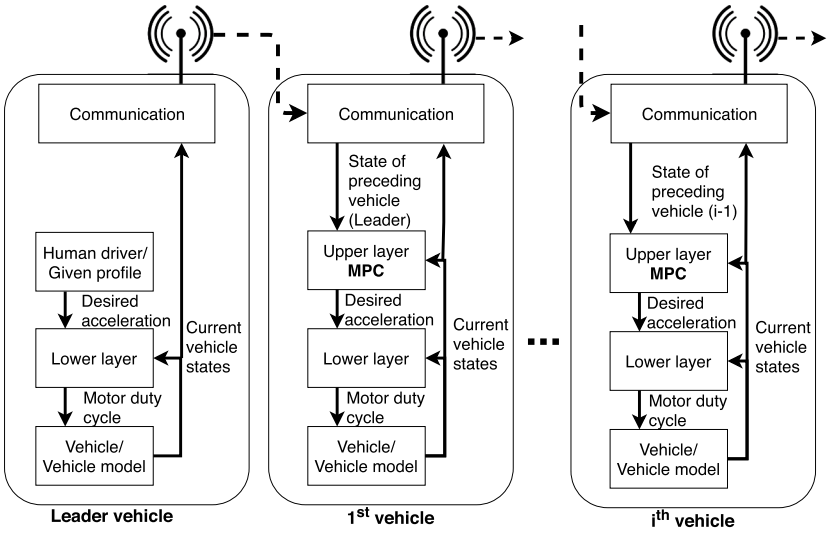


Fig. 1. Overall platoon control architecture.

ETSI CAM standard (ETSI, 2011).

The lower-layer controller is used as a tracking controller. Its objective is to reach the desired acceleration set by the upper-layer within a certain time i.e. before the new status of the preceding vehicle arrives over V2V wireless communication. In principle, the lower-layer controller computes the required motor duty cycle in % (see Fig. 2) which then changes the throttle plate angle so that the energy generated from the engine is enough to move the vehicle forward and reach the desired acceleration set by the upper-layer controller within a short interval. The lower-layer controller receives the desired acceleration over fast and reliable in-vehicle networks (e.g. FlexRay or Ethernet) (Makowitz and Temple, 2006). The lower-layer controller may be implemented with a short sampling period (e.g. 2 ms, 5 ms, 10 ms) considering typical automotive architecture (Goswami et al., 2013). Obviously, a shorter time to achieve a desired acceleration is desirable for a faster response.

4. Longitudinal vehicle modeling

The longitudinal vehicle motion system is composed of the following systems (Corneliu and Alexandru, 2018) (see Fig. 2):

i. **Throttle:** The throttle system is composed of throttle actuator (DC-servo motor), throttle plate, and return spring. The throttle plate angle is controlled using a DC servo motor by changing the motor duty cycle in % and thus, regulating the airflow in the intake manifold. The throttle plate angle θ_t ranges between 0 and 90 degrees. $\theta_t = 0$ denotes a closed throttle (the accelerator pedal is not pressed) and $\theta_t = 90$ denotes a completely opened throttle plate (the accelerator pedal is fully pressed). A mechanical stopper prevents the throttle to go beyond this range.
ii. **Engine:** The engine produces torque T_e which is related to the throttle plate angle θ_t.
iii. **Driveline:** The driveline delivers the generated torque to the driving wheels. Thus, the traction force F is generated through the contact between the driving wheels and the road surface.
iv. **Vehicle dynamics:** The effects of the rolling resistance, drag force, road slope, gravitational force and traction force on the vehicle body and its speed v.

As proposed in Ulsoy et al. (2012), Tsujii et al. (1990), the powertrain system (throttle, engine, driveline) can simply be modeled as a DC-servo motor (throttle actuator), since it is possible to regulate vehicle speed through throttle plate angle adjustments. In this section, we present the vehicle dynamics and the throttle actuator model that are used for the lower-layer control design and therefore used to obtain the model of a platoon consisting of N vehicles.

4.1. Vehicle dynamics

Using Newton's second law, the balance of the forces acting on the vehicle longitudinal axis is described as follows (Ulsoy et al., 2012),

$$m\frac{d\overline{v}}{dt} = \overline{F} - mg\sin\overline{\theta} - fmg\cos\overline{\theta} - 0.5\rho AC(\overline{v} + v_w)^2, \tag{1}$$

where $\overline{v}$ is the forward velocity, v_w is the wind velocity, $\overline{F}$ is the tractive force, ρ is the air density, C is the drag coefficient, A is the vehicle cross-sectional area, f is the rolling-resistance coefficient, m is vehicle mass, g is the gravitational acceleration, $\overline{\theta}$ is the road inclination angle. The above equation is nonlinear in the forward velocity $\overline{v}$, linearization can be done by applying first-order Taylor approximation around the equilibrium point $\{v_0,\ F_0,\ \theta_0\}$. At equilibrium (at $\frac{d\overline{v}}{dt} = 0$), Eq. 1 can be solved for:

$$F_0 = mg\sin\theta_0 + fmg\cos\theta_0 + 0.5\rho AC(v_0 + v_w)^2, \tag{2}$$

where F_0 can be found by assuming reasonable values for $v_0,\ \theta_0,\ m,\ f,\ A,\ C,\ v_w,\ \rho$. Then the linearized model can be written as follows,

$$\tau\dot{v} + v = K(F + \mathcal{E}), \tag{3}$$

where the perturbed variables are defined as, $v = \overline{v} - v_0,\ F = \overline{F} - F_0,\ \theta = \overline{\theta} - \theta_0$. The parameters $\tau,\ K$ and $\mathcal{E}$ are defined as follows,

$$\tau = (m/(\rho CA(v_0 + v_w))),\ K = (1/(\rho CA(v_0 + v_w))),\ \mathcal{E} = mg(f\sin\theta_0 - \cos\theta_0)\theta,$$

where $\mathcal{E}$ can be ignored for a flat road (no inclination).

By taking Laplace transforms of Eq. 3, the transfer function of the longitudinal vehicle model can be represented by,

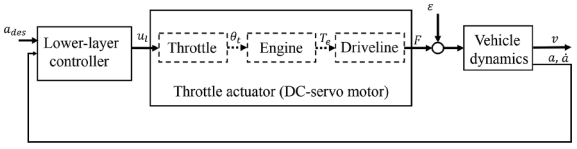


Fig. 2. Block diagram of longitudinal system.

$$G(s) = \frac{v}{F} = \frac{K}{\tau s + 1}. \tag{4}$$

Eq. 4 captures the relation between the forward velocity and the tractive force.

4.2. Throttle actuator

The input to the first-order vehicle model (Eq. 4) is the tractive force which is provided by the throttle actuator that is modeled as a DC-servo motor (see Fig. 2).

The DC-servo motor dynamics is given by the following second order transfer function (Tsujii et al., 1990),

$$G_a(s) = \frac{F}{u_l} = \frac{K_a}{s(\tau_a s + 1)}, \tag{5}$$

where u_l is the lower-layer control input (motor duty cycle in %). K_a and τ_a are motor parameters. The input to this motor model is the motor-drive duty cycle (in %) and the output is the throttle plate angle θ_t. The output of the throttle actuator (throttle plate angle) can be directly translated into tractive force F (Ulsoy et al., 2012). In other words, by opening the throttle to certain angle, the generated power and thus the tractive force responsible for generating motion is proportional to the throttle opening.

Inspired by Ulsoy et al. (2012), Tsujii et al. (1990) the following motor parameters are assumed: $\tau_a = 0.005$ s, $K_a = 100$ (s.Newton/%), $\tau = 100$ s and $K = 0.075$ (m/s)/Newton.

4.3. Vehicle's state-space model

The state-space model of the vehicle is obtained by combining the longitudinal vehicle motion model (Eq. 4) and actuator dynamics (Eq. 5). The overall transfer function of the vehicle model can be represented as,

$$G_{all}(s) = G_a(s)G(s) = \frac{v}{u_l} = \frac{KK_a}{s(\tau_a s + 1)(\tau s + 1)}. \tag{6}$$

Transforming Eq. 6 into the time domain yields the following third order linear differential equation:

$$\tau\tau_a \dddot{v} + (\tau + \tau_a)\ddot{v} + \dot{v} = KK_a u_l, \tag{7}$$

since $\dot{v} = a$, where a is the vehicle acceleration, Eq. 7 can be represented as:

$$\tau\tau_a \ddot{a} + (\tau + \tau_a)\dot{a} + a = KK_a u_l, \tag{8}$$

Eq. 8 represents the continuous-time model of longitudinal vehicle dynamics for a vehicle in a platoon. The notation in Eq. 8 can be generalized to represent any vehicle i in the platoon with $i = 1, \ldots, N$, where N is the number of vehicles in the platoon. The general representation of the i^{th} vehicle can be shown as follows,

$$\tau^i \tau_a^i \ddot{a}^i + \left(\tau^i + \tau_a^i\right)\dot{a}^i + a^i = K^i K_a^i u_l^i, \tag{9}$$

where $\tau^i,\ \tau_a^i,\ K^i,\ K_a^i$ are model parameters of the i^{th} vehicle. A heterogeneous platoon i.e. non-identical vehicles can be obtained by considering different vehicle model parameters. In this work, we assume homogeneous platoons i.e. all vehicles have the same model parameters.

Using Eq. 9, the continuous-time state-space model of the i^{th} vehicle can be represented as:

$$\dot{x}_l^i = A_l^i x_l^i + B_l^i u_l^i, \tag{10}$$

$$y_l^i = \mathcal{C}_l x_l^i.$$

The state vector x_l^i is defined as: $x_l^i = \begin{pmatrix} a^i \\ \dot{a}^i \end{pmatrix} \in \mathbb{R}^{2\times 1}$,where $a^i,\ \dot{a}^i$ denote the acceleration and the rate of change of acceleration of the i^{th} vehicle respectively. $\mathbb{R}$ is the set of real numbers. The state matrix A_l^i and the input vector B_l^i of the i^{th} vehicle are defined as,

$$A_l^i = \begin{pmatrix} 0 & 1 \\ \dfrac{-1}{\tau^i \tau_a^i} & \dfrac{-\left(\tau^i + \tau_a^i\right)}{\tau^i \tau_a^i} \end{pmatrix} \in \mathbb{R}^{2\times 2}, B_l^i = \begin{pmatrix} 0 \\ \dfrac{K^i K_a^i}{\tau^i \tau_a^i} \end{pmatrix} \in \mathbb{R}^{2\times 1}.$$

The output vector is defined as $\mathcal{C}_l = [1\ \ 0] \in \mathbb{R}^{1\times 2}$ and the output state y_l^i is the vehicle acceleration.

5. Lower-layer control design

5.1. Model discretization

The lower-layer controller which deals with the vehicle model will be implemented on a digital device (ECU). Thus a discrete version of the continuous-time vehicle model Eq. 10 is required. By discretizing Eq. 10 with sampling period h_l we obtain,

$$x_l^i(k+1) = \Phi_l^i x_l^i(k) + \Gamma_l^i u_l^i(k), \tag{11}$$

$$y_l^i(k) = C_l x_l^i(k)$$

where,

$$x_l^i(k) = \begin{pmatrix} a^i(k) \\ \delta a^i(k) \end{pmatrix} \in \mathbb{R}^{2\times 1}.$$

$x_l^i(k+1)$ denotes the discretized state vector at time $t = (k+1)h_l$. $a^i(k)$ and $\delta a^i(k)$ are the discretized acceleration and rate of change of acceleration at time step $t = kh_l$ of the i^{th} vehicle. $\Phi_l^i \in \mathbb{R}^{2\times 2}$ and $\Gamma_l^i \in \mathbb{R}^{2\times 1}$ represent the discretized system matrices with sampling period h_l. They are defined as follows:

$$\Phi_l^i = e^{A_l^i h_l},\ \Gamma_l^i = \int_0^{h_l} e^{A_l^i s} B_l^i ds.$$

The choice of h_l is driven by the sampling periods supported by common automotive operating systems such as OSEK (OSEK Consortium, 2005). For example, OSEK supports periods of 2, 3, 5, 10, 20 ms, etc. A shorter h_l requires a higher resource usage (in terms of communication and computation) of the in-vehicle electrical and electronic architecture.

5.2. Controller design

We consider a state-feedback control law in the lower-layer controller of the following form,

$$u_l^i(k) = \kappa^i x_l^i(k) + F^i a_{des}^i(k), \tag{12}$$

where $a_{des}^i(k)$, $\kappa^i \in \mathbb{R}^{1\times 2}$, $F^i \in \mathbb{R}^{1\times 1}$ are the desired acceleration, feedback gain and feedforward gain of the i^{th} vehicle, respectively. Substituting Eq. 12 into Eq. 11, we obtain the closed-loop system,

$$x_l^i(k+1) = (\Phi_l^i + \Gamma_l^i \kappa^i) x_l^i(k) + \Gamma_l^i F^i a_{des}^i(k). \tag{13}$$

Gains κ^i, F^i have to be designed such that the discrete-time closed-loop system Eq. 13 is stable and reaches the desired acceleration $a_{des}^i(k)$ within short time. It should be noted that stability is guaranteed by placing the eigen values of $(\Phi_l^i + \Gamma_l^i \kappa^i)$ inside the unit circle which can be obtained by proper design of κ^i.

Feedback gain κ^i is designed using *pole placement* (Kautsky et al., 1985) which places the poles of the discrete-time system Eq. 13 inside the unit circle. Other design methods can be chosen such as the LQR method. We avoided the unnecessary design complexity arising from tuning LQR weighting matrices.

There are two important criteria that affect the control performance and should be considered while choosing the pole locations and the sampling period h_l. These criteria are (i) *settling time* τ_s, (ii) *input saturation* U_{max}. Settling time τ_s is defined as the time needed by the control output to reach a close proximity (98%) of the reference ($a_{des}^i(k)$ in our case). The input saturation U_{max} is the maximum control input value that can be obtained (actuator saturation i.e. $|u_l^i(k)| \leqslant U_{max}$); in our example, $U_{max} = 100\%$ of motor duty cycle.

Choosing poles closer to zero causes the control action to be more aggressive. This results in a shorter settling time τ_s (i.e. the control output will reach the desired value more quickly), which at the same time might cause the control input $u_l^i(k)$ to exceed the actuator saturation limit U_{max}. The faster response achieved by minimizing settling time τ_s has direct effect on $\dot{a}^i$ (the first derivative of acceleration or rate of change of acceleration in m/s^3) which might exceed safety limits and cause injury to passengers. Therefore, poles should be chosen carefully in order to achieve short settling time and not to exceed actuator saturation or safety limits. Thus, while designing the lower-layer controller, we consider the following performance criteria:

1. -7 m/s$^3 \leqslant \dot{a}^i \leqslant 5$ m/s^3 (Powell and Palacín, 2015).
2. settling time $\tau_s \leqslant 100$ ms.
3. control input $u_l^i \leqslant 100\%$ of motor duty cycle.

Performance criterion 2 is our design choice based on which the upper-layer is designed (or, in other words, headway distance is chosen and MPC is tuned). This implies that a given acceleration reference is achieved before a new reference arrives. The requirement

is realistic for modern cars. Moreover, it is not a strict requirement for the upper-layer. The settling time in reality can be longer than 100 ms and the upper-layer performance can still be maintained by adjustment in the headway distance and MPC retuning. We present results for the mentioned choice of settling time. Results for other settling times are similar.

Choosing a short sampling period h_l improves the control performance but it also increases the processor load. In the next section we examine the lower-layer control performance under different pole locations and sampling periods of 2, 5, and 10 ms.

5.3. State estimation and observer design

For the lower-layer control in Eq. 12, the states should be available every time step h_l. The acceleration $a^i(k)$ can be measured using an IMU (inertial measurement unit) sensor and the rate of change of acceleration (jerk) δa^i can be computed by differentiating the acceleration as:

$$\delta a^i = \frac{a^i(k) - a^i(k-1)}{h_l}.$$

However, this method might be challenging because of measurement noise. Thus, we design an observer to estimate the state δa^i. The observer dynamics is defined as,

$$\widehat{x}_l^i(k+1) = \Phi_l^i \widehat{x}_l^i(k) + \Gamma_l^i u_l^i(k) + \mathcal{L}_{obs}\left(y_l{}^i(k) - \widehat{y}_l{}^i(k)\right), \tag{14}$$

where $\widehat{x}_l^i(k)$, $\widehat{y}_l{}^i(k)$ are the estimate of the state and output, respectively. $\mathcal{L}_{obs}$ is the observer gain which can be obtained in different ways e.g. using Kalman filter methodology (Gelb, 1974). Then the control input defined previously in Eq. 12 can be computed based on the estimated state as follows:

$$u_l^i(k) = \kappa^i \widehat{x}_l^i(k) + F^i a_{des}^i(k). \tag{15}$$

5.4. Design example of lower-layer controller

To simulate the behavior of the lower-layer control, we assume that messages from the preceding vehicle are received with 10 Hz frequency i.e. every 100 ms. Also, the upper-layer controller is assumed to be in place, i.e. it computes and delivers the desired acceleration to the lower-layer. We test the lower-layer controller using step change in acceleration every 100 ms. To consider a realistic driving scenario, the driver is assumed to increase the acceleration with rate 0.02 m/s^2 every 100 ms. In other words, if the driver keeps pressing the pedal with the same rate, the acceleration will linearly increase from 0 to 3.3 m/s^2 in 16 s and the velocity will increase from 0 to 100 km/h. In our case-study, we simulate only three step changes for 300 ms. As seen in Fig. 3, acceleration increases from 0 m/s^2 to 0.02 m/s^2 then it increases again to 0.04 m/s^2 and finally it drops to 0 m/s^2. As shown in Table 1 and Fig. 3, different sets of poles with $h_l \in \{2, 5, 10\}$ ms can be chosen to satisfy the lower-layer performance criteria.

6. Platoon modeling under PF topology

6.1. Network topology

V2V communication and radar-based communication between vehicles in a platoon introduce different network topologies that describe the information flow between vehicles. An example is the Predecessor-Following (PF) topology where a follower vehicle receives information from its direct predecessor only. In Predecessor-Leader-Following (PLF) topology, a follower vehicle receives information from its direct predecessor and from the platoon leader vehicle. Whereas in Two-Predecessors-Follower (TPF), the vehicle receives information from its two direct predecessors (Zheng et al., 2014). PLF and TPF type of topologies are interesting in applications such as trajectory planning where the leader computes and shares the trajectory with the other vehicles. In this work, PF network topology is considered since the state of the directly preceding vehicle is crucial for safe platoon operation. For example, in case of emergency braking, the immediately following vehicle (which receives this information over wireless communication under PF

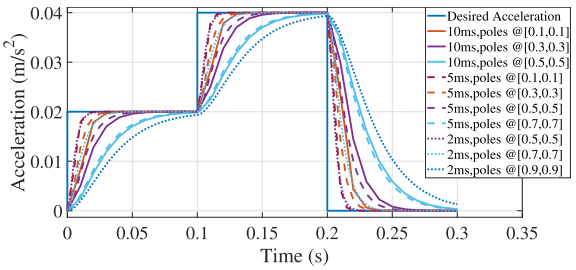


Fig. 3. Acceleration profile at 2 ms, 5 ms and 10 ms lower-layer sampling period for different pole locations.

Table 1
Lower-layer control performance using different pole locations and different sampling period for in-vehicle network.

	Pole location	u_l^i – motor duty cycle in %	τ_s (ms)	Max a^i (m/s^3)	Min a^i (m/s^3)
10 ms-sampling	[0.5,0.5]	15.3531	100	0.5053	−0.9969
	[0.3,0.3]	30.2193	80	0.9799	−1.9595
	[0.1,0.1]	49.9587	40	1.6197	−3.2393
5 ms-sampling	[0.7,0.7]	15.6240	100	0.5324	−1.0551
	[0.5,0.5]	42.1816	60	1.0000	−1.9999
	[0.3,0.3]	82.6814	35	1.9599	−3.9198
	[0.1,0.1]	136.6809	25	3.2398	−6.4797
2 ms-sampling	[0.9,0.9]	10.7057	100	0.4005	−0.7627
	[0.7,0.7]	72.7933	44	1.3230	−2.6460
	[0.5,0.5]	202.2130	22	2.5000	−5.0000

topology) has to react first to avoid a crash. While we demonstrated our approach considering PF topology, it can be further extended to other topologies.

6.2. V2V communication

IEEE 802.11p is an amendment to IEEE802.11a where the physical layer properties are modified to cope with the rapidly changing vehicle position. IEEE802.11p provides more robustness against fading and increases the tolerance for multipath propagation effects of signals in a vehicular environment (Lin et al., 2010). As defined by the European Telecommunications Standards Institute (ETSI) (ETSI, 2011; ETSI, 2014a), V2V wireless communication for platooning under the IEEE 802.11p standard (Jiang and Delgrossi, 2008) uses the CCH (control channel) to share information between vehicles. Two types of messages share the control channel, the periodic Cooperative Awareness Message (CAM) and the event-triggered Decentralized Environmental Notification Message (DENM) (ETSI, 2011). DENMs are used to issue warnings in emergency situations (ETSI, 2013). CAMs are sent periodically to create and maintain awareness between vehicles and roadside units and therefore used for platoon applications. CAMs follow a transmission or message rate of 10 Hz (i.e. with sampling period 100 ms) when the channel load is less than 70%; under a channel load greater than 70%, message rate of CAMs can be lowered to 1 Hz controlled by a Decentralized Congestion Control (DCC) algorithm (ETSI, 2014b). Data types contained in all generated CAMs by a vehicle shall include all fast-changing (dynamic) measured status information such as heading, speed, position and acceleration. Optionally, CAMs might include slow-changing or static status such as the status of the exterior lights. The accuracy of the dynamic vehicle status should also be sent along with the actual vehicle status for safe cooperative ITS (Intelligent Transport Systems); vehicle's status might be indicated as 'unavailable' in case of error in measuring for example.

The number of vehicles that causes 70% channel load can be estimated as follows. The 100% channel load can be calculated assuming that one vehicle can start sending a message immediately after the transmission of the previous message is completed, which is an ideal case excluding MAC protocol delays. If we consider a 300 Byte message length of 0.4 ms, the total number of messages per second is $1000/0.4 = 2500$. At 10 Hz message rate per vehicle, this can allow 250 vehicles to transmit without conflict at 100% load. The 250 vehicles is calculated based on a 10 Hz message rate of 300 Byte messages in a 6 Mbps channel. One 300 Byte message takes around 0.4 ms in a 6 Mbps channel, which is used in the IEEE 802.11p V2X (vehicle-to-everything) communication. At 70% load, the number of vehicles is $250 \times 0.7 = 175$. We can take a 1 km access zone to estimate the vehicle-vehicle distance, which means a vehicle can listen to others within 500 meter distance. If we consider an 8-lane highway and 1 km communication zone, the 250 vehicles can be distributed over 8 lanes within 500 m around the listening vehicle to derive the average inter-vehicle distance: $1000/(250/8) = 32$ m. 32 m vehicle-vehicle distance is a quite normal situation in busy times. With such an estimation, 70% channel load means 175 vehicles using 10 Hz message rate. The average vehicle-vehicle distance on an 8-lane highway (using 1 km communication zone) is $1000/(175/8) = 45.6$ m. In other words, if the vehicle density is as high as one per 45 m on an 8-lane highway, the channel is 70% loaded. Note that the above is an ideal estimation without considering message collision due to the MAC protocol, which means the measured channel load by a vehicle is usually lower due to collisions and the received signal power loss over the channel from the sender. In our simulation, we take 200 vehicles in a 1 km zone to simulate 70% measured channel load case.

6.3. Platoon modelling

Overall platoon behavior depends on the inter-vehicle longitudinal dynamics which is defined by introducing two state variables, position error Δd^i and velocity error Δv^i. They are defined as follows,

$$\Delta d^i = d^i - d^i_{des}, \tag{16}$$

$$\Delta v^i = v^{i-1} - v^i, \tag{17}$$

where Δd^i is defined as the error between the actual gap d^i and the desired inter-vehicle gap d^i_{des} between the i^{th} vehicle and the $(i-1)^{th}$ vehicle. Similarly, Δv^i is the velocity error between the i^{th} and the $(i-1)^{th}$ vehicles. v^i is the velocity of the i^{th} vehicle. d^i and d^i_{des} are

defined as follows (see Fig. 4),

$$d^i_{des} = d_0 + \tau_h v^i, \quad d^i = q^{i-1} - q^i - L^i, \tag{18}$$

where d_0 is the gap between vehicles at standstill, τ_h is the constant headway time (the time the i^{th} vehicle needs to reach the position of the $(i-1)^{th}$ vehicle when $d_0 = 0$). L^i, q^i are the length and position of the i^{th} vehicle, respectively. In our case study, we consider $d_0 = 1$ m and $L^i = 4$ m.

To represent the inter-vehicle dynamics in state-space form, we start by differentiating Eq. 16 and Eq. 17. The first order derivative of Δd^i and Δv^i are represented as follows,

$$\Delta \dot{d}^i = \Delta v^i - \tau_h a^i, \tag{19}$$

$$\Delta \dot{v}^i = a^{i-1} - a^i, \tag{20}$$

where a^{i-1} is the acceleration of the $(i-1)^{th}$ vehicle received over V2V wireless communication.

The state-space representation of the inter-vehicle longitudinal dynamics is obtained from Eq. 19 and Eq. 20 and can be written in the following form,

$$\dot{x}^i_m = A^i_m x^i_m + H^i_m x^i_l + G^i_m a^{i-1}, \tag{21}$$

where,

$$x^i_m = \begin{bmatrix} \Delta d^i \\ \Delta v^i \end{bmatrix} \in \mathbb{R}^{2\times 1},\ A^i_m = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{2\times 2},\ H^i_m = \begin{bmatrix} -\tau_h & 0 \\ -1 & 0 \end{bmatrix} \in \mathbb{R}^{2\times 2},\ G^i_m = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \in \mathbb{R}^{2\times 1}.$$

Combining the vehicle model Eq. 10 with the inter-vehicle longitudinal dynamics Eq. 21, we obtain the platoon model under PF topology,

$$\dot{x}^i_u = A^i_u x^i_u + B^i_u \underbrace{\begin{bmatrix} u^i_l \\ a^{i-1} \end{bmatrix}}_{\in \mathbb{R}^{2\times 1}}, \tag{22}$$

where $A^i_u = \begin{bmatrix} A^i_l & \mathbf{0} \\ H^i_m & A^i_m \end{bmatrix} \in \mathbb{R}^{4\times 4}$, $x^i_u = \begin{bmatrix} x^i_l \\ x^i_m \end{bmatrix} = \begin{bmatrix} q^i \\ a^i \\ \Delta d^i \\ \Delta v^i \end{bmatrix} \in \mathbb{R}^{4\times 1}$, $B^i_u = \begin{bmatrix} B^i_l & \mathbf{0} \\ \mathbf{0} & G^i_m \end{bmatrix} \in \mathbb{R}^{4\times 2}$.

7. Multi-rate control scheme

The separation between upper and lower-layers allows for the multi-rate control scheme which implies that both layers may run with different sampling periods. The lower-layer which is responsible for achieving the longitudinal motion of the vehicle and attaining the desired acceleration within a specified time period must run much faster than the upper-layer which gathers information of the other vehicles and finds the desired acceleration of the current vehicle. This is inherently possible since the lower-layer is implemented on an in-vehicle architecture without any dependency on the slower and unreliable wireless communication and allows for a faster execution and a shorter sampling period. The lower-layer controller is scheduled with a 2 ms sampling period in our case study. On the other hand, the upper-layer communicates over the IEEE 802.11p and hence, the sampling period is constrained by 10 Hz

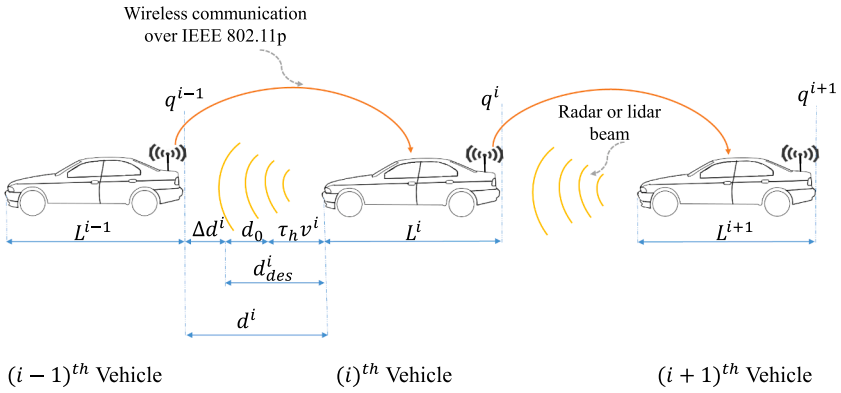


Fig. 4. Consecutive vehicles in a platoon.

communication frequency of the beacon messages (assuming no DCC is implemented). The sampling rate of the upper-layer controller is therefore 100 ms. Fig. 1 gives an overview of multi-rate concept of the layered control design.

A discrete version of the continuous-time platoon model is required in order to be able to design and implement the upper-layer controller on a digital ECU.

The upper-layer controller computes the desired acceleration considering the platoon model in Eq. 22. Overall system behavior depends on the relation between the lower-layer sampling period h_l and the upper-layer sampling period h_u. Generally, $h_u \gg h_l$ and h_u is an integer multiple of h_l, i.e.,

$$\eta = \frac{h_u}{h_l}, \quad \eta \in \mathbb{N}, \tag{23}$$

where $\mathbb{N}$ represents the set of natural numbers. We illustrate the design considering $h_u = 100$ ms and $h_l = \{2, 10, 20\}$ ms. To this end, we first discretize Eq. 22 at sampling period h_l as follows,

$$x_u^i(k+1) = \Phi_u^i x_u^i(k) + \Gamma_u^i \begin{bmatrix} u_l^i(k) \\ a^{i-1}(k) \end{bmatrix}, \tag{24}$$

where Φ_u^i and Γ_u^i are the discretized version of A_u^i and B_u^i, respectively. They are defined as follows,

$$\Phi_u^i = e^{A_u^i h_l} \in \mathbb{R}^{4\times4}, \; \Gamma_u^i = \int_0^{h_l} e^{A_u^i s} B_u^i ds \in \mathbb{R}^{4\times2},$$

and, $x_u^i(k) = \begin{bmatrix} a^i(k) \\ \delta a^i(k) \\ \Delta d^i(k) \\ \Delta v^i(k) \end{bmatrix} \in \mathbb{R}^{4\times1}$. Since h_l is used as the sampling period, $u_l^i(k)$ as per Eq. 12 can be used to simplify Eq. 24. By substituting Eq. 12 into Eq. 24, we obtain the following,

$$x_u^i(k+1) = \Phi_u^i x_u^i(k) + \Gamma_u^i \begin{bmatrix} \kappa^i x_l^i(k) + F^i a_{des}^i(k) \\ a^{i-1}(k) \end{bmatrix}. \tag{25}$$

By knowing that $x_u^i = \begin{bmatrix} x_l^i \\ x_m^i \end{bmatrix}$, it is possible to obtain a simplified version of Eq. 25 by collecting the similar terms together. To do this, we rewrite Φ_u^i, Γ_u^i and κ^i in the following forms,

$$\Phi_u^i = \begin{bmatrix} \Phi_{11} & \Phi_{12} & \Phi_{13} & \Phi_{14} \\ \Phi_{21} & \Phi_{22} & \Phi_{23} & \Phi_{24} \\ \Phi_{31} & \Phi_{32} & \Phi_{33} & \Phi_{34} \\ \Phi_{41} & \Phi_{42} & \Phi_{43} & \Phi_{44} \end{bmatrix}, \; \Gamma_u^i = \begin{bmatrix} \Gamma_{11} & \Gamma_{12} \\ \Gamma_{21} & \Gamma_{22} \\ \Gamma_{31} & \Gamma_{32} \\ \Gamma_{41} & \Gamma_{42} \end{bmatrix}, \; \kappa^i = \begin{bmatrix} \kappa_{11} & \kappa_{12} \end{bmatrix}.$$

Then Eq. 25 can be reformulated as,

$$x_u^i(k+1) = \begin{bmatrix} \Phi_{11} & \Phi_{12} \\ \Phi_{21} & \Phi_{22} \\ \Phi_{31} & \Phi_{32} \\ \Phi_{41} & \Phi_{42} \end{bmatrix} x_l^i(k) + \begin{bmatrix} \Phi_{13} & \Phi_{14} \\ \Phi_{23} & \Phi_{24} \\ \Phi_{33} & \Phi_{34} \\ \Phi_{43} & \Phi_{44} \end{bmatrix} x_m^i(k) + \underbrace{\begin{bmatrix} \Gamma_{11} \\ \Gamma_{21} \\ \Gamma_{31} \\ \Gamma_{41} \end{bmatrix}}_{\Xi_u^i \in \mathbb{R}^{4\times1}} \left(\begin{bmatrix} \kappa_{11} & \kappa_{12} \end{bmatrix} x_l^i(k) + F^i a_{des}^i(k) \right) + \underbrace{\begin{bmatrix} \Gamma_{12} \\ \Gamma_{22} \\ \Gamma_{32} \\ \Gamma_{42} \end{bmatrix}}_{\Psi_u^i \in \mathbb{R}^{4\times1}} a^{i-1}(k) \tag{26}$$

By collecting similar terms together, we obtain the following simplified discrete platoon model,

$$x_u^i(k+1) = \Pi_u^i x_u^i(k) + \Xi_u^i F^i u^i(k) + \Psi_u^i a^{i-1}(k) \tag{27}$$

where,

$$\Pi_u^i = \begin{bmatrix} \Phi_{11} + \Gamma_{11}\kappa_{11} & \Phi_{12} + \Gamma_{11}\kappa_{12} & \Phi_{13} & \Phi_{14} \\ \Phi_{21} + \Gamma_{21}\kappa_{11} & \Phi_{22} + \Gamma_{21}\kappa_{12} & \Phi_{23} & \Phi_{24} \\ \Phi_{31} + \Gamma_{31}\kappa_{11} & \Phi_{32} + \Gamma_{31}\kappa_{12} & \Phi_{33} & \Phi_{34} \\ \Phi_{41} + \Gamma_{41}\kappa_{11} & \Phi_{42} + \Gamma_{41}\kappa_{12} & \Phi_{43} & \Phi_{44} \end{bmatrix} \in \mathbb{R}^{4\times4}, \; u^i(k) = a_{des}^i(k).$$

The platoon model Eq. 27 is discretized with the lower-layer sampling period h_l and it does not yet capture the upper-layer dynamics. In other words, since $h_u = \eta h_l$ (Eq. 23), the lower-layer control loop executes η times within one upper-layer sampling period h_u. We unroll the loop (i.e. Eq. 27) η times to obtain the upper-layer dynamics. In other words, $x_u^i(k+\eta)$ is found by recursively solving $x_u^i(k+j)$ for $j = 1, \ldots, \eta$. For example, $x_u^i(k+2)$ is found as,

$$x_u^i(k+2) = \Pi_u^i x_u^i(k+1) + \Xi_u^i F^i u^i(k+1) + \Psi_u^i a^{i-1}(k+1)$$

By substituting the definition of $x_u^i(k+1)$ as per Eq. 27 in the above equation, we obtain the following:

$$x_u^i(k+2) = \left(\Pi_u^i\right)^2 x_u^i(k) + \left(\Pi_u^i + \mathcal{I}\right)\Xi_u^i F^i u^i(k) + \left(\Pi_u^i + \mathcal{I}\right)\Psi_u^i a^{i-1}(k).$$

where $\mathcal{I} \in \mathbb{R}^{4\times4}$ is the identity matrix (the elements of the principal diagonal are ones and all other elements are zeros).

Remark 1. It should be noted that in the previous formula, $u^i(k+1) = u^i(k)$ and $a^{i-1}(k+1) = a^{i-1}(k)$. This is correct since the new acceleration value of the preceding vehicle i.e. $a^{i-1}(k+1)$ is received only at time $t = (k+\eta)h_l$ (every 100 ms in our case study) and then the new control input $u^i(k+1)$ is computed (once every 100 ms). Therefore, at step $k+j$ or at time $t = (k+j)h_l, j = 1, \ldots, \eta-1$ then $u^i(k+j) = u^i(k)$ and $a^{i-1}(k+j) = a^{i-1}(k)$. See Fig. 5 for more details.

Similarly, $x_u^i(k+3)$ is obtained as follows,

$$x_u^i(k+3) = \left(\Pi_u^i\right)^3 x_u^i(k) + \left(\left(\Pi_u^i\right)^2 + \Pi_u^i + \mathcal{I}\right)\Xi_u^i F^i u^i(k) + \left(\left(\Pi_u^i\right)^2 + \Pi_u^i + \mathcal{I}\right)\Psi_u^i a^{i-1}(k).$$

By following the same procedure, $x_u^i(k+\eta)$, is represented in terms of $x_u^i(k),\ a^{i-1}(k),\ u^i(k)$ as follows,

$$\begin{aligned} x_u^i(k+\eta) = \left(\Pi_u^i\right)^\eta x_u^i(k) &+ \left(\left(\Pi_u^i\right)^{\eta-1} + \left(\Pi_u^i\right)^{\eta-2} + \ldots + \Pi_u^i + \mathcal{I}\right)\Xi_u^i F^i u^i(k) + \\ &\left(\left(\Pi_u^i\right)^{\eta-1} + \left(\Pi_u^i\right)^{\eta-2} + \ldots + \Pi_u^i + \mathcal{I}\right)\Psi_u^i a^{i-1}(k). \end{aligned}$$

Therefore, the platoon model that captures the lower-layer dynamics and the upper-layer sampling period h_u can be represented as,

$$x^i(k+1) = \alpha^i x^i(k) + \beta^i u^i(k) + \gamma^i a^{i-1}(k), \tag{28}$$

where,

$$x^i(k+1) := x_u^i(k+\eta), \tag{29}$$

$$x^i(k) := x_u^i(k) = \begin{bmatrix} a^i(k) \\ \delta a^i(k) \\ \Delta d^i(k) \\ \Delta v^i(k) \end{bmatrix} \in \mathbb{R}^{4\times1}, \tag{30}$$

$$\begin{aligned} \alpha^i &:= \left(\Pi_u^i\right)^\eta, \\ \beta^i &:= \left(\left(\Pi_u^i\right)^{\eta-1} + \left(\Pi_u^i\right)^{\eta-2} + \ldots + \Pi_u^i + \mathcal{I}\right)\Xi_u^i F^i, \\ \gamma^i &:= \left(\left(\Pi_u^i\right)^{\eta-1} + \left(\Pi_u^i\right)^{\eta-2} + \ldots + \Pi_u^i + \mathcal{I}\right)\Psi_u^i. \end{aligned}$$

Fig. 5 explains the relation between $x^i$and x_u^i that is defined in Eq. 29 and Eq. 30. Initially $k \in \mathbb{N}$ and $x^i(k)$ and $x_u^i(k)$ are equal. At $k+1$ (or when time evolves for one step), $x^i(k+1)$ represents the new state of the platoon vehicle obtained at time $t = (k+1)h_u$ (after 100 ms in our case study). $x_u^i(k+1)$ is the state of the platoon vehicle obtained at time $t = (k+1)h_l$ i.e. after 2 ms, 10 ms or 20 ms (based on the chosen h_l). Therefore, by unrolling the loop Eq. 27 η times then we obtain $x^i(k+1) := x_u^i(k+\eta)$.

8. Upper-layer controller design using MPC

There are certain objectives that should be satisfied by each vehicle in a platoon. The important objectives are: following the speed

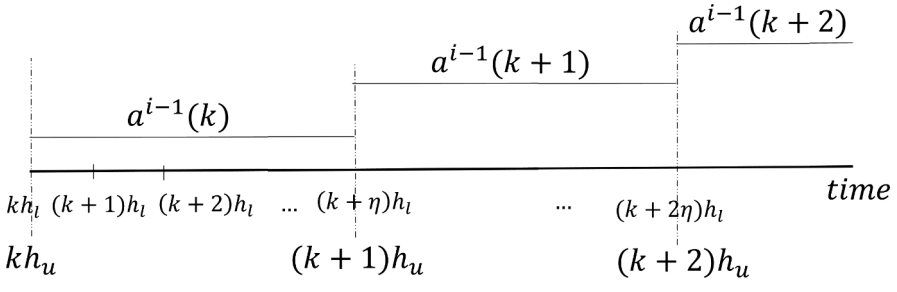


Fig. 5. Relation between h_l and h_u.

of the predecessor, avoiding cut-ins from adjacent lanes and rear-end collisions. Moreover, acceleration and speed should not exceed upper and lower bounds. In addition, minimizing the spacing error between vehicles and sudden changes in acceleration (for comfortable driving) are to be considered. Model Predictive Control (MPC) (Maciejowski, 2002) is the control framework that can handle these objectives and constraints. MPC is widely spread in industry for its ability to handle highly complex multi-variable processes with constraints on inputs, internal states and outputs.

MPC solves an optimization problem on-line every time step based on a model of the system and constraints on input and states. At each time step k, the current state of the system is fed back to the MPC controller and by using the prediction model, the future evolution of this system is calculated from time step k to time step $k + N_p$, where N_p is the prediction horizon length. Next, a predefined cost function is minimized taking into account the constraints defined on inputs and states. The output of this optimization problem is a sequence of control inputs $u_{k|k}, \ldots, u_{N_p-1+k|k}$ where only the first input $u_{k|k}$ is applied to the system and the whole process is repeated in the next time step $k + 1$.

8.1. MPC objectives and constraints for a platoon vehicle

In our MPC formulation, we consider tracking capability and fuel economy as the optimization objectives; more specifically we consider the following objectives:

1. minimizing the gap between vehicles while maintaining a safe inter-vehicle distance.
2. tracking the speed and acceleration profiles of the preceding vehicle.
3. avoiding rear-end collision with the preceding vehicle if it decelerates.
4. avoiding cut-ins from adjacent lanes if the preceding vehicle accelerates.
5. minimizing sudden changes in acceleration to maintain passenger comfort and to reduce fuel consumption.To achieve objectives 1, 2 and 5, we consider the following cost function that has to be minimized,

$$J = w_{\Delta v}\left(\Delta v^i\right)^2 + w_{\Delta d}\left(\Delta d^i\right)^2 + w_u\left(u^i\right)^2 + w_{\delta_a}\left(\delta a^i\right)^2 + w_a\left(a^i\right)^2, \tag{31}$$

where $w_{\Delta v}, w_{\Delta d}, w_u, w_{\delta_a}, w_a$ are weighting parameters that penalize $\Delta v^i, \Delta d^i, u^i$ and δa^i and a^i, respectively. Choosing suitable values for the weighting parameters ensures that $\Delta v^i, \Delta d^i, u^i$ and δa^i and a^i converge to zero. Tracking capability is achieved by minimizing position error Δd^i and velocity error Δv^i in this cost function (objectives 1 and 2). Since fuel economy is directly affected by acceleration and its rate of change (Zhang and Ioannou, 2006), minimizing $w_u(u^i)^2 + w_{\delta_a}(\delta a^i)^2$ in the cost function ensures better fuel economy. It should be noted that the desired acceleration a^i_{des} (the control input u^i that is to be computed using MPC and delivered to the lower-layer) and vehicle acceleration a^i (the lower-layer control output) have a similar effect on improving fuel efficiency; thus, to avoid redundancy w_a can be set to zero. Moreover, by minimizing the rate of change of acceleration δa, driving comfort can be achieved.

Another factor that achieves better fuel efficiency is decreasing the headway time τ_h (Eq. 18) i.e. reducing the gap between vehicles (see Eq. 19). Decreasing the gap between vehicles reduces the aerodynamic drag force experienced between vehicles which has a direct effect on decreasing the fuel consumption (Deng, 2016).

For driving comfort and fuel economy, acceleration ($a^i(k)$), desired acceleration ($a^i_{des}(k)$) and rate of change of acceleration ($\delta a^i(k)$) are bounded within certain limits. Therefore, constraints defined in Eq. 32, Eq. 33 and Eq. 34 are added.

$$a_{min} \leqslant a^i \leqslant a_{max}, \tag{32}$$

$$a_{min} \leqslant a^i_{des} \leqslant a_{max}, \tag{33}$$

$$\delta a_{min} \leqslant \delta a^i \leqslant \delta a_{max}. \tag{34}$$

To avoid cut-ins from adjacent lanes in case that the preceding vehicle moves with constant speed, then minimizing the cost function ensures that Δd^i and Δv^i converge to small values. This means that no vehicles will occupy the small gap between platoon vehicles. Otherwise, if the preceding vehicle accelerates or decelerates, inequalities defined in Eq. 35 are required to guarantee that tracking errors do not exceed specific bounds. Thus large inter-vehicle gaps are avoided to prevent cut-ins from adjacent lanes.

$$\Delta d_{min} \leqslant \Delta d^i \leqslant \Delta d_{max}, \ \Delta v_{min} \leqslant \Delta v^i \leqslant \Delta v_{max}. \tag{35}$$

For avoiding rear-end collision when the preceding vehicle starts to decelerate, the constraint defined in Eq. 36 is added. Here the actual gap between vehicles d^i is bounded from above by d_{max} and from below by d_0.

$$d_0 \leqslant d^i \leqslant d_{max}. \tag{36}$$

It should be noted that to handle sudden braking in emergency situations, more conservative inequalities should be considered, see e.g. (Turri et al., 2016). Moreover, in case of emergency braking, event triggered DENM (ETSI, 2011) messages are sent to other vehicles. Vehicles transmit an “emergency electronic brake lights” DENM message (over the shared control channel) if the emergency brake lights are set “on” and the deceleration value is equal to or higher than a predefined “emergency braking deceleration” value (ETSI,

2013). We limit the scope of our paper to the CAM messages. That is, our proposed method with CAM messages is not meant for emergency scenarios which require more conservative constraints and additional feedback using, for example, DENM messages.

8.2. MPC numerical computation

In order to solve the optimization problem and to find the global minimum, we need to formulate our problem in the standard (convex) quadratic programming (QP) form,

$$\min_{U_k^i} \quad \frac{1}{2}\left(U_k^i\right)^T G^i U_k^i + \left(U_k^i\right)^T \mathcal{F}^i \tag{37}$$

$$s.t. \quad LU_k^i \leqslant C + wx^i(k) \tag{38}$$

Therefore different QP solvers can be used to find the optimal solution. It should be noted that, to find a global minimum, the problem should be convex. In other words, if the problem is in the above QP form, if there are no constraints and if $G^i \succ 0$, then the cost function is clearly convex (otherwise there might be infinitely many attainable negative values and therefore no minimum). If there are constraints in the form of linear inequalities, the surfaces on which they are active are hyperplanes. Therefore, the constrained cost function can be shown to be a convex surface, parts of which have been cut off by flat surfaces. So a global minimum is then still attainable (Maciejowski, 2002).

To write the optimization problem in QP form, we should write the prediction model, the cost function and constraints in a more compact form. In the following we provide more details on how to formulate the MPC problem in QP form.

Prediction model: A prediction model is required in order to predict the future states of the system over the future horizon of length N_p. By minimizing the cost function and by knowing the predicted future evolution of the system, a sequence of optimal control inputs has to be calculated that satisfy the objectives and constraints over the horizon N_p. From the platoon model (Eq. 28), the prediction model for the i^{th} vehicle can be written as,

$$x_{k+j+1|k}^i = \alpha^i x_{k+j|k}^i + \beta^i u_{k+j|k}^i + \gamma^i a_{k+j|k}^{i-1}, \quad j = 0, \ldots, N_p - 1. \tag{39}$$

The notation $x_{k+j+1|k}^i$ represents the predicted state at step $k+j+1$ when the prediction is done at step k for the i^{th} vehicle. For $j = 0$,

$$x_{k|k}^i = x^i(k), \quad \text{where} \quad x^i(k) = \begin{bmatrix} a^i(k) \\ \delta a^i(k) \\ \Delta d^i(k) \\ \Delta v^i(k) \end{bmatrix},$$

i.e. the predicted state at step k when the prediction is done at step k equals the current (measured) state of the i^{th} vehicle. Similarly, $u_{k+j|k}^i$ for $j = 0, \ldots, N_p - 1$ represents the sequence of the calculated (optimal) control inputs over the prediction horizon N_p.

Since the future evolution of the preceding vehicle is not known in advance, we assume that the predicted acceleration values $a_{k+j|k}^{i-1}$, $j = 0, \ldots, N_p - 1$, of the $(i-1)^{th}$ vehicle are constants and equal $a^{i-1}(k)$, i.e. the actual acceleration of the $(i-1)^{th}$ vehicle received at step k (every 100 ms).

To write the prediction model Eq. 39 in a more compact form, we unroll the loop over the prediction horizon N_p:

$$\begin{aligned} x_{k+1|k}^i &= \alpha^i x_{k|k}^i + \beta^i u_{k|k}^i + \gamma^i a^{i-1}(k), \\ x_{k+2|k}^i &= \alpha^i x_{k+1|k}^i + \beta^i u_{k+1|k}^i + \gamma^i a^{i-1}(k), \\ &\vdots \\ x_{k+N_p|k}^i &= \alpha^i x_{k+N_p-1|k}^i + \beta^i u_{k+N_p-1|k}^i + \gamma^i a^{i-1}(k). \end{aligned}$$

In previous equations, by recursively substituting $x_{k+j-1|k}^i$ into $x_{k+j|k}^i$, we can rewrite the predicted state $x_{k+j|k}^i$ as a function of the current (measured) state (since $x_{k|k}^i = x^i(k)$) and the predicted control inputs $u_{k|k}^i, \ldots, u_{k+N_p-1|k}^i$. Grouping the above equations into matrix formulation, we obtain the following,

$$\begin{bmatrix} x^i_{k+1|k} \\ x^i_{k+2|k} \\ \vdots \\ x^i_{k+N_p|k} \end{bmatrix} = \begin{bmatrix} \alpha^i \\ (\alpha^i)^2 \\ \vdots \\ (\alpha^i)^{N_p-1} \end{bmatrix} x^i(k) + \begin{bmatrix} \beta^i & 0 & \dots & 0 \\ \alpha^i\beta^i & \beta^i & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ (\alpha^i)^{N_p-1}\beta^i & (\alpha^i)^{N_p-2}\beta^i & \dots & \beta^i \end{bmatrix} \begin{bmatrix} u^i_{k|k} \\ u^i_{k+1|k} \\ \vdots \\ u^i_{k+N_p-1|k} \end{bmatrix} + \begin{bmatrix} \gamma^i & 0 & \dots & 0 \\ \alpha^i\gamma^i & \gamma^i & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ (\alpha^i)^{N_p-1}\gamma^i & (\alpha^i)^{N_p-2}\gamma^i & \dots & \gamma^i \end{bmatrix} \begin{bmatrix} a^{i-1}(k) \\ a^{i-1}(k) \\ \vdots \\ a^{i-1}(k) \end{bmatrix}.$$

Therefore, in a more compact form, where the predicted states are functions of the calculated control inputs only, the prediction model can be written as,

$$X^i_k = \zeta^i x^i(k) + \nu^i U^i_k + \varkappa^i \Lambda^{i-1}_k, \tag{40}$$

where,

$$X^i_k = \begin{bmatrix} x^i_{k+1|k} \\ x^i_{k+2|k} \\ \vdots \\ x^i_{k+N_p|k} \end{bmatrix},\ \zeta^i = \begin{bmatrix} \alpha^i \\ (\alpha^i)^2 \\ \vdots \\ (\alpha^i)^{N_p-1} \end{bmatrix},\ \nu^i = \begin{bmatrix} \beta^i & 0 & \dots & 0 \\ \alpha^i\beta^i & \beta^i & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ (\alpha^i)^{N_p-1}\beta^i & (\alpha^i)^{N_p-2}\beta^i & \dots & \beta^i \end{bmatrix},\ U^i_k = \begin{bmatrix} u^i_{k|k} \\ u^i_{k+1|k} \\ \vdots \\ u^i_{k+N_p-1|k} \end{bmatrix},$$

$$\Lambda^{i-1}_k = \begin{bmatrix} a^{i-1}(k) \\ a^{i-1}(k) \\ \vdots \\ a^{i-1}(k) \end{bmatrix},\ \varkappa^i = \begin{bmatrix} \gamma^i & 0 & \dots & 0 \\ \alpha^i\gamma^i & \gamma^i & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ (\alpha^i)^{N_p-1}\gamma^i & (\alpha^i)^{N_p-2}\gamma^i & \dots & \gamma^i \end{bmatrix}$$

Cost function: The cost function defined in Eq. 31 has to be defined over the horizon N_p. It is reformulated as follows,

$$J(x^i(k), U^i_k) = \left(x^i_{k+N_p|k}\right)^T P\left(x^i_{k+N_p|k}\right) + \sum_{j=0}^{N_p-1}\left[\left(x^i_{k+j|k}\right)^T Q\left(x^i_{k+j|k}\right) + \left(u^i_{k+j|k}\right)^T R\left(u^i_{k+j|k}\right)\right]. \tag{41}$$

The elements of the weighting matrix Q are the corresponding weighting coefficients that penalize the distance of the predicted states $x^i_{k+j|k}$ to zero. Similarly R is the weighting parameter that penalizes the control inputs $u^i_{k+j|k}$. Q and R are defined as follows,

$$Q = \begin{bmatrix} w_a & 0 & 0 & 0 \\ 0 & w_{da} & 0 & 0 \\ 0 & 0 & w_{\Delta d} & 0 \\ 0 & 0 & 0 & w_{\Delta v} \end{bmatrix} \in \mathbb{R}^{4\times4},\ R = [\,w_u\,] \in \mathbb{R}^{1\times1}.$$

P and Q are positive semi-definite matrices ($P \succeq 0$ and $Q \succeq 0$). R is positive definite i.e. $R \succ 0$. P is the terminal cost that penalizes the terminal state $x^i_{k+N_p|k}$.

By expanding the summation over the horizon N_p, the cost function can be written as a function of the predicted states and calculated inputs as follows,

$$J(x^i(k), U^i_k) = \left(x^i_{k|k}\right)^T Q x^i_{k|k} + \underbrace{\begin{bmatrix} x^i_{k+1|k} \\ x^i_{k+2|k} \\ \vdots \\ x^i_{k+N_p|k} \end{bmatrix}^T}_{(X^i_k)^T} \underbrace{\begin{bmatrix} Q & 0 & \dots & 0 \\ 0 & Q & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & P \end{bmatrix}}_{\overline{Q}} \underbrace{\begin{bmatrix} x^i_{k+1|k} \\ x^i_{k+2|k} \\ \vdots \\ x^i_{k+N_p|k} \end{bmatrix}}_{X^i_k} + \underbrace{\begin{bmatrix} u^i_{k|k} \\ u^i_{k+1|k} \\ \vdots \\ u^i_{k+N_p-1|k} \end{bmatrix}^T}_{(U^i_k)^T} \underbrace{\begin{bmatrix} R & 0 & \dots & 0 \\ 0 & R & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & R \end{bmatrix}}_{\overline{R}} \underbrace{\begin{bmatrix} u^i_{k|k} \\ u^i_{k+1|k} \\ \vdots \\ u^i_{k+N_p-1|k} \end{bmatrix}}_{U^i_k}.$$

Since $x^i_{k|k} = x^i(k)$, we have,

$$J\left(x^i(k), U^i_k\right) = \left(x^i(k)\right)^T Q x^i(k) + \left(X^i_k\right)^T \overline{Q} X^i_k + \left(U^i_k\right)^T \overline{R} U^i_k. \tag{42}$$

The above cost function can be simplified by using the definition of X^i_k (see Eq. 40) which yields,

$$J\left(x^i(k), U_k^i\right) = x^i(k)^T Q x^i(k) + \left(U_k^i\right)^T \overline{R} U_k^i + \left(\zeta^i x^i(k) + \nu^i U_k^i + \varkappa^i \Lambda_k^{i-1}\right)^T \overline{Q}\left(\zeta^i x^i(k) + \nu^i U_k^i + \varkappa^i \Lambda_k^{i-1}\right).$$

After simplifying the above expression, the cost function in a compact form is expressed as,

$$J\left(x^i(k), U_k^i\right) = \frac{1}{2}\left(U_k^i\right)^T G^i U_k^i + {U_k^i}^T \mathcal{F}^i, \tag{43}$$

where,

$$G^i = 2\left(\overline{R} + \left(\nu^i\right)^T \overline{Q} \nu^i\right), \tag{44}$$

$$\mathcal{F}^i = 2\left(\nu^i\right)^T \overline{Q}\left(\zeta^i x^i(k) + \varkappa^i \Lambda_k^{i-1}\right). \tag{45}$$

One can prove the positive definiteness of G^i since $\overline{R} \succ 0$ and $\overline{Q} \succ 0$. It is noted that while simplifying the above formula, the terms $\left(x^i(k)^T\left[Q + (\zeta^i)^T \overline{Q} \zeta^i\right] x^i(k)\right)$ and $\left(2(\Lambda_k^{i-1})^T (\varkappa^i)^T \overline{Q} \varkappa^i \Lambda_k^{i-1}\right)$ were omitted. Since they are not functions of the predicted control input U_k^i so they can be considered as constants (the value that minimizes the cost function will not change by adding constants to the cost function).

Constraints matrices: In this section, we expand the previously defined constraints over the prediction horizon N_p and reformulate them in a more compact form.

The constraints defined by Eq. 35, Eq. 32 and Eq. 34 can be combined in one inequality as follows,

$$x_{min} \leqslant x^i(k) \leqslant x_{max}, \tag{46}$$

where $x_{min} = \begin{bmatrix} a_{min} \\ \delta a_{min} \\ \Delta d_{min} \\ \Delta v_{min} \end{bmatrix}$ and $x_{max} = \begin{bmatrix} a_{max} \\ \delta a_{max} \\ \Delta d_{max} \\ \Delta v_{max} \end{bmatrix}$. To include the rear-end collision avoidance constraint (Eq. 36) to the inequality Eq. 46, we need to rewrite d^i as a function of Δd^i and Δv^i. From Eq. 16 we know that $d^i = \Delta d^i + d_{des}^i$, then Eq. 36 can be written as,

$$d_0 \leqslant \Delta d^i + d_{des}^i \leqslant d_{max},$$

Substituting d_{des}^i in the above inequality with its definition from Eq. 18, we obtain,

$$0 \leqslant \Delta d^i + \tau_h v^i \leqslant d_{max} - d_0.$$

As $v^i = v^{i-1} - \Delta v^i$ (see Eq. 17), the above inequality can be written as,

$$-\tau_h v^{i-1} \leqslant \Delta d^i - \tau_h \Delta v^i \leqslant d_{max} - d_0 - \tau_h v^{i-1}. \tag{47}$$

Therefore, the constraints on the state $x^i(k)$ are defined as,

$$\underbrace{\begin{bmatrix} a_{min} \\ \delta a_{min} \\ \Delta d_{min} \\ \Delta v_{min} \\ -\tau_h v^{i-1} \end{bmatrix}}_{\overline{x}_{min}} \leqslant \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & -\tau_h \end{bmatrix}}_{\widetilde{A}} \begin{bmatrix} a^i(k) \\ \delta a^i(k) \\ \Delta d^i(k) \\ \Delta v^i(k) \end{bmatrix} \leqslant \underbrace{\begin{bmatrix} a_{max} \\ \delta a_{max} \\ \Delta d_{max} \\ \Delta v_{max} \\ d_{max} - d_0 - \tau_h v^{i-1} \end{bmatrix}}_{\overline{x}_{max}},$$

or,

$$\overline{x}_{min} \leqslant \widetilde{A} x^i(k) \leqslant \overline{x}_{max}. \tag{48}$$

To obtain the constraints of the predicted states over the horizon N_p, the above inequality can be redefined as,

$$\overline{x}_{min} \leqslant \widetilde{A} x_{k+j+1|k}^i \leqslant \overline{x}_{max}, \qquad j = 0, \ldots, N_p - 1. \tag{49}$$

Similarly, the constraints on the control input $a_{min} \leqslant u^i(k) \leqslant a_{max}$ (Eq. 33) can be generalized to represent the predicted control inputs over the horizon N_p as follows,

$$a_{min} \leqslant u_{k+j|k}^i \leqslant a_{max}, \qquad j = 0, \ldots, N_p - 1. \tag{50}$$

with a_{min} and a_{max} are the lower and upper bounds of the desired acceleration. Combining Eq. 49 and Eq. 50 in one inequality, we obtain,

$$\underbrace{\begin{bmatrix} \mathbf{0}_{1\times4} \\ \mathbf{0}_{1\times4} \\ -\widetilde{A}_{5\times4} \\ \widetilde{A}_{5\times4} \end{bmatrix}}_{M_j} x^i_{k+j+1|k} + \underbrace{\begin{bmatrix} -1 \\ 1 \\ \mathbf{0}_{5\times1} \\ \mathbf{0}_{5\times1} \end{bmatrix}}_{E_j} u^i_{k+j|k} \leqslant \underbrace{\begin{bmatrix} -a_{min} \\ a_{max} \\ \bar{x}_{max} \\ -\bar{x}_{min} \end{bmatrix}}_{b_j},$$

or,

$$M_j x^i_{k+j+1|k} + E_j u^i_{k+j|k} \leqslant b_j, \qquad j = 0, \ldots, N_p - 1, \tag{51}$$

where $M_j \in \mathbb{R}^{12\times4}, E_j \in \mathbb{R}^{12\times1}, b_j \in \mathbb{R}^{12\times1}$. By separating the inequality on the terminal constraint, we obtain,

$$M_j x^i_{k+j|k} + E_j u^i_{k+j|k} \leqslant b_j, \qquad j = 0, \ldots, N_p - 1 \tag{52}$$

$$M_{N_p} x^i_{k+N_p|k} \leqslant b_{N_p} \tag{53}$$

where the terminal constraint matrices M_{N_p} and b_{N_p} have to be computed for ensuring stability (Borrelli et al., 2017). It is proved in Limón et al. (2006) that stability of MPC can be obtained when considering a zero terminal constraint set. In this work, we consider M_{N_p} and b_{N_p} to be zero.

By expanding the above inequalities over the horizon N_p, we obtain the following,

$$\underbrace{\begin{bmatrix} M_0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}}_{D} x^i_{k|k} + \underbrace{\begin{bmatrix} 0 & 0 & \ldots & 0 \\ M_1 & 0 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & M_{N_p} \end{bmatrix}}_{M} \underbrace{\begin{bmatrix} x^i_{k+1|k} \\ x^i_{k+2|k} \\ \vdots \\ x^i_{k+N_p|k} \end{bmatrix}}_{X^i_k} + \underbrace{\begin{bmatrix} E_0 & 0 & \ldots & 0 \\ 0 & E_1 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & E_{N_p-1} \\ 0 & 0 & \ldots & 0 \end{bmatrix}}_{\iota} \underbrace{\begin{bmatrix} u^i_{k|k} \\ u^i_{k+1|k} \\ \vdots \\ u^i_{k+N_p-1|k} \end{bmatrix}}_{U^i_k} \leqslant \underbrace{\begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_{N_p} \end{bmatrix}}_{C}.$$

Knowing that $x^i_{k|k} = x^i(k)$, we obtain,

$$Dx^i(k) + MX^i_k + \iota U^i_k \leqslant C. \tag{54}$$

Using the definition of X^i_k (Eq. 40), the constraints can be rewritten as,

$$LU^i_k \leqslant C + wx^i(k) - M\varkappa^i \Lambda^{i-1}_k \tag{55}$$

where $L = \iota + M\nu^i$ and $w = -D - M\zeta^i$.

Minimizing the cost function Eq. 43 under the constraints Eq. 55 gives the following sequence of optimal control inputs over the horizon N_p:

$(U^i_k)^* := \begin{bmatrix} (u^i_{k|k})^* \\ (u^i_{k+1|k})^* \\ \vdots \\ (u^i_{k+N_p|k})^* \end{bmatrix}$, where only the first control input $(u^i_{k|k})^*$ will be applied. The other elements are discarded. At the next step $k+1$, a new sequence $(U^i_{k+1})^*$ is computed and only the first optimal value $(u^i_{k+1|k+1})^*$ is applied. It should be noted that in case of packet losses the next optimal value can be used.

Remark 2. It is assumed that the leader platoon vehicle is manually controlled by a human driver. Thus, the platoon leader is equipped with only the previously explained lower-layer control to track the desired acceleration profile.

9. Simulation results

9.1. Key parameters under consideration

In this work, we are interested in fuel efficiency while maintaining safety by ensuring string stability and good tracking performance. We evaluate the performance of our platooning algorithms with respect to three key parameters – lower-level sampling period h_l, headway time τ_h and level of congestion. The value of h_l depends on the sensing and actuating technologies as well as in-vehicle communication technologies. The value of τ_h is directly related to the safety aspects and chosen based on the target level of autonomy and fuel economy. The congestion level further depends on the traffic (number of vehicles equipped with V2V devices). We evaluate the influence and interplay of these three key factors using CReTS (Ibrahim et al., 2018). In this work, we consider three different

representative of lower-layer sampling periods $h_l = 2$ ms, 10 ms, and 20 ms. The $h_l = 10$ ms, and 20 ms are feasible in the currently used sensing, actuating and in-vehicle communication (e.g., CAN). The $h_l = 2$ ms requires advanced (possibly future) sensing and actuating technologies along with faster communication buses such as FlexRay and TT-Ethernet. In particular, we need to measure vehicle acceleration in real-time to realize the proposed control law. With respect to sensing technologies for this purpose, low-cost accelerometers operate at up to 100 Hz and 3-axis digital accelerometers which are available in the market have an output data rate up to 1000 Hz possibly making $h_l = 2$ ms feasible in vehicles.

We consider a wide range of headway time, i.e., $0.2 \text{ s} \leqslant \tau_h \leqslant 2$ s. This range is chosen taking inspiration from literature and considering the safety and fuel efficiency aspects. For example, in Zegers et al. (2017), the headway time is 0.3 s and $d_0 = 2$ m. Moreover, field experiments have been done in the Energy ITS Project (Tsugawa, 2013) where three heavy trucks and one light truck were fully automated and drove with 80 km/h and 4.7 m inter-vehicle distance (i.e. headway time = 0.21 s). The choice of these numbers and associated safety are highly dependent on the reliability of communication, sensing, actuating, and control design techniques. With the proposed control technique, by taking into account the characteristics of communication protocols, we show that it is feasible to achieve a shorter headway time than state-of-the-art methods like (Zegers et al., 2017) and hence we demonstrate a higher fuel-efficiency.

9.2. Simulation scenarios

We evaluate the performance of the platoon vehicles in terms of string stability, tracking capability and fuel economy under different congestion levels, headway-time values and lower-layer sampling periods. We use CReTS framework for the evaluation considering realistic network behavior. As mentioned ealier, CReTS is a co-simulation framework consisting of the ns-3 (network simulator), SUMO (traffic simulator) and Matlab (for control design and interfacing between ns-3 and SUMO). In our simulations, we consider five platoon vehicles. The platoon leader is named Veh0, the 1^{st} following vehicle is Veh1, the 2^{nd} follower is Veh2, the 3^{rd} is Veh3, the 4^{th} is Veh4. A realistic highway scenario is simulated in SUMO. We consider a road section of 3 km length with 4 lanes in each direction (see Fig. 6). Vehicles move from the left to the right and when they reach the end of the road they take a U-turn and move in the opposite direction. The lanes are numbered from one to four. Lane 1 is dedicated to platoon vehicles. The length of platoon vehicles and additional traffic vehicles is 4 m. The velocity of the additional traffic vehicles ranges from 0 to 30 m/s whereas their acceleration ranges from -2 to 3 m/s^2. To eliminate the effects of the network boundaries, we restrict our performance evaluation to the region of interest of 1 km in the 3 km road shown in Fig. 6. In ns-3, we consider communication parameters reported in ETSI (2015); they are listed in Table 2. The channel model parameters are obtained from the highway scenario specification in the ETSI standard (ETSI, 2012).

To show the robustness of the control algorithm and its reliability to packet losses, different simulation scenarios have been developed to create congestion in the communication channel used by the platoon and the vehicles on the highway. It is assumed that all the vehicles on the highway communicate via IEEE 802.11p, so by changing vehicle density on the highway, we obtain different network congestion and therefore different control performance. The number of vehicles in each scenario is shown in Table 3. Scenario-0 in Table 3 considers the case where only platoon vehicles are equipped with V2V devices and therefore perfect communication is experienced (no packets losses and minimum delay). In Scenario-1, we test the controller in case of 300 vehicles equipped with V2V devices (including the five platooned vehicles). In Scenario-2, 600 vehicles are equipped with V2V devices. Similarly, Scenario-3 and Scenario-4 define the cases where 800 vehicles and 1000 vehicles are equipped with V2V devices, respectively. The average distance between non-platoon vehicles in our simulation scenarios can be seen also in Table 3. For example in Scenario-4, the

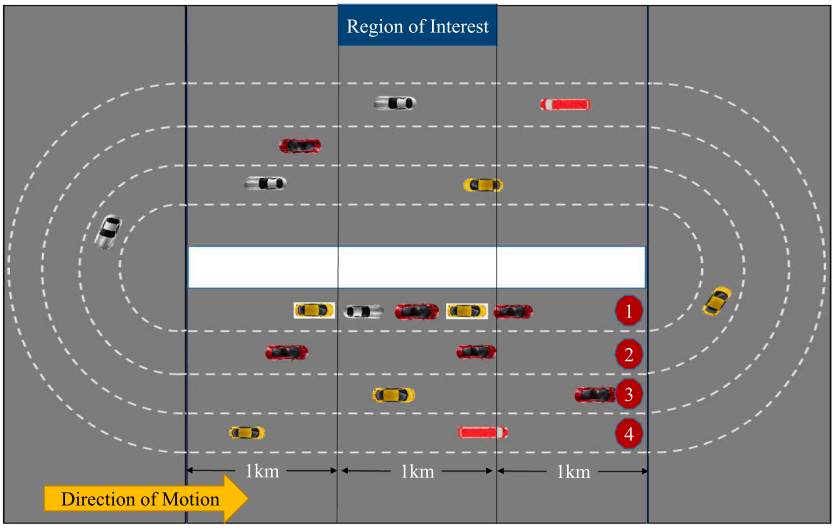


Fig. 6. SUMO highway traffic scenario. Lane 1 is dedicated to platoon vehicles and all vehicles communicate via the IEEE 802.11p.

Table 2
Communication parameters.

Carrier sensing threshold	−85 dBm	
Transmit power	24 dBm	
(Large scale fading)	Path loss exponent 1 (until 80 m)	1.9
Dual slope model (ETSI, 2012)	Path loss exponent 2 (after 80 m)	3.8
	Distance bin in meters	m
(Small scale fading)	0–50	3
Nakagami m model (ETSI, 2012)	51–150	1.5
	> 150	1
Payload size	300 Bytes	

Table 3
Simulation scenarios used in the CReTS (Ibrahim et al., 2018) framework.

	Number of vehicles	Level of congestion	Channel load	Average non-platoon vehicle-vehicle distance
Scenario-0	5	low	<70%	–
Scenario-1	300	low	<70%	80 m
Scenario-2	600	medium	≈70%	40 m
Scenario-3	800	high	≈100%	30 m
Scenario-4	1000	traffic jam	>100%	24 m

average distance between vehicles on an 8-lane highway of 3 km length is calculated as: 3000/(1000/8)=24 m (see Section 6.2). Scenario-0 and Scenario-1 represent a highway at low congestion level (i.e. channel load < 70%). Scenario-2 shows a moderately congested highway with channel load ≈ 70%. Scenario-3 and Scenario-4 represent a highly congested highway with channel load > 70%. Scenario-0–Scenario-4 defined in Table 3 are simulated in CReTS with lower-layer sampling period $h_l = 2$ ms for headway time $\tau_h = 0.2$ s. Moreover, Scenario-2 and Scenario-3 are also tested in CReTS with $h_l = 10$ ms for $\tau_h = 0.2$ s and 0.8 s. Additionally, Scenario-0 is simulated for a wider range of h_l and τ_h i.e. for $h_l = 2$ ms, 10 ms, 20 ms and $\tau_h = 0.2$ s,…,2 s.

Remark 3. We consider Scenario-3 and Scenario-4 for evaluation purpose only. As already explained, if the channel load > 70% then message rate should be lowered and controlled by a DCC algorithm (part of our future work). Also, in case of persistent packet loss with a high channel load, vehicle control should switch from CACC to ACC where the control action is calculated based on the information received from a vehicle's on-board sensors only. This is beyond the scope of this paper.

9.3. Network performance

The behavior of the control algorithm depends on the number of packets received at the destination vehicle. Packet reception ratio (PRR) is defined as the percentage of packets received by the destination vehicle from the transmissions by the source vehicle. For the scenarios defined in Table 3, average PRR and average delay have been computed for every wireless link between platoon vehicles. We define the wireless links between platoon vehicles as shown in Table 4. For example, Link-0 considers the wireless link between any two consecutive platoon vehicles with no intermediate vehicles between them. Link-1 considers the wireless link where there is one intermediate vehicle between any two consecutive platoon vehicles. Link-2 considers the case where there are 2 intermediate vehicles between the sender and the receiver. Link-3 defines the unique case where there are 3 vehicles between the sender and the receiver vehicle. It should be noted that only Link-0 is of interest in the control design since we consider PF topology. The average PRR and average delay of other links are represented for completeness; if one is interested in using PLF topology or TPF topology instead of PF topology, the costs of using these links should be clear. We run the simulations five times for each scenario using CReTS for confidence. The average results are reported in Table 5. We notice from Table 5 that the average PRR decreases by increasing the number of intermediary platoon vehicles that exist between the sender and the receiver. Moreover, PRR decreases by increasing the number of communicating vehicles on the highway. On the other hand, simulation results show also that the delay experienced by the messages over the IEEE 802.11p communication standard is low and can be ignored in the controller design given a network with at most a medium level of congestion (600 communicating vehicles over a road length of 3 km); for higher densities of communicating vehicles,

Table 4
V2V links between platoon vehicles.

Link-0	Link-1	Link-2	Link-3
veh0→veh1	*veh0→veh2*	*veh0→veh3*	*veh0→veh4*
veh1→veh2	*veh1→veh3*	*veh1→veh4*	
veh2→veh3	*veh2→veh4*		
veh3→veh4			

Table 5
Average Packet Reception Ratio (PRR) and average delay experienced between platoon vehicles for the five traffic scenarios defined in Table 3. The measurement in this table is averaged for five simulation runs for the case where $\tau_h = 0.2$ s and $h_l = 2$ ms. Similar values can be obtained for other simulation parameters.

		Link-0	Link-1	Link-2	Link-3
Scenario-0	Average PRR % Average delay (ms)	100 3.864	100 4.0231	100 4.9146	100 5.5004
Scenario-1	Average PRR % Average delay (ms)	92.8929 3.4776	92.6095 3.6922	94.0286 4.4462	94.5714 4.5595
Scenario-2	Average PRR % Average delay (ms)	78.4999 7.0207	76.8188 7.0328	75.3131 7.9947	75.4746 7.8943
Scenario-3	Average PRR % Average delay (ms)	65.0625 12.1004	60.9048 11.3835	55.9643 11.0225	47.1786 9.7457
Scenario-4	Average PRR % Average delay (ms)	38.5786 18.1660	33.0286 16.1033	27.4143 14.6430	19.2857 9.7608

communication delays can no longer be ignored.

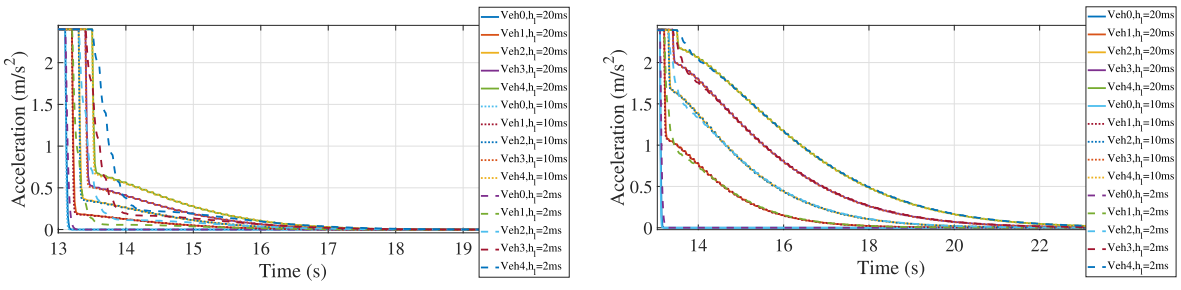
9.4. Control performance – Tracking capability and string stability

The performance of our approach is evaluated in terms of tracking capability (i.e. tracking the acceleration and velocity profiles of preceding vehicles), string stability (attenuation of disturbances), and minimum gap obtained between vehicles. The performance is shown in Figs. 7–13. In order to test our approach under different driving situations, the acceleration trajectory is designed as follows: initially, acceleration gradually increases from 0 to 2.5 m/s^2 and is then kept fixed at 2.5 m/s^2 for 10 s (see Fig. 8a for example). As shown in Fig. 8b, velocity increases from zero to 100 km/h in 13 s. The velocity is kept constant at 100 km/h by dropping the acceleration to zero. The acceleration is kept at zero for 27 s and then sudden braking is applied by dropping the acceleration to -3 m/s^2 and keeping it fixed for 5 s at -3 m/s^2. Then the velocity decreases from 100 km/h to 40 km/h in 5 s. Again, sudden increase in acceleration is applied where acceleration increases from zero to 1.5 m/s^2. Then the velocity increases from 40 km/h to 94 km/h in 10 s.

The following parameters are used in our simulations: $h_u = 0.1$, $d_0 = 1$. Pole locations of the lower-layer controller are chosen at [0.9, 0.9] for $h_l = 2$ ms, [0.2, 0.2] for $h_l = 10$ ms and [0.1, 0.1] for $h_l = 20$ ms. After MPC tuning, the following parameters are selected: $N_p = 15$, $w_{\Delta v} = 50$, $w_{\Delta d} = 80$, $w_u = 30$, $w_{\delta_a} = 10$, $w_a = 0$. In the following, we show influence of the three key parameters on the tracking performance and string stability.

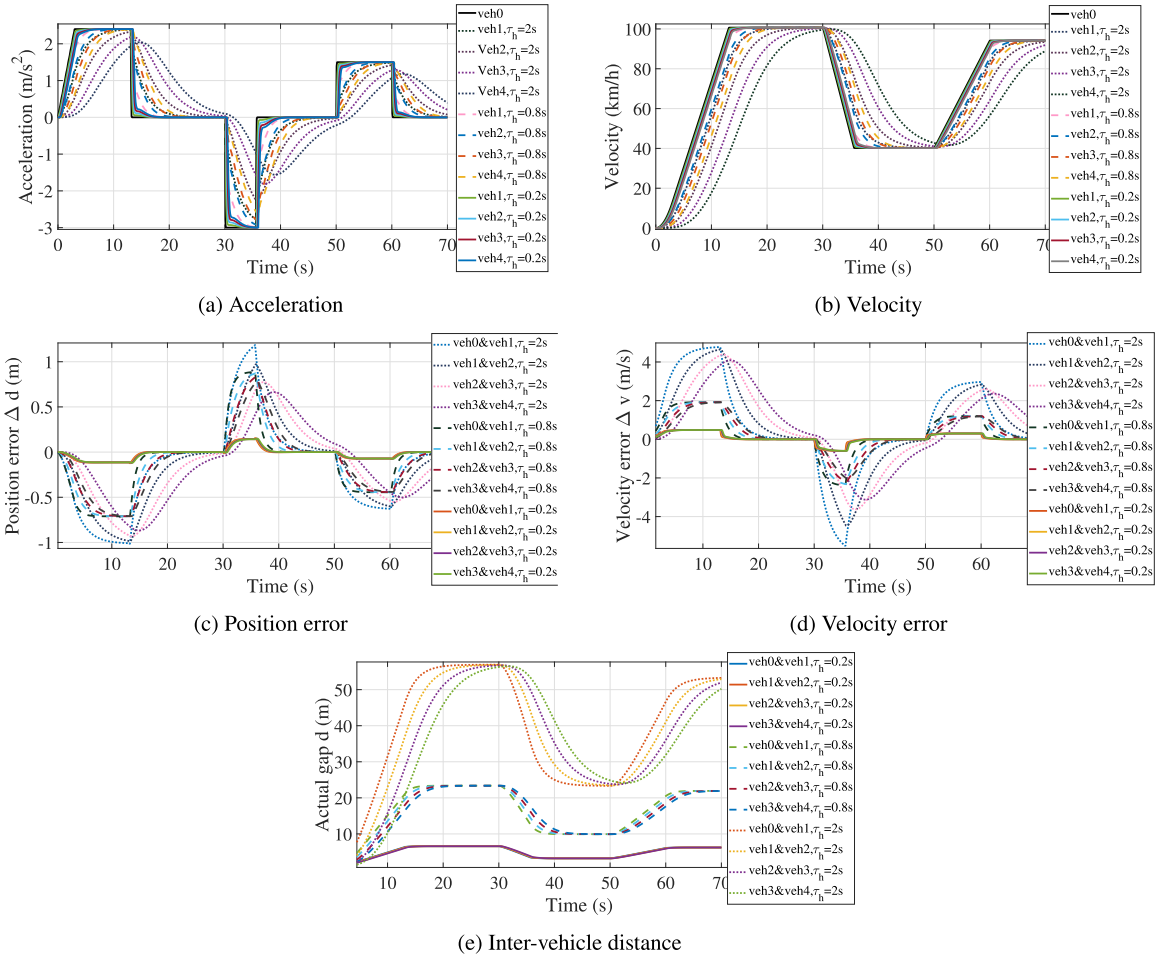
9.4.1. The performance under different lower layer sampling periods

Fig. 7 compares between the acceleration of the platoon vehicles for $h_l = 2$ ms, 10 ms, 20 ms under headway time $\tau_h = 0.2$ s (Fig. 7a) and $\tau_h = 0.8$ s (Fig. 7b). In case of short headway time as in Fig. 7a, $h_l = 2$ ms makes the system smoother during convergence to zero compared to the cases for $h_l = 10$ ms and 20 ms. Therefore, a short h_l is required if a short τ_h is applied to avoid abrupt changes in acceleration. Of course this is challenging since it requires fast sensing, actuating and in-vehicle communication as we already explained. From Fig. 7b it is noted that there is not much difference for the different h_l values for a long τ_h. This happens because of the long inter-vehicle distance and the change in acceleration is slow. On the contrary, for a short headway time, e.g., 0.2 s a fast response is required because of the very short gap between vehicles in order to avoid collision. We notice that string stability is preserved for both cases.



(a) Headway time $\tau_h = 0.2$s (b) Headway time $\tau_h = 0.8$s

Fig. 7. {Scenario-0}, $h_l = \{2$ ms, 10 ms, 20 ms$\}$ and $\tau_h = \{0.2$ s, 0.8 s$\}$.



(a) Acceleration (b) Velocity

(c) Position error (d) Velocity error

(e) Inter-vehicle distance

Fig. 8. {Scenario-0}, $h_l = \{2\text{ ms}\}$ and $\tau_h = \{0.2\text{ s}, 0.8\text{ s}, 2\text{ s}\}$.

9.4.2. The performance under different headway-time values

Fig. 8 compares the performance of the platoon vehicles under headway times $\tau_h = 0.2$ s (solid lines), 0.8 s (dashed lines), 2 s (dotted lines) for a fixed lower-layer sampling period $h_l = 2$ ms. Fig. 8e shows that the inter-vehicle distance between platoon vehicles reaches 56 m when vehicles drive with speed 100 km/h (see Fig. 8b) for $\tau_h = 2$ s. For $\tau_h = 0.8$ s and 0.2 s, inter-vehicle distance reaches 23 m and 6.5 m, respectively. Clearly, a shorter τ_h leads to a smaller inter-vehicle distance and therefore more fuel savings.

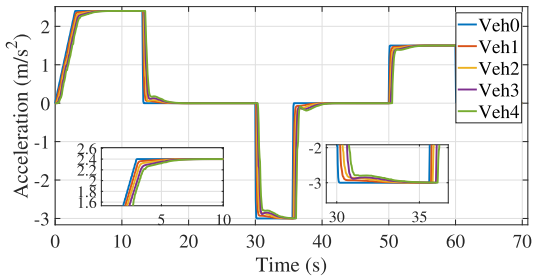
Fig. 8a, b show the acceleration and velocity profiles of the platoon vehicles. It is noted that good tracking is achieved for the lower values of the headway time i.e. for $\tau_h = 0.2$ s, 0.8 s. For $\tau_h = 2$ s, Fig. 8a and b show that tracking performance deteriorates with $\tau_h = 2$ s (i.e., longer τ_h). This can be explained better using Fig. 7. As seen in Fig. 7, by suddenly decreasing the acceleration of the leader vehicle from 2.5 m/s^2 to 0 m/s^2, we notice that the followers in Fig. 7a converge to zero faster (i.e., 4 s) than the followers (i.e., around 9 s) in Fig. 7b. Therefore, a shorter headway time is needed to ensure a nice tracking of the preceding vehicle velocity and acceleration.

Fig. 8c shows that the position error for $\tau_h = 0.2$ s is almost zero, whereas the position error increases for $\tau_h = 0.8$ s and 2 s. Similar behavior is noticed in Fig. 8d where the velocity error Δv is bounded by -5 m/s from below and by 4 m/s from above for $\tau_h = 2$ s. For $\tau_h = 0.8$ s, -2 m/s $< \Delta v <$ 2 m/s and for $\tau_h = 0.2$ s, -0.5 m/s $< \Delta v <$ 0.5 m/s. Therefore, for vehicle platooning, where minimum position and velocity errors are required, a short headway time is mandatory. String stability is preserved for these cases.

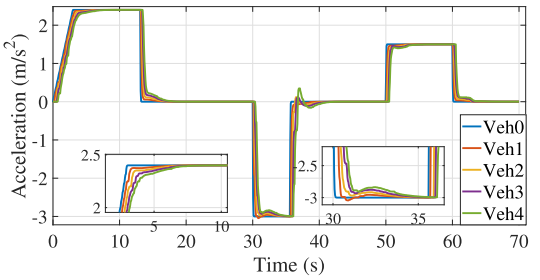
9.4.3. The performance under different congestion levels

Figs. 9–12 show the performance of the platoon vehicles for the traffic scenarios defined in Table 3 for lower-layer sampling rate $h_l = 2$ ms and headway time $\tau_h = 0.2$ s. Furthermore, in Fig. 13, we show the influence of network congestion by considering Scenario-2 (600 vehicles) and Scenario-3 (800 vehicles) for $h_l = 10$ ms and $\tau_h = 0.2$ s and 0.8 s to demonstrate the interplay of congestion level, h_l and τ_h.

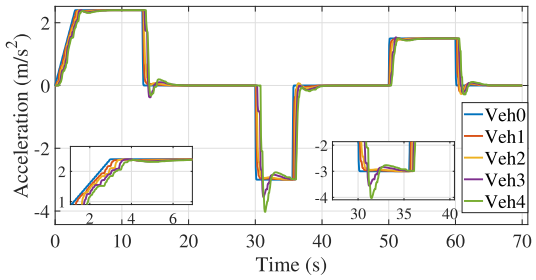
Fig. 9a, b, Fig. 10a, b show a nice tracking capability for platoon vehicles in Scenario-1 and Scenario-2 where string stability is preserved during acceleration and deceleration even in Scenario-2 where packet reception ratio (PRR) = 78.4% compared to PRR = 100% for Scenario-0. Also acceleration profiles do not show huge oscillations which can be translated into smooth and comfortable



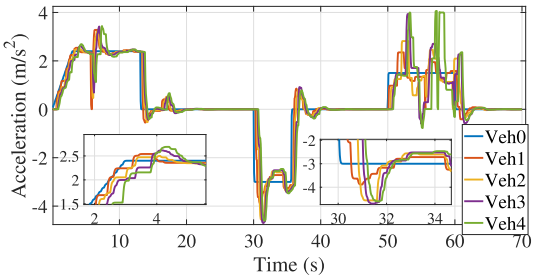
(a) Scenario-1: 300 vehicles use V2V communication



(b) Scenario-2: 600 vehicles use V2V communication

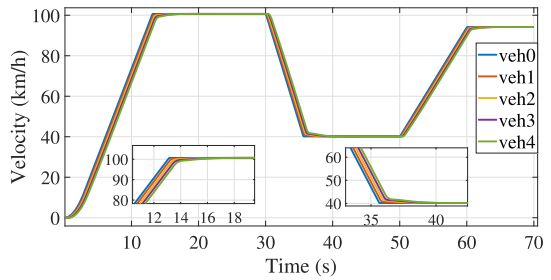


(c) Scenario-3: 800 vehicles use V2V communication

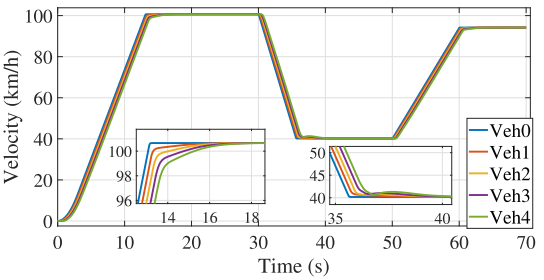


(d) Scenario-4: 1000 vehicles use V2V communication

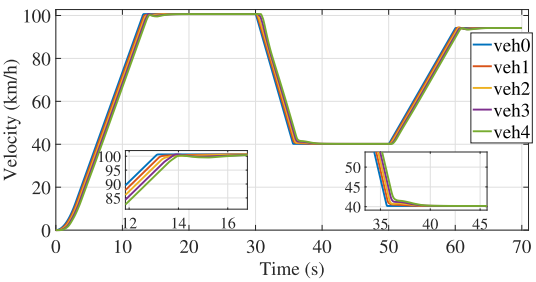
Fig. 9. {Scenario-1–Scenario-4}, $h_l = \{2\text{ ms}\}$ and $\tau_h = \{0.2\text{ s}\}$.



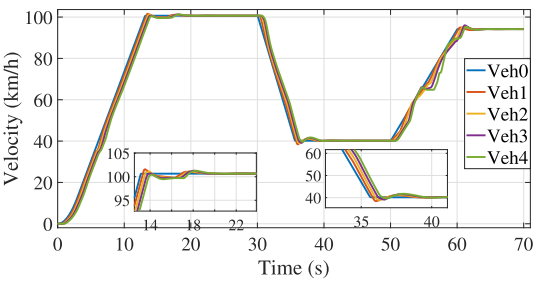
(a) Scenario-1: 300 vehicles use V2V communication.



(b) Scenario-2: 600 vehicles use V2V communication.



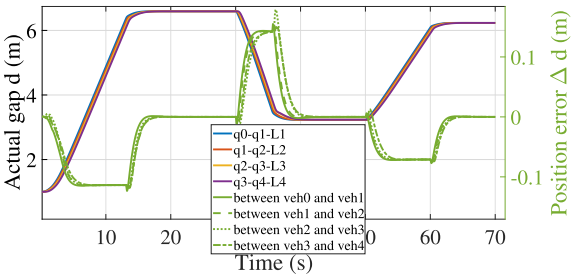
(c) Scenario-3: 800 vehicles use V2V communication.



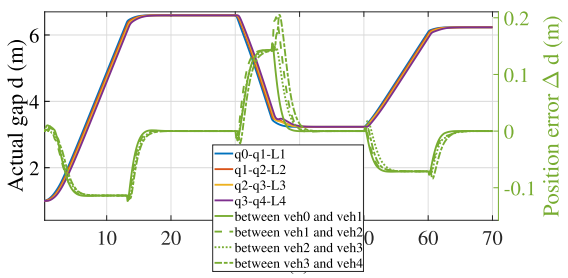
(d) Scenario-4: 1000 vehicles use V2V communication.

Fig. 10. {Scenario-1–Scenario-4}, $h_l = \{2\text{ ms}\}$ and $\tau_h = \{0.2\text{ s}\}$.

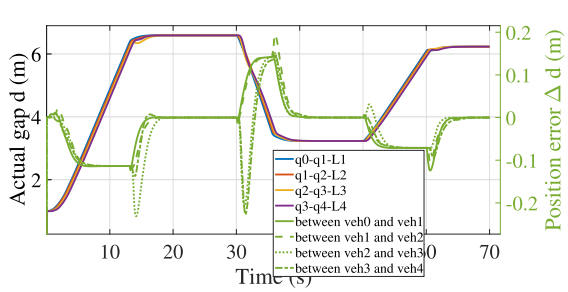
driving. Error in velocity is minimum as seen in Fig. 12a, b. Δv^i is bounded from below by -0.6 m/s and from above by 0.45 m/s for Scenario-1 and Scenario-2. On the other hand, it is hard to guarantee string stability in Scenario-3 and Scenario-4 where a large number of vehicles are equipped with V2V devices and therefore experience more packet loss. Acceleration profiles show high oscillations which affects comfort and maybe safety of passengers, see Fig. 9c, d, Fig. 10c, d. Tracking capability can be analysed for Scenario-3 and Scenario-4 from Fig. 12c, d. Δv^i in Scenario-3, which shows better velocity tracking than Scenario-4, is bounded from below by -0.7 m/s and from above by 0.5 m/s. In Scenario-4, Δv^i is bounded between -1 m/s and 1.5 m/s. Position errors between platoon vehicles are shown in Fig. 11 for the simulation scenarios. In case of constant speed, Δd^i converges to zero. When the vehicle accelerates, Δd^i



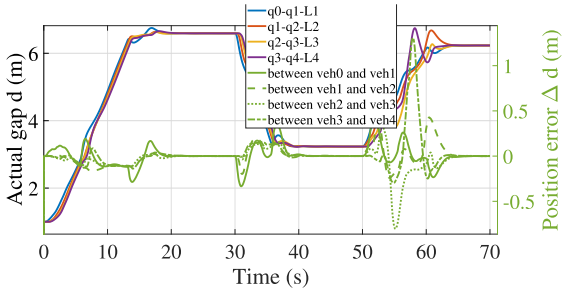
(a) Scenario-1: 300 vehicles use V2V communication.



(b) Scenario-2: 600 vehicles use V2V communication.

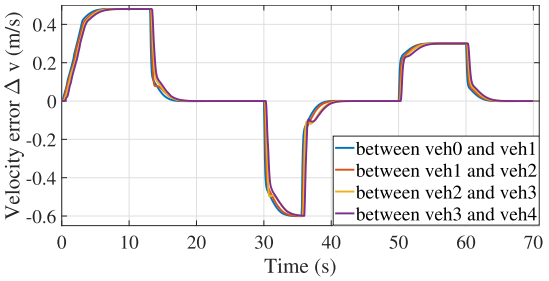


(c) Scenario-3: 800 vehicles use V2V communication.

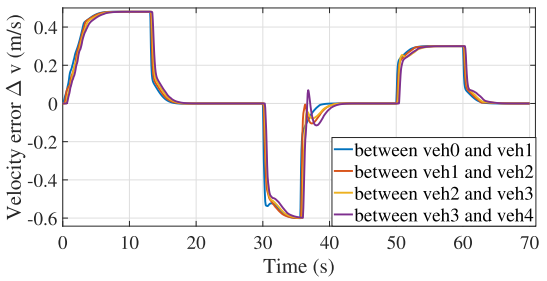


(d) Scenario-4: 1000 vehicles use V2V communication.

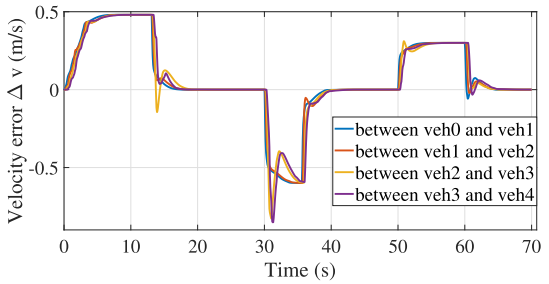
Fig. 11. {Scenario-1–Scenario-4}, $h_l = \{2\text{ ms}\}$ and $\tau_h = \{0.2\text{ s}\}$.



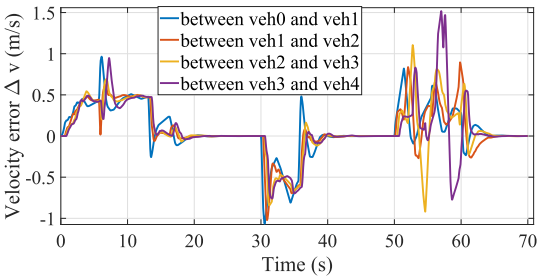
(a) Scenario-1: 300 vehicles use V2V communication.



(b) Scenario-2: 600 vehicles use V2V communication.



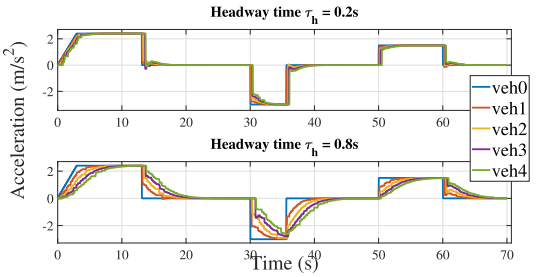
(c) Scenario-3: 800 vehicles use V2V communication.



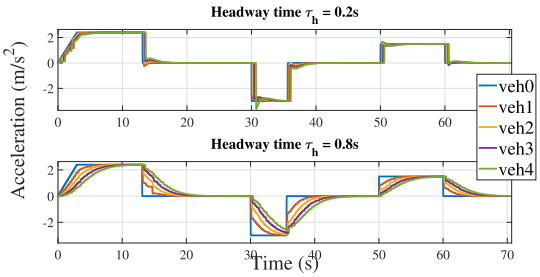
(d) Scenario-4: 1000 vehicles use V2V communication.

Fig. 12. {Scenario-1–Scenario-4}, $h_l = \{2\text{ ms}\}$ and $\tau_h = \{0.2\text{ s}\}$.

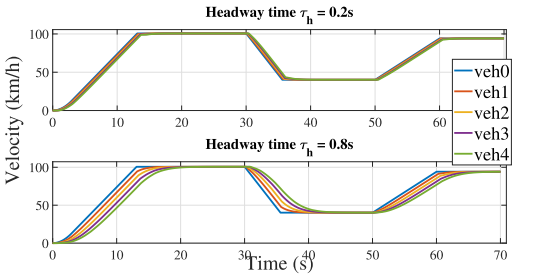
becomes negative. That is because during acceleration, the gap between vehicles increases so the following vehicle tries to minimize the gap to its preceding vehicle by applying more acceleration in order to avoid cut-ins from other adjacent lanes. Similar interpretation applies in case of deceleration where Δd^i becomes positive. The following vehicle brakes hard in order to avoid a crash. Thus the gap between vehicles slightly increases. Fig. 11 shows also the actual gap d^i between vehicles for the simulation scenarios. As stated earlier in Section 6, the gap between vehicles depends on their velocities and the headway time. We notice from Fig. 11 that d^i is constant during the periods where the vehicles move with a constant velocity. d^i increases when vehicles accelerate and d^i decreases during deceleration. Vehicles do not experience crashes in any scenario since $d^i > 1$ m at all times (1 m is the minimum gap between



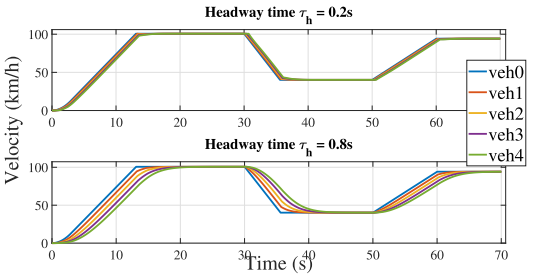
(a) Acceleration of platoon vehicles in Scenario-3.



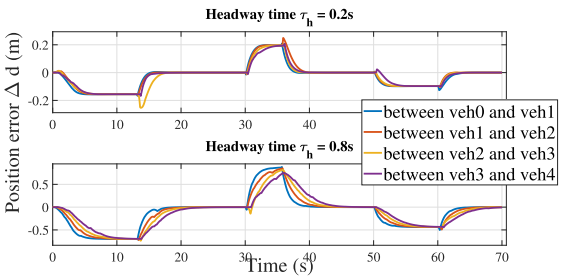
(b) Acceleration of platoon vehicles in Scenario-2.



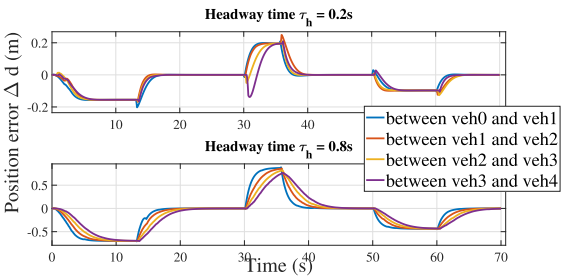
(c) Velocity of platoon vehicles in Scenario-3.



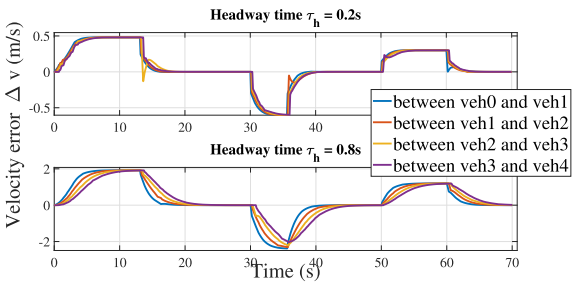
(d) Velocity of platoon vehicles in Scenario-2.



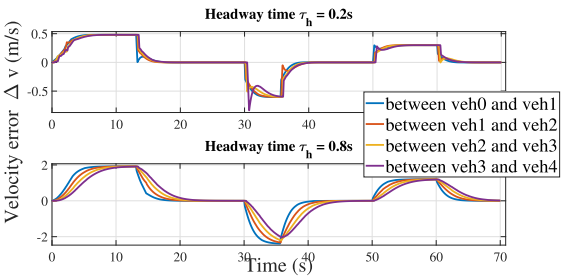
(e) Position error between platoon vehicles in Scenario-3.



(f) Position error between platoon vehicles in Scenario-2.



(g) Velocity error between platoon vehicles in Scenario-3.



(h) Velocity error between platoon vehicles in Scenario-2.

Fig. 13. {Scenario-2, Scenario-3}, h_l = {10 ms} and τ_h = {0.2 s, 0.8 s}.

vehicles at standstill, see Section 6 for more details). Δd^i is bounded between −0.13 m and 0.2 m for Scenario-1 and Scenario-2. By increasing the number of vehicles equipped with V2V devices, communication between vehicles experiences more packet loss and hence position error increases. In Scenario-3 where 62.6% of packets are received, Δd^i ranges between −0.25 m and 0.3 m. Whereas with PRR = 38.6% as in Scenario-4, Δd^i ranges between −0.6 m and 1.5 m.

Scenario-2 and Scenario-3 are simulated using CReTS for $h_l = 10$ ms and $\tau_h = 0.2$ s and 0.8 s and shown in Fig. 13. The average PRR and average delay for those scenarios are reported in Table 6. It is noted that the PRR and delay in Table 6 are very similar to the PRR and delay for the corresponding scenarios in Table 5 (note that all scenarios in Table 5 are simulated with $\tau_h = 0.2$ s and $h_l = 2$ ms). Therefore the network performance is not affected by changing the lower-layer sampling rate and the headway time. Moreover, from Fig. 13, we notice that Scenario-2 and Scenario-3 have similar acceleration and velocity profiles and string stability is preserved for both cases. It is noted that there is no difference between the acceleration and velocity profiles for Scenario-2 with $h_l = 10$ ms (Fig. 13) and Scenario-2 with $h_l = 2$ ms (Fig. 9b and Fig. 10b). We notice the smoothness of the acceleration and velocity profiles for Scenario-3

Table 6
Average Packet Reception Ratio (PRR) and average delay experienced between platoon vehicles for Scenario-2 (600 vehicles) and Scenario-3 (800 vehicles) for $\tau_h = 0.2$ s and 0.8 s and $h_l = 10$ ms.

		Link-0	Link-1	Link-2	Link-3
Scenario-2 $\tau_h = 0.2$ s, $h_l = 10$ ms	Average PRR %	80.8214	78	75.1429	75.5714
	Average delay (ms)	6.8990	6.9470	8.0835	8
Scenario-2 $\tau_h = 0.8$ s, $h_l = 10$ ms	Average PRR %	80.8214	78.4762	74	69.1429
	Average delay (ms)	6.8724	6.9947	8.0832	7.9256
Scenario-3 $\tau_h = 0.2$ s, $h_l = 10$ ms	Average PRR %	71.1786	66.3810	60.4286	51.5714
	Average delay (ms)	12.1170	11.6420	11.4866	10.4765
Scenario-3 $\tau_h = 0.8$ s, $h_l = 10$ ms	Average PRR %	60.1429	55.7143	50.2857	37.8571
	Average delay (ms)	12.0731	11.2378	11.1680	9.8151

with $h_l = 10$ ms even with PRR = 60% for $\tau_h = 0.8$ s and PRR = 71% for $\tau_h = 0.2$ s. This is different from Scenario-3 in Fig. 9c with $h_l = 0.2$ s where huge oscillation occur due to packet loss. It can be concluded that, changing the lower-layer sampling rate does not affect the performance under medium congestion level. However, under high level of congestion, longer lower-layer sampling period behaves better than short $h_l$values. Position error and velocity errors are similar to the cases explained earlier.

9.5. Fuel efficiency analysis

In this section, we analyse fuel consumption of the platooning vehicles for different simulation scenarios. The percentage of air-drag reduction is directly related to the inter-vehicle gap (Deng, 2016; Hussein and Rakha, 2020). On the other hand, the aerodynamic drag force is inversely related to the percentage of air-drag reduction. Thus, a lower inter-vehicle gap implies a higher percentage of air-drag reduction and a lower aerodynamic drag force (and fuel consumption).

9.5.1. Fuel consumption model

As shown in Deng (2016), the fuel consumption model can be represented follows:

$$fuel^i = \frac{1}{H\varpi}\int_{t_0}^{t_f}\left[\varrho\mu m^i g v^i + 0.5\rho AC\left(1-\phi_{drag}\right)\left(v^i\right)^3 + m^i a^i v^i\right]dt, \tag{56}$$

$$\varrho = \begin{cases} 1 & \text{if} \quad \mu m^i g v^i + 0.5\rho AC\left(1-\phi_{drag}\right)\left(v^i\right)^3 + m^i a^i v^i > 0 \\ 0 & \text{otherwise} \end{cases}$$

The above model computes the instantaneously consumed fuel with respect to changing acceleration and velocity. ϕ_{drag} is the air-drag reduction due to the proximity of other vehicles. For passenger cars, ϕ_{drag} can be obtained from Fig. 14 ((Hussein and Rakha, 2020)) for the leading vehicle (the blue curve in Fig. 14), first follower (red curve) and other followers (yellow curve). As can be seen from Fig. 14, air-drag reduction ϕ_{drag} is inversely related to the inter-vehicle distance d^i. Curve fitting can be used to approximate the data points in Fig. 14 into an power function which can be represented as,

$$\phi_{drag} = \begin{cases} \mathfrak{C} - \mathfrak{A}\left(d^i\right)^{\mathfrak{B}} & 0 \leqslant d^i \leqslant \mathfrak{G} \\ 0 & d^i > \mathfrak{G} \end{cases} \tag{57}$$

The parameters $\mathfrak{A}, \mathfrak{B}, \mathfrak{C}$ can be obtained using curve fitting tools e.g. **cftool** from Matlab. The parameters $\mathfrak{A}, \mathfrak{B}, \mathfrak{C}, \mathfrak{G}$ are shown in Table 7. Moreover, the following parameters are used: $m^i = 1700\ kg,\ \rho = 1.29\ kg/m^3,\ A = 3.12\ m^2,\ C = 0.367,\ g = 9.8\ m/s^2,\ \varpi = 0.4,\ H = 36\ MJ/L$. ϱ is 1 if the engine is active and zero elsewhere. ϖ is the average engine efficiency. H is calorific power of the fuel.

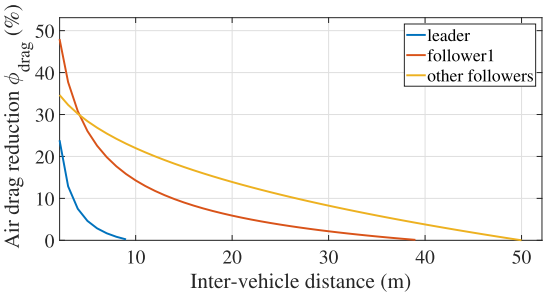


Fig. 14. Air drag reduction ϕ_{drag} adapted from Hussein and Rakha (2020).

Table 7
Parameters for the power function $\phi_{drag} = \mathfrak{C} - \mathfrak{A}(d^i)^{\mathfrak{B}}$ obtained using curve fitting tools for the curves in Fig. 14.

	$\mathfrak{A}$	$\mathfrak{B}$	$\mathfrak{C}$	$\mathfrak{G}$
leader	−0.7538	−1.407	−0.032	10
follower1	−0.9004	−0.4909	−0.148	40
other followers	0.1487	0.3278	0.5361	50

9.5.2. State-of-the-art (SOTA) scenario

In van Nunen et al. (2017), a 25 Hz V2V message rate and a 100 Hz sampling rate of the lower-layer are adopted i.e. $h_l = 10$ ms, $h_u = 40$ ms. We run our algorithm with these new upper and lower sampling rates. For the new system, a few parameters have to be changed. Pole locations of the lower-layer controller have to be chosen closer to zero for the new lower-layer sampling rate; we select poles at [0.2, 0.2] (see Section 5 for more details), horizon length of MPC $N_p = 40$, $h_l = 10$ ms and $h_u = 40$ ms.

9.5.3. Fuel efficiency under different upper-layer and lower-layer sampling periods

In this section, we analyse the fuel consumption for the simulation scenarios described earlier and compare it with the SOTA scenario (van Nunen et al., 2017). Here we analyse the fuel savings where $\tau_h = 0.2$ s. Scenario-2 with $\tau_h = 0.2$ s and $h_l = 10$ ms is named as Scenario-2*. Similarly, Scenario-3 is named as Scenario-3* for $\tau_h = 0.2$ s and $h_l = 10$ ms. Similarly, Scenario-0 with $\tau_h = 0.2$ s and $h_l = 10$ ms is named as Scenario-0* and with $\tau_h = 0.2$ s and $h_l = 20$ ms is named as Scenario-0**.

Table 8 shows the fuel consumption in litre for the platoon vehicles for four cases: (i) Scenario-0–Scenario-4 where the lower-layer sampling period $h_l = 2$ ms (ii) Scenario-0* and Scenario-2*, Scenario-3* where $h_l = 10$ ms (iii) Scenario-0** where $h_l = 20$ ms (v) SOTA scenario where $h_l = 10$ ms and $h_u = 40$ ms. From Table 8, we notice that consumed fuel decreases towards the end of the platoon for all scenarios. In comparison to the SOTA scenario, other scenarios consume less fuel due to the smooth behavior related to acceleration and velocity. However, the fuel consumption increases for scenarios with high lower-layer sampling rate i.e. 10 ms (Scenario-0*, Scenario-2*, Scenario-3*) and 20 ms (Scenario-0**) as seen in Table 8. Table 9 shows the fuel savings of the platoon vehicles which are normalized with respect to the SOTA scenario. We notice from Table 9 that platoon vehicles have an average savings of 18% over the SOTA scenario for the scenarios with lower-layer sampling rate $h_l = 2$ ms i.e. Scenario-0–Scenario-4. Fuel saving with respect to the SOTA scenario goes down with longer lower-layer sampling periods in Scenario-0*, Scenario-2*, Scenario-3* and Scenario-0**. Moreover, scenarios with high lower-layer sampling period h_l also consume more fuel than the scenarios with low h_l values. That is because of the abrupt changes in acceleration and velocity encountered with high h_l and low h_u.

9.5.4. Fuel efficiency under different headway-time values and different lower-layer sampling rates

In this section, we analyse the fuel consumption under headway time $0.3\text{ s} \leqslant \tau_h \leqslant 2$ s and for lower-layer sampling periods $h_l = 2$ ms, 10 ms, 20 ms. For a fair comparison, we consider an acceleration profile for which the controller converges to the desired acceleration (unlike the cases with long τ_h as shown in Fig. 8). Towards this, we consider the following profile: the velocity increases from 0 km/h to 100 km/h in 40 s and remain fixed for 16.6 min. We run the control algorithm with this new trajectory for $0.3\text{ s} \leqslant \tau_h \leqslant 2$ s and $h_l = 2$ ms, 10 ms, 20 ms in order to obtain the instantaneously changing acceleration and velocity of the platoon followers.

9.5.5. Absolute fuel savings

Fig. 15 shows the overall fuel consumption for each vehicle in the platoon over the simulation time for the headway time $\tau_h = 0.3$ s, 0.6 s, 0.8 s, 1 s, 1.5 s, 2 s for lower-layer sampling period $h_l = 2$ ms, 10 ms, 20 ms. The consumed fuel in Fig. 15 are computed for two cases (i) considering the air-drag reduction Eq. 57 due to increasing/decreasing the gap between vehicles (ii) not considering the air-drag reduction i.e. $\phi_{drag} = 0$.

As seen in Fig. 15, the platoon leader consumes the same amount of fuel in both cases, with/without considering air-drag reduction. Moreover, it is noticed that platoon vehicles consume less fuel for low h_l. Different headway-time values, with/without considering air-drag reduction, do not affect the leader fuel consumption. For other followers, we notice that fuel consumption decreases significantly towards the end of the platoon for low τ_h values, e.g., 0.3 s,…, 1 s while considering air-drag reduction. On the other hand, for large τ_h values, e.g., 1.5 s and 2 s the fuel consumption is not affected by the air-drag reduction due to the huge gap between vehicles. For example, for $\tau_h = 0.3$ s headway and $h_l = 2$ ms, veh4 consumes 1.185 litre whereas veh0 consumes 1.45 litre. On the other hand, with $\phi_{drag} = 0$, veh4 has nearly no fuel reduction, i.e., 1.448 litre. Moreover, for $\tau_h = 2$ s and $h_l = 20$ ms, veh4 consumes 1.691 litre (with/without air-drag reduction) and veh0 consume 1.706 litre. Therefore, there is nearly no fuel reduction for a longer τ_h.

9.5.6. Normalized fuel savings

Fig. 16 shows the normalized fuel savings for the platoon vehicles under headway time $\tau_h = 0.3$ s, 0.4 s, 0.5 s, 0.6 s, 0.7 s, 0.8 s, 0.9 s, 1 s, 1.5 s, 2 s. The normalized fuel savings is computed as follows,

$$\text{fuelsavings} = \frac{\text{fuel}_{\text{with}\phi_{drag}} - \text{fuel}_{\phi_{drag}=0}}{\text{fuel}_{\phi_{drag}=0}}.$$

That is, fuel consumption with considering air-drag reduction and by setting $\phi_{drag} = 0$, is differentiated and divided by the consumed fuel where $\phi_{drag} = 0$. We show fuel savings for $\tau_h = 0.3$ s for lower-layer sampling periods $h_l = 2$ ms, 10 ms and 20 ms. For other τ_h

Table 8
Consumed fuel in litre for the platoon vehicles for simulation scenarios with $\tau_h = 0.2$ s and $h_l = 2$ ms (Scenario-0–Scenario-4), $h_l = 10$ ms (Scenario-0*, Scenario-2*, Scenario-3*), $h_l = 20$ ms (Scenario-0**), and $h_l = 10$ ms and $h_u = 40$ ms (SOTA scenario).

	Consumed fuel $\times 10^{-2}$				
	Veh0	Veh1	Veh2	Veh3	Veh4
Scenario-0	9.1052	8.2103	8.0098	7.9814	7.9552
Scenario-1	9.1052	8.2104	8.0092	7.9823	7.9564
Scenario-2	9.1052	8.2103	8.0205	7.9929	7.9614
Scenario-3	9.1052	8.2060	7.9935	7.9596	7.9320
Scenario-4	9.1052	8.1772	8.0006	7.9537	7.9346
Scenario-0*	9.8286	8.8512	8.6398	8.6099	8.5811
Scenario-2*	9.8286	9.4270	9.0073	8.9712	8.9453
Scenario-3*	9.8286	9.4271	9.0075	8.9682	8.9367
Scenario-0**	10.7249	9.6589	9.4292	9.3964	9.3643
SOTA scenario (van Nunen et al., 2017)	11.1367	10.0215	9.7810	9.7439	9.7085

Table 9
Normalized fuel savings for platooning vehicles in different scenarios with respect to the SOTA scenario (van Nunen et al., 2017).

	Normalized fuel savings over SOTA scenario (van Nunen et al., 2017)				
	Veh0	Veh1	Veh2	Veh3	Veh4
Scenario-0	18.24%	18.07%	18.11%	18.09%	18.06%
Scenario-1	18.24%	18.07%	18.11%	18.08%	18.05%
Scenario-2	18.24%	18.07%	17.99%	17.97%	17.99%
Scenario-3	18.24%	18.12%	18.28%	18.31%	18.29%
Scenario-4	18.24%	18.40%	18.20%	18.37%	18.27%
Scenario-0*	11.75%	11.68%	11.67%	11.64%	11.61%
Scenario-2*	11.75%	5.93%	7.91%	7.93%	7.86%
Scenario-3*	11.75%	5.93%	7.91%	7.96%	7.95%
Scenario-0**	3.69%	3.62%	3.59%	3.57%	3.55%

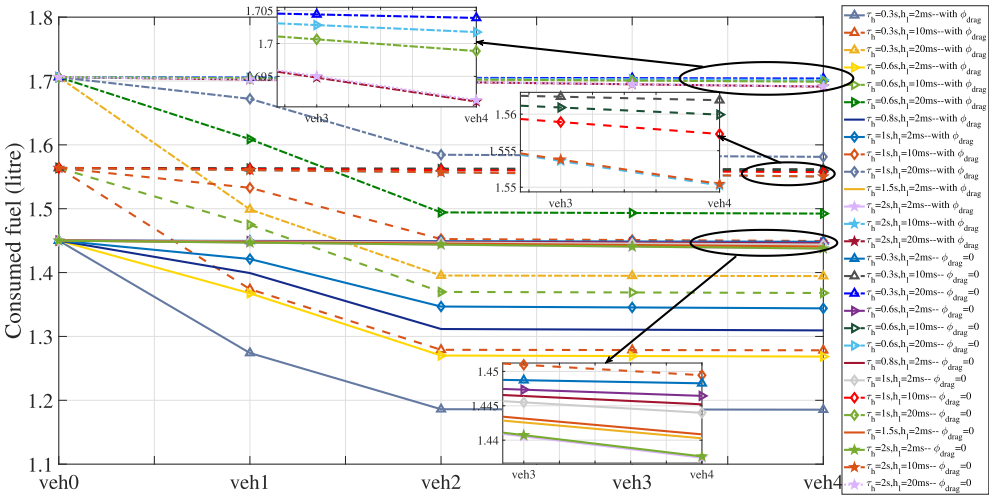


Fig. 15. {Scenario-0}, $h_l = \{2$ ms, 10 ms, 20 ms$\}$ and $\tau_h = \{0.3$ s, 0.6 s, 0.8 s, 1 s, 1.5 s, 2 s$\}$.

values, fuel savings is computed and shown only for $h_l = 2$ ms. It is observed that fuel savings are the same for different h_l values for the same headway time. From Fig. 16, we notice that veh0 has zero fuel saving. The first follower (veh1) has less fuel savings than other followers i.e. veh2, veh3 and veh4. For example, for $\tau_h = 0.3$ s, veh1 shows savings of 12.11%, 5.63% for $\tau_h = 0.6$ s and 1.5% for $\tau_h = 1$ s. On the other hand, other followers have 18% savings for $\tau_h = 0.3$ s, 12.3% for $\tau_h = 0.6$ s and 6.924% for $\tau_h = 1$ s. Therefore, fuel savings is highly affected by the inter-vehicle distance. For long headway-time values e.g. for $\tau_h = 1.5$ s, there is no fuel saving.

10. Embedded implementation

In order to thoroughly test the performance of the previous algorithm in a more realistic environment, we implemented it on

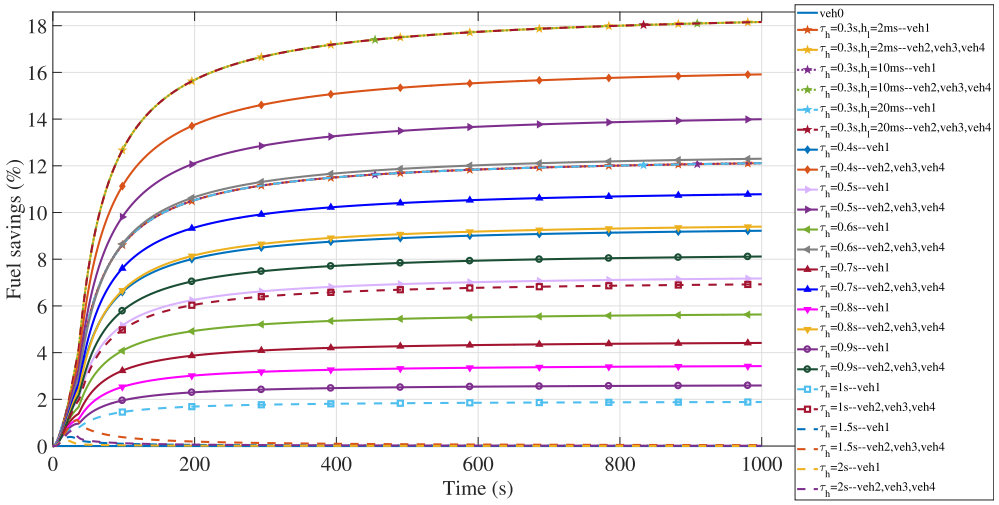


Fig. 16. {Scenario-0}, $h_l = \{2\text{ ms}, 10\text{ ms}, 20\text{ ms}\}$ and $\tau_h = \{0.3\text{ s}, 0.4\text{ s}, 0.5\text{ s}, 0.6\text{ s}, 0.7\text{ s}, 0.8\text{ s}, 0.9\text{ s}, 1\text{ s}, 1.5\text{ s}, 2\text{ s}\}$.

embedded platforms. The implementation was proven to be feasible in Soroa et al. (2019).

10.1. Hardware-in-the-loop (HIL)

We selected the embedded platform Cohda Wireless MK5, developed by Cohda Wireless and NXP for V2X applications. This platform has an NXP i.MX6 DualLite @ 800 MHz processor and 1 GB of volatile memory. It uses an Ubuntu (Linux) distribution as its Operating System (OS), which is not a Real-Time OS (RTOS). We use 4 of these devices running the same algorithm. Each of them simulates a different vehicle in the platoon, creating a platoon with a leader and three followers. The devices are connected to each other and to a computer using a wired network, and on top of that they use the wireless network for communication during the platoon emulation (see Fig. 17).

In this setup, the communication between the vehicles is performed using the 5.9 GHz wireless network, while the wired network is only used for time synchronization and interfacing with the computer. In real vehicles, the time synchronization can be performed using GPS.

In a real vehicle, the GPS measurement is not precise enough to give a reliable measurement of the position and the speed of the vehicle. This measurement would be done using radar, LIDAR or vision sensors, all of which introduce sensor noise. In our emulation the position, the speed, and the acceleration of each vehicle are simulated and sent to the other vehicles using V2V messages. Then the position reading is distorted using random noise equivalent to the noise level of radar systems.

10.2. MPC solver implementation

The MPC problem is solved using a FGM (Fast Gradient Method) solver. This custom FGM solver used in this paper and implemented on MK5 devices is adopted from Kögel and Findeisen (2011). FGM computes the gradient of the cost function for the current sensed state, which, using the compact formulation from Eq. 43, can be computed as:

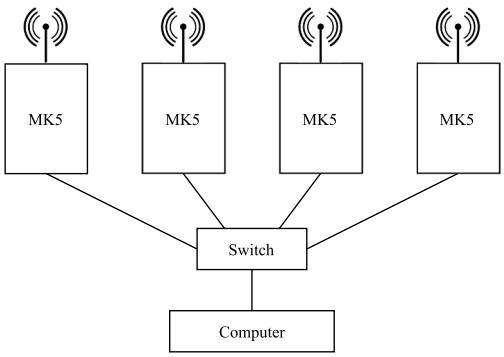


Fig. 17. Hardware setup for the Hardware-In-The-Loop (HIL) emulation.

$$\nabla J = U_k^i G^i + \mathcal{F}^i, \tag{58}$$

where the $\mathcal{F}^i$ matrix is a function of the sensed state $x^i(k)$.

The algorithm used in the Matlab implementation is described in the following, where $P_V(\cdot)$ is a projection function which projects the function on the feasible set. It is computed as

$$P_V(w,h) = arg\ min_{q\in V}\|q-v\|^2,\ v = w - h\nabla J, \tag{59}$$

where V is the feasible constrained set of the control inputs defined as,

$$V = \{a_{min} \leqslant u^i(k) \leqslant a_{max}\}, \tag{60}$$

and h is the step size chosen such that $J(U_{k+1}^i) \leqslant J(U_k^i)$. The maximum number of iterations is computed as,

$$i_{max} \leqslant min\left(\frac{\ln 2\epsilon - \ln Ld^2}{\ln\left(1-\sqrt{\frac{\mu}{L}}\right)}, \sqrt{\frac{2Ld^2}{\epsilon}} - 2\right),\ d^2 = N_p\frac{(a_{max}-a_{min})^2}{2}, \tag{61}$$

where a_{max} and a_{min} are the lower and upper bounds of the control input, respectively, N_p is the horizon length, μ and L are the minimum and maximum eigenvalues of the matrix G^i, and ϵ is the suboptimality level.

The value of the matrix G^i (Eq. 44) and the matrices $2\nu_i^T\overline{Q}\zeta_i$ and $2\nu_i^T\overline{Q}\varkappa_i$ (used to compute F^i, Eq. 45) can be considered to be constant; therefore they are only computed at the initialization. This allows us to reduce the execution time of the MPC solver at run time.

The FGM solver and the matrix computations were first implemented in Matlab and then converted to C using automatic code generation from Matlab (Matlab Coder). The tool is not capable of handling all the Matlab functions, but we need to perform a continuous to discrete model transformation and compute the eigenvalues of G^i, both of which are not standard in C. The continuous to discrete transform has been approximated by finding a linear relation between the discrete model and the time headway (the only value involved in the continuous model that can vary without changing the vehicle). The eigenvalues are computed using the GNU Scientific Library (GSL).

10.3. Lower-layer implementation

The lower-layer is implemented as a function following the controller described in Section 5. This function takes as parameters the values Γ_l^i and Φ_l^i defining the vehicle model (Eq. 11), the values of κ^i and F^i used in the state-feedback controller (Eq. 12), and the full vehicle state ($x_l^i(k)$, the position, and the velocity of the vehicle).

The function computes the state of the vehicle at time $k+1$ using the Eq. 14 for $x_l^i(k+1)$, integrating the acceleration to obtain the velocity, and integrating the velocity to obtain the position. See Eq. 62, where t is the step size (2 ms), s is the position, v is the velocity and a is the acceleration. The variables with the subindex 0 represent the initial value.

$$\int\int a\,dt = \int v\,dt = s,\ s = s_0 + v_0\cdot t + \frac{1}{2}a\cdot t^2,\ v = v_0 + a\cdot t. \tag{62}$$

The output of the function is the updated state of the vehicle. The function can be used as the lower-layer controller of the vehicle running the function but at the same time it can be used to predict the position of the preceding vehicle based on the last received predecessor state and the time elapsed since that information was generated. This is used to compensate the effect of the message delay.

10.4. Causes for non-ideal behaviour

The behaviour of the embedded implementation is not equal to the behaviour of the theoretical version. The mismatch in the behaviour has several causes, some of which are exclusive to the embedded implementation (lower-layer "freezing") and others which would also be present in a real implementation (noise, delays, packet drops).

- **Noise:** Any real sensor has some measurement error. In the case of radar this error is not negligible (Weltzien et al., 1958). In this emulation we add normally distributed noise and we filter it using a simple low-pass digital filter, but the filter is unable to completely restore the original signal.
- **Delays:** There is delay from the moment the vehicle states are sensed until this information is sent to the follower vehicle, received, and used. We reduce the impact of this issue by adding to the V2V message the time stamp at which the vehicle state sent in the message was sensed. This allows the follower to estimate the age of the received information and apply prediction techniques to estimate the current state of the predecessor, see Section 10.3. The prediction assumes that the predecessor uses the same lower-layer controller and assumes that the current acceleration of the predecessor is the same as its desired acceleration. With those

assumptions it can use the lower-layer controller at the follower to predict the movement of the predecessor. This prediction is not perfect because it assumes that the predecessor acceleration remains constant, and the prediction uses 2 ms discrete steps, therefore it can not adapt to the exact amount of delay.

- **Packet drops:** There is interference from other 5.9 GHz sources that results in unpredictable packet drops. When a new message is not received, the upper layer keeps the same value for the desired acceleration.
- **Lower-layer "freezing":** As the Operating System (OS) used in these embedded platforms is not a Real-Time OS (RTOS), it can not guarantee that the execution period of the different tasks will be kept constant. In our emulation, the state of the vehicle is computed (not sensed) and therefore if the lower-layer thread fails to run or is delayed, the vehicle state is not updated (the vehicle doesn't move), and it is equivalent to the vehicle "freezing" in time. These glitches are very short and in a real vehicle they would not happen, as the sensors measurement is independent from the lower-layer execution.

In a real life experiment we can expect to introduce some more errors due to any mismatch between the plant model and the real plant. This model is used for the control and the prediction of the predecessor's state. On top of that we can also expect disturbances from the other vehicles on the road. For a real life experiment it is recommended to use a RTOS.

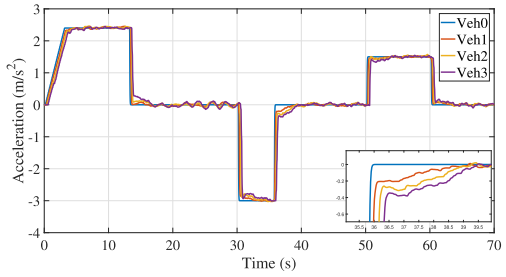
10.5. Results on embedded platform

The embedded implementation results show that this control strategy for vehicle platooning is also satisfactory in a more realistic emulation scenario. In Fig. 18a we can see the acceleration for a platoon of four vehicles following each other using a MPC control horizon of size 15 ($N_p = 15$), with a time headway of 0.2 s and a message rate of 10 Hz. It can be seen that the behaviour of the system is string stable regarding the acceleration profile but it is noisier than the emulation results. The velocity profile, the position error and the velocity error can be seen in Fig. 18b-d.

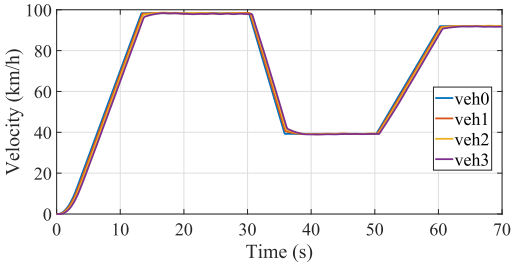
Due to the interference with other 5.9 GHz sources, we compare between Δd and Δv of the embedded implementation to Scenario-2 as seen in Fig. 18c and d. We see in Fig. 18c that Δd in the embedded implementation experiences a larger bounds than Scenario-2 in the Matlab simulation. In Fig. 18d, Δv for the embedded implementation shows similar performance to the Matlab simulation, however, with slightly more oscillatory behavior.

11. Conclusions

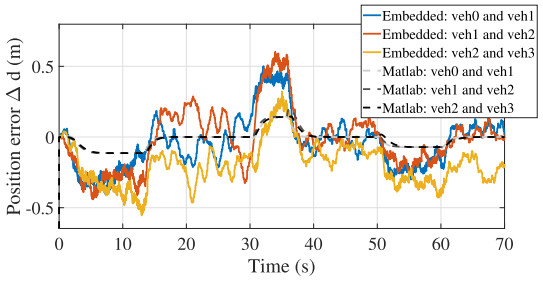
In this paper, we presented a multi-rate multi-layered vehicle platoon control that is compliant with both V2V and in-vehicle



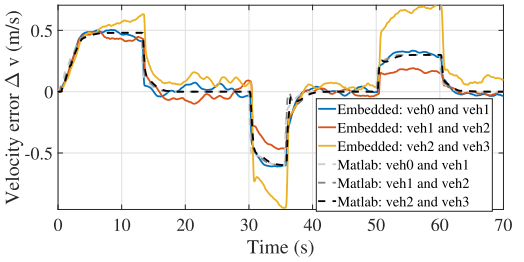
(a) Acceleration profile for a four vehicle platoon with 10Hz message rate, 0.2s time headway and a control horizon of size 15, embedded implementation.



(b) Velocity profile for a four vehicle platoon with 10Hz message rate, 0.2s time headway and a control horizon of size 15, embedded implementation.



(c) Comparison between position error for a four vehicle platoon with 10Hz message rate, 0.2s time headway and a control horizon of size 15, in an embedded implementation and Matlab implementation of Scenario-2.



(d) Comparison between velocity error for a four vehicle platoon with 10Hz message rate, 0.2s time headway and a control horizon of size 15 in an embedded implementation and Matlab implementation of Scenario-2.

Fig. 18. Embedded implementation results.

standards. We evaluated and validated the control scheme with extensive simulation of network and traffic behavior as well as on embedded communication units. Feasibility of the presented scheme is a step forward to the real deployment of such technology under various real-life constraints such as compliance with standards, protocols, equipment and cost. Our scheme achieves improved fuel efficiency and better usage of road capacity while ensuring safety. In future work, we plan to extend the approach to also operate under congested network conditions when DCC is applied to adjust message rate or other communication parameters, or when there is a need to switch to ACC. When communication delays increase, it may moreover be interesting to take into account the communication delay in the control design. Another interesting research direction is to adapt our proposed design approach for merging and splitting maneuvers of vehicles.

CRediT authorship contribution statement

Amr Ibrahim: Conceptualization, Methodology, Investigation, Validation, Writing - original draft. **Dip Goswami:** Conceptualization, Writing - review & editing, Supervision, Funding acquisition. **Hong Li:** Resources, Validation, Writing - review & editing. **Iñaki Martín Soroa:** Software, Investigation. **Twan Basten:** Writing - review & editing, Supervision, Funding acquisition.

Acknowledgements

This research has received funding from the European Union's Horizon 2020 Framework Programme for Research and Innovation under grant agreement no 674875 (oCPS).

References

Bergenhem, C., Huang, Q., Benmimoun, A., Robinson, T., 2010. Challenges of platooning on public motorways. In: 17th world congress on intelligent transport systems 2010 Oct 25, pp. 1–12.
Bergenhem, C., Shladover, S., Coelingh, E., Englund ,C., Tsugawa, S., 2012. Overview of platooning systems. In: Proceedings of the 19th ITS World Congress, Oct 22-26, Vienna, Austria (2012).
Bernardini, D., 2018. ODYS and GM bring online MPC to production!— ODYS. URL: http://www.odys.it/odys-and-gm-bring-online-mpc-to-production/.
Bleris, L.G., Vouzis, P.D., Arnold, M.G., Kothare, M.V., 2006. A co-processor FPGA platform for the implementation of real-time model predictive control. In: 2006 American Control Conference 2006 Jun 14. IEEE, pp. 6.
Borrelli, F., Bemporad, A., Morari, M., 2017. Predictive control for linear and hybrid systems. Cambridge University Press.
Corneliu, L., Alexandru, T., 2018. Control-Oriented Models for vehicle longitudinal motion. Tech. Sci. 3 (3), 251–264.
Deng, Q., 2016. A general simulation framework for modeling and analysis of heavy-duty vehicle platooning. IEEE Trans. Intell. Transp. Syst. 17 (11), 3252–3262.
Di Bernardo, M., Salvi, A., Santini, S., 2014. Distributed consensus strategy for platooning of vehicles in the presence of time-varying heterogeneous communication delays. IEEE Trans. Intell. Transp. Syst. 16 (1), 102–112.
Dolk, V.S., Ploeg, J., Heemels, W.M., 2017. Event-triggered control for string-stable vehicle platooning. IEEE Trans. Intell. Transp. Syst. 18 (12), 3486–3500.
Dunbar, W.B., Caveney, D.S., 2011. Distributed receding horizon control of vehicle platoons: Stability and string stability. IEEE Trans. Autom. Contr. 57 (3), 620–633.
Eben Li, S., Li, K., Wang, J., 2013. Economy-oriented vehicle adaptive cruise control with coordinating multiple objectives function. Veh. Syst. Dyn. 51 (1), 1–7.
ETSI, T., 2011. Intelligent transport systems (ITS); vehicular communications; basic set of applications; part 2: Specification of cooperative awareness basic service. Draft ETSI TS. 20, 448–451.
ETSI, T., 2012. STDMA Recommended Parameters and Settings for Cooperative ITS; Access Layer Part. ETSI ITS, Tech. Rep.
ETSI, T., 2013. Intelligent Transport Systems (ITS); V2X Applications; Part 1: Road Hazard Signalling (RHS) application requirements specification. 2013;101 539-1 V1.1.1.
ETSI, T., 2014. Intelligent Transport Systems (ITS); Users and applications requirements; Part 2: Applications and facilities layer common data dictionary. ETSI TS. 102, 894–892.
ETSI, T., 2014. Intelligent Transport Systems (ITS); Cross Layer DCC Management Entity for operation in the ITS G5A and ITS G5B medium; Report on Cross layer DCC algorithms and performance evaluation. ETSI TR 101 612; 2014 Sep.
ETSI, T., 2015. Cross Layer DCC Management Entity for operation in the ITS G5A and ITS G5B medium; Validation set-up and results, ETSI ITS, Tech. Rep.
Fiengo, G., Lui, D.G., Petrillo, A., Santini, S., Tufo, M., 2019. Distributed robust PID control for leader tracking in uncertain connected ground vehicles with V2V communication delay. IEEE/ASME Trans. Mechatron. 24 (3), 1153–1165.
Gao, F., Li, S.E., Zheng, Y., Kum, D., 2016. Robust control of heterogeneous vehicular platoon with uncertain dynamics and communication delay. IET Intell. Transp. Syst. 10 (7), 503–513.
Gelb, A. (Ed.), Applied optimal estimation. MIT Press.
Gong, J., Zhao, Y., Lu, Z., 2018. Sampled-data vehicular platoon control with communication delay. Proc. Inst. Mech. Eng. Part I: J. Syst. Control Eng. 232(1), 39–49.
Goswami, D., Masrur, A., Schneider, R., Xue, C.J., Chakraborty, S., 2013. Multirate controller design for resource-and schedule-constrained automotive ECUs. In: 2013 Design, Automation & Test in Europe Conference & Exhibition (DATE) 2013 Mar 18. IEEE, pp. 1123–1126.
Han, S.Y., Chen, Y.H., Wang, L., Abraham, A., 2013. Decentralized longitudinal tracking control for cooperative adaptive cruise control systems in a platoon. In: 2013 IEEE International Conference on Systems, Man, and Cybernetics 2013 Oct 13. IEEE.
Hussein, A.A., Rakha, H.A., 2020. Vehicle Platooning Impact on Drag Coefficients and Energy/Fuel Saving Implications. arXiv preprint arXiv:2001.00560. 2020 Jan 2.
Ibrahim, A., Math, C.B., Goswami, D., Basten, T., Li, H., 2018. Co-simulation framework for control, communication and traffic for vehicle platoons. In: 2018 21st Euromicro Conference on Digital System Design (DSD) 2018 Aug 29. IEEE, pp. 352–356.
Jiang, D., Delgrossi, L., 2008. IEEE 802.11 p: Towards an international standard for wireless access in vehicular environments. In: VTC Spring 2008-IEEE Vehicular Technology Conference 2008 May 11. IEEE, pp. 2036–2040.
Jin, I.G., Avedisov, S.S., He, C.R., Qin, W.B., Sadeghpour, M., Orosz, G., 2018. Experimental validation of connected automated vehicle design among human-driven vehicles. Transp. Res. Part C: Emerg. Technol. 1 (91), 335–352.
Kato, S., Tsugawa, S., Tokuda, K., Matsui, T., Fujii, H., 2002. Vehicle control algorithms for cooperative driving with automated vehicles and intervehicle communications. IEEE Trans. Intell. Transp. Syst. 3 (3), 155–161.
Kautsky, J., Nichols, N.K., Van Dooren, P., 1985. Robust pole assignment in linear state feedback. Int. J. Contr. 41 (5), 1129–1155.
Kianfar, R., Falcone, P., Fredriksson, J., 2014. A control matching-based predictive approach to string stable vehicle platooning. IFAC Proc. Vol. 47 (3), 10700–10705.
Kögel, M., Findeisen, R., 2011. A fast gradient method for embedded linear predictive control. IFAC Proc. Vol. 44 (1), 1362–1367.
Lee, D.N., 1976. A theory of visual control of braking based on information about time-to-collision. Perception 5 (4), 437–459.
Li, S., Li, K., Rajamani, R., Wang, J., 2010. Model predictive multi-objective vehicular adaptive cruise control. IEEE Trans. Control Syst. Technol. 19 (3), 556–566.
Li, S.E., Zheng, Y., Li, K., Wang, J., 2015. An overview of vehicular platoon control under the four-component framework. In: 2015 IEEE Intelligent Vehicles Symposium (IV) 2015 Jun 28. IEEE, pp. 286–291.

Limón, D., Alamo, T., Salas, F., Camacho, E.F., 2006. On the stability of constrained MPC without terminal constraint. IEEE Trans. Autom. Contr. 51 (5), 832–836.
Lin, W.Y., Li, M.W., Lan, K.C., Hsu, C.H., 2010. A Comparison of 802.11 a and 802.11 p for V-to-I Communication: A Measurement Study. In: International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness 2010 Nov 17. Springer, Berlin, Heidelberg, pp. 559-570.
Liu, X., Goldsmith, A., Mahal, S.S., Hedrick, J.K., 2001. Effects of communication delay on string stability in vehicle platoons. In: ITSC 2001. 2001 IEEE Intelligent Transportation Systems. Proceedings (Cat. No. 01TH8585) 2001 Aug 25. IEEE, pp. 625–630.
Liu, B., Jia, D., Lu, K., Ngoduy, D., Wang, J., Wu, L., 2017. A joint control-communication design for reliable vehicle platooning in hybrid traffic. IEEE Trans. Veh. Technol. 66 (10), 9394–9409.
Luo, L.H., Liu, H., Li, P., Wang, H., 2010. Model predictive control for adaptive cruise control with multi-objectives: comfort, fuel-economy, safety and car-following. J. Zhejiang Univ. Sci. A 11 (3), 191–201.
Maciejowski, J.M., 2002. Predictive control: with constraints. Pearson Education.
Makowitz, R., Temple, C., 2006. FlexRay-a communication network for automotive control systems. In: 2006 IEEE International Workshop on Factory Communication Systems 2006 Jun 27. IEEE, pp. 207–212.
Marsden, G., McDonald, M., Brackstone, M., 2001 1. Towards an understanding of adaptive cruise control. Transp. Res. Part C: Emerg. Technol. 9 (1), 33–51.
Maxim, A., Caruntu, C.F., Lazar, C., 2016. Distributed model predictive control algorithm for vehicle platooning. In: 2016 20th International Conference on System Theory, Control and Computing (ICSTCC) 2016 Oct 13. IEEE, pp. 657–662.
Milanés, V., Shladover, S.E., Spring, J., Nowakowski, C., Kawazoe, H., Nakamura, M., 2013. Cooperative adaptive cruise control in real traffic situations. IEEE Trans. Intell. Transp. Syst. 15 (1), 296–305.
Mizuochi, M., Tsuji, T., Ohnishi, K., 2007. Multirate sampling method for acceleration control system. IEEE Trans. Ind. Electron. 54 (3), 1462–1471.
Naus, G., Vugts, R., Ploeg, J., Van de Molengraft, M., Steinbuch, M., 2009. Towards on-the-road implementation of cooperative adaptive cruise control. In: Proc. 16th World Congr. Exhib. Intell. Transp. Syst. Serv. 2009 Sep 21.
Naus, G.J., Vugts, R.P., Ploeg, J., van De Molengraft, M.J., Steinbuch, M., 2010. String-stable CACC design and experimental validation: A frequency-domain approach. IEEE Trans. Veh. Technol. 59 (9), 4268–4279.
Öncü, S., Van de Wouw, N., Heemels, W.M., Nijmeijer, H., 2012. String stability of interconnected vehicles under communication constraints. In: 2012 IEEE 51st IEEE conference on decision and control (cdc) 2012 Dec 10. IEEE, pp. 2459–2464.
Öncü, S., Ploeg, J., Van de Wouw, N., Nijmeijer, H., 2014. Cooperative adaptive cruise control: Network-aware analysis of string stability. IEEE Trans. Intell. Transp. Syst. 15 (4), 1527–1537.
Orosz, G., 2016. Connected cruise control: modelling, delay effects, and nonlinear behaviour. Veh. Syst. Dyn. 54 (8), 1147–1176.
OSEK Consortium, 2005. OSEK/VDX operating system, version 2.2. 3, February 2005.
Ploeg, J., Scheepers, B.T., Van Nunen, E., Van de Wouw, N., Nijmeijer, H., 2011. Design and experimental evaluation of cooperative adaptive cruise control. In: 2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC) 2011 Oct 5. IEEE, pp. 260–265.
Ploeg, J., Semsar-Kazerooni, E., Lijster, G., van de Wouw, N., Nijmeijer, H., 2014. Graceful degradation of cooperative adaptive cruise control. IEEE Trans. Intell. Transp. Syst. 16 (1), 488–497.
Powell, J.P., Palacín, R., 2015. Passenger stability within moving railway vehicles: limits on maximum longitudinal acceleration. Urban Rail Transit. 1 (2), 95–103.
Qin, W.B., Gomez, M.M., Orosz, G., 2016. Stability and frequency response under stochastic communication delays with applications to connected cruise control design. IEEE Trans. Intell. Transp. Syst. 18 (2), 388–403.
Rajamani, R., 2011. Vehicle dynamics and control. Springer Science & Business Media.
SAE international, 2016. Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles. SAE International (J3016).
Sancar, F.E., Fidan, B., Huissoon, J.P., Waslander, S.L., 2014. MPC based collaborative adaptive cruise control with rear end collision avoidance. In: 2014 IEEE Intelligent Vehicles Symposium Proceedings 2014 Jun 8. IEEE, pp. 516–521.
Saxena, A., Li, H., Goswami, D., Math, C.B., 2016. Design and analysis of control strategies for vehicle platooning. In: 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC) 2016 Nov 1. IEEE, pp. 1805–1812.
Soroa, I.M., Ibrahim, A., Goswami, D., Li, H., 2019. Feasibility study and benchmarking of embedded MPC for vehicle platoons. In: st International Workshop on Autonomous Systems Design 2019 Mar 1.
Swaroop, D.V., 1997. String stability of interconnected systems: An application to platooning in automated highway systems.
Swaroop, D.V., Hedrick, J.K., Chien, C.C., Ioannou, P., 1994. A comparision of spacing and headway control laws for automatically controlled vehicles1. Veh. Syst. Dyn. 23 (1), 597–625.
Tiganasu, A., Lazar, C., Caruntu, C.F., 2017. Cyber Physical Systems-Oriented Design of Cooperative Control for Vehicle Platooning. In: 2017 21st International Conference on Control Systems and Computer Science (CSCS) 2017 May 29. IEEE, pp. 465–470.
Tsugawa, S., 2013. Final report on an automated truck platoon within energy ITS project. In: Presentation at the International Task Force on Vehicle-Highway Automation's 17th Annual Meeting 2013 Oct 13.
Tsujii, M., Takeuchi, H., Oda, K., Ohba, M., 1990. Application of self-tuning to automotive cruise control. In: 1990 American Control Conference 1990 May 23. IEEE, pp. 1843–1848.
Tuohy, S., Glavin, M., Hughes, C., Jones, E., Trivedi, M., Kilmartin, L., 2014. Intra-vehicle networks: A review. IEEE Trans. Intell. Transp. Syst. 16 (2), 534–545.
Turri, V., Besselink, B., Johansson, K.H., 2016. Cooperative look-ahead control for fuel-efficient and safe heavy-duty vehicle platooning. IEEE Trans. Control Syst. Technol. 25 (1), 12–28.
Ulsoy, A.G., Peng, H., Çakmakci, M., 2012. Automotive control systems. Cambridge University Press.
Van Arem, B., Van Driel, C.J., Visser, R., 2006. The impact of cooperative adaptive cruise control on traffic-flow characteristics. IEEE Trans. Intell. Transp. Syst. 7 (4), 429–436.
van de Sluis, J., Chen, L., Garcia-Sol, L., 2015. DEL_i-GAME_D3.2 proposal for extended message set for supervised automated driving. In: European Commission, Seventh Framework Programme. 2015 Sep.
van Nunen, E., Verhaegh, J., Silvas, E., Semsar-Kazerooni, E., van de Wouw, N., 2017. Robust model predictive cooperative adaptive cruise control subject to V2V impairments. In: 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC) 2017 Oct 16. IEEE, pp. 1–8.
Wang, P., Sun, Z., Tan, J., Huang, Z., Zhu, Q., Zhao, W., 2016. Development and evaluation of cooperative adaptive cruise controllers. In: 2015 IEEE International Conference on Mechatronics and Automation (ICMA) 2015 Aug 2. IEEE, pp. 1607–1612.
Weltzien, C., Diekhans, N., Harms, H.H., 1958. Distance Measurement Applications with Automotive Radar Sensors in Agricultural Environment. InCIGR World Congress 2006, 173–174.
Wills, A.G., Knagge, G., Ninness, B., 2011. Fast linear model predictive control via custom integrated circuit architecture. IEEE Trans. Control Syst. Technol. 20 (1), 59–71.
Xu, L., Wang, L.Y., Yin, G., Zhang, H., 2014. Communication information structures and contents for enhanced safety of highway vehicle platoons. IEEE Trans. Veh. Technol. 63 (9), 4206–4220.
Yanakiev, D., Kanellakopoulos, I., 1995. Variable time headway for string stability of automated heavy-duty vehicles. In: Proceedings of 1995 34th IEEE Conference on Decision and Control 1995 Dec 13, vol. 4. IEEE, pp. 4077–4081.
Zegers, J.C., Semsar-Kazerooni, E., Fusco, M., Ploeg, J., 2017. A multi-layer control approach to truck platooning: Platoon cohesion subject to dynamical limitations. In: 2017 5th IEEE International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS) 2017 Jun 26. IEEE, pp. 128–133.
Zeng, T., Semiariy, O., Saad, W., Bennis, M., 2019. Joint communication and control for wireless autonomous vehicular platoon systems. IEEE Trans. Commun.
Zhang, J., Ioannou, P.A., 2006. Longitudinal control of heavy trucks in mixed traffic: Environmental and fuel economy considerations. IEEE Trans. Intell. Transp. Syst. 7 (1), 92–104.
Zhang, R., Cai, L., Pan, J., 2017. MAC Protocols for High Data-Rate Wireless Networks. In: Resource Management for Multimedia Services in High Data Rate Wireless Networks 2017. Springer, New York, NY, pp. 9–42.

Zhao, R.C., Wong, P.K., Xie, Z.C., Zhao, J., 2017. Real-time weighted multi-objective model predictive controller for adaptive cruise control systems. Int. J. Automotive Technol. 18 (2), 279–292.

Zheng, Y., Li, S.E., Wang, J., Wang, L.Y., Li, K., 2014. Influence of information flow topology on closed-loop stability of vehicle platoon with rigid formation. In: 17th International IEEE Conference on Intelligent Transportation Systems (ITSC) 2014 Oct 8. IEEE, pp. 2094–2100.

Zheng, Y., Li, S.E., Li, K., Borrelli, F., Hedrick, J.K., 2016. Distributed model predictive control for heterogeneous vehicle platoons under unidirectional topologies. IEEE Trans. Control Syst. Technol. 25 (3), 899–910.

Zhou, H., Saigal, R., Dion, F., Yang, L., 2012. Vehicle platoon control in high-latency wireless communications environment: Model predictive control method. Transp. Res. Rec. 2324 (1), 81–90.

Zometa, P., Kögel, M., Faulwasser, T., Findeisen, R., 2012. Implementation aspects of model predictive control for embedded systems. In: 2012 American Control Conference (ACC) 2012 Jun 27. IEEE, pp. 1205–1210.