Jerk Beaconing: A Dynamic Approach to Platooning

Michele Segata*†, Falko Dressler*‡, Renato Lo Cigno†
*Computer and Communication Systems, Institute of Computer Science, University of Innsbruck, Austria
†Dept. of Information Engineering and Computer Science, University of Trento, Italy
‡Distributed Embedded Systems, Dept. of Computer Science, University of Paderborn, Germany
{segata,dressler}@ccs-labs.org,
locigno@disi.unitn.it

***Abstract*—Automated car following, or platooning, is a promising Inter-Vehicle Communication (IVC) application which has the potential of reducing traffic jams, improving safety, and decreasing fuel consumption by forming groups of vehicles which autonomously follow a common leader. The application works by sharing vehicles' data through high frequency periodic beaconing which, due to channel congestion, might not work in highly dense scenarios. To address this issue, in this paper we propose a dynamic approach called *Jerk Beaconing* which exploits vehicle dynamics to share data only when needed. The results, compared to a commonly assumed 10 Hz beaconing, show huge benefits in term of network resource saving. Moreover, our approach outperforms static beaconing in terms of safety as well, as it is able to keep inter-vehicle distance closer to the desired gap even in highly demanding scenarios.**

I. Introduction

Cooperative driving with information exchange via Inter-Vehicle Communication (IVC) is a promising concept to solve modern traffic jam problems and increase the safety of vehicles. The idea is that cooperative systems should make vehicles more and more autonomous, controlling them in a safe and efficient way, and should relieve humans from driving. Some steps have already been taken with the introduction of the Cruise Control (CC), or the Adaptive Cruise Control (ACC), control systems which are capable to maintain a desired speed and, with the help of a radar, automatically keep a safe distance from the front vehicle. An ACC system, however, is designed to keep a large safe gap, which is comparable to the one that must be kept by humans. The research community, particularly large projects such as PATH and SARTRE, thus started to develop and test an improved system: The Cooperative Adaptive Cruise Control (CACC) [1], [2].

A CACC uses information shared by other vehicles through IVC to reduce system's reaction time, enabling the possibility of minimizing the inter-vehicle gap and realize the so called *platooning* application. The vision is to form group of vehicles where only the leader is required to drive. The other vehicles in the platoon automatically accelerate, brake, and steer to follow their leader. The benefits of platooning range from improved road traffic throughput [3] to reduced fuel consumption thanks to lower air resistance [4]. Moreover, automated car following can improve safety and reduce driving stress.

The benefits of platooning, however, come at a cost. A CACC requires frequent data updates to operate safely. Some studies implementing this system in real cars suggests a beacon rate of 10 Hz [5]. Without using an intelligent data dissemination mechanism, such a high update rate cannot be sustained by a standard IEEE 802.11p radio interface in a dense scenario [6]. In a previous work we show that taking into account the specific requirements of the CACC in the design of a combined Time Division Multiple Access (TDMA) and Transmit Power Control (TPC) dissemination mechanism can provide the application with frequent updates even in heavily congested scenarios [6]. In this paper we make a further step to make platooning communications' footprint on the channel light exploiting cross layer information to reduce the required beacon rate when the platoon behavior is smooth and regular. We develop a dissemination protocol, called *Jerk Beaconing*, that exploits vehicle dynamics to determine when to send a beacon. In practice, if the state of a vehicle remains unchanged, the protocol does not send updates to its followers which can predict its current state by using previously received information. By sending data only when really needed, we drastically reduce the network load without compromising passengers' safety.

Our main contributions can be summarized as follows:

- We develop a protocol that uses vehicle's state information to dynamically adapt the beacon rate and send updates only when needed;
- We couple it with a reliable delivery mechanisms which ensures correct reception or notifies the application of the failure with a minimum protocol overhead;
- We test our protocol against a static 10 Hz beaconing in a large traffic jam scenario, showing the benefits of our approach in terms of safety and reduced network load.

II. Background and Related Work

In several years of platooning research, the community proposed different CACC algorithms [5], [7]–[9], which differ for their design and characteristics. The first design aspect is the *control topology*, i.e., the logical communication links that exist between vehicles in the same platoon. For instance, the CACC developed during the PATH project [7] exploits information received from the leader and the vehicle directly in front. The controller by Ploeg et. al. [5], instead, uses data from the front vehicle only. A further example is the CACC in [8], where the control topology is configurable and can be changed at runtime, with the possibility of having an all-to-all communication. The design of the control topology impacts on

978-1-4673-9411-6/15/$31.00 ©2015 IEEE

its characteristics: The most important in terms of application layer benefits is the spacing policy, i.e., the inter-vehicle gap required to ensure the stability of the platoon. The control topology used by the PATH CACC guarantees stability for a fixed gap, independent of the cruising speed, permitting to have inter-vehicle distances as small as a few meters [7]. For this reason we consider the PATH CACC for our analysis, as it is the one that can give the highest benefits in term of road throughput and fuel saving.

For the sake of completeness, we briefly describe the control formula. A more detailed description can be found in [7], [10]. The concept behind all CACC algorithms is simple: The controller takes inputs from sensors and on board radio interface, and computes a control input u (a desired acceleration) to be sent to the electronic control unit for actuation. In particular, PATH's CACC computes the control input u_i as follows:

$$u_i = \alpha_1 u_{i-1} + \alpha_2 u_0 + \alpha_3 \left(-d_{\text{radar}} + d_d\right) + \alpha_4 \left(\dot{x}_i - \dot{x}_0\right) + \alpha_5 \left(\dot{x}_i - \dot{x}_{i-1}\right), \quad (1)$$

where i is the index of the vehicle in the platoon (1 being the first follower), $\dot{x}_i$ is the speed, and d_{radar} and d_d are the actual and the desired distance respectively. The α_i parameters are controller gains that can be configured to change the behavior of the controller (see [10] for further details). The objective of the controller is to maintain a distance d_d between each vehicle in the platoon. The actual distance is measured by the radar, while desired acceleration and speed of leader and front vehicle are obtained through wireless communication.

As in [10], to mimic engine actuation delay effects, we use a first order low pass filter with a time constant τ, i.e., with the following transfer function:

$$\ddot{x}_i(s) = \frac{1}{1+\tau s} u_i, \quad (2)$$

where $\ddot{x}_i$ is the actual vehicle acceleration.

Given that the controller requires data of other vehicles, we need a radio interface and a communication protocol to share such information. Clearly, in a dense environment, this poses a challenge in terms of network congestion. This issue in IVC has been thoroughly investigated by the community since more than a decade. Still, researchers and standardization bodyies are trying to come up with new solutions that are capable to fulfill applications' requirements without congesting the network.

Initial works focused on congestion issues caused by the uncontrolled re-propagation of safety messages, i.e., the so called *broadcast storm* problem [11]. In that paper, three different solutions have been proposed, which exploit the distance from the sender to decide when and whether to re-broadcast a packet at all. In particular, a probabilistic, a slotted, and a combined approach have been considered: The farther the receiver from the sender, the higher the probability (or the lower the waiting time) for re-transmitting the message. The results, compared to a standard 1-persistent mechanism, show huge benefits in terms of packet reception rate and reduced network load. In [12], the authors further investigate the probabilistic scheme by considering packet aggregation, showing a network load reduction with no differences in terms of safety benefits.

The broadcast storm problem, however, only occurs in the case of event-triggered data packets (Decentralized Environmental Notification Messages (DENMs) in the ETSI ITS standard [13]) that require re-propagation, such as emergency braking messages. Another envisioned communication mechanism for vehicular safety applications is through periodic, single-hop beacons, known as Cooperative Awareness Messages (CAMs) or Basic Safety Messages (BSMs). The beacon frequency, however, highly depends on application requirements, and different applications might need to access the channel concurrently. For these reasons the community proposed to dynamically adjust the beacon rate and/or the transmit power to cope with network congestion [14]–[17].

In [14] the authors propose a distributed algorithm which dinamically adapts transmit power to keep under control the load caused by CAMs. Their approach solves an optimization problem to compute the power assignment for each vehicle that maximizes the transmit power under a maximum load constraint. The algorithm takes into account DENMs as well. Sommer et. al. [15], instead, develop an algorithm called Dynamic Beaconing (DynB) which maintains the channel load at a fixed, predefined value. Each vehicle periodically measures the channel load and decreases/increases its beacon rate if the load is higher/lower than the desired one. Similarly, in [17], the authors develop LIMERIC, a linear rate-control algorithm that makes the channel load to converge to a desired value. The algorithm is configurable by means of two parameters that control fairness, stability, and steady state convergence. The paper formally derives stability conditions and convergence time. ETSI, instead, developed the Decentralized Congestion Control (DCC) algorithm as a standard to be used for IVC in Europe [16]. The algorithm works by simultaneously adapting rate, transmit power, modulation, and Clear Channel Assessment (CCA) threshold. DCC periodically measures channel load but, differently from [15], [17], does not use a control theoretical approach. The algorithm is based on a state machine where each state uses a different beacon rate, transmit power, modulation, and CCA. The current active state is decided depending on the measured channel load. This approach, however, is unstable and might lead to oscillations [18]. ETSI also defined a dynamics-driven protocol [19] which, however, is not suited for a platooning application [6].

The discussed dynamic approaches are, in general, application unaware. They try to deliver enough data to all the applications by trying to avoid excessive channel load and thus packet losses. For some particular applications, however, this is not enough. In a previous work [6], we compare a periodic beaconing approach combined with a transmit power control rule against DynB and DCC for a platooning application. DynB and DCC are capable of keeping the channel load and packet losses under control, but at the expenses of a large packet inter-arrival time which is unacceptable for platooning.

For this reason, some works try to focus more on application rather than network layer performance. One example is the

work by Huang et. al. [20]. The authors develop an algorithm that uses a model predictive mechanism which computes a suspected tracking error and, upon that, derives a probability of transmission to decide whether to transmit or not at the next time step. Moreover, the algorithm dynamically changes the transmit power depending on the sensed channel load. In a high density scenario, the channel load increases: The algorithm thus decreases the transmit power to avoid congestion and to reach only nearby vehicles, which are the most important from a safety point of view. Conversely, when the network is sparse, the algorithm uses a high transmit power to reach farther vehicles. The approach, compared to a standard 10 Hz and 2 Hz beaconing, shows better tracking accuracy for nearby vehicles. A similar approach is presented by Bansal et. al. [21]. The authors couple LIMERIC [17] with a suspected tracking error as in [20]. Zemouri et. al. [22], instead, take a slightly different approach. They propose to first converge to a rate that limits channel busy ratio and collisions, and, if needed, to reduce the transmit power if a certain update requirement cannot be met. In a similar approach, Sepulcre et. al. [23] develop a protocol which considers both congestion and awareness for deciding when to send a beacon. In particular, each application can provide the protocol with an update requirement, and the protocol will consider it when computing the beacon interval. In another work [24], the authors propose a distributed optimization problem that adapts the probability of transmission in a slotted p-persistent approach using an utility function that considers both the expected delay and the safety benefit.

The approaches in [20]–[22], [24] improve application awareness in terms of safety, but are thought for standard CAM or BSM dissemination, which is undirected (broadcast) and does not consider that some messages might be more important than others. For the platooning application, messages from the leader are directed to all followers, and messages from the followers are directed to the vehicle immediately behind. In the next section, we thus exploit this to develop a vehicle dynamics based protocol coupled with a reliability mechanism that reduces network utilization while ensuring safety of a CACC application.

III. Jerk Beaconing

The idea of having a dynamic message rate for platooning is based on the observation that if the control quantities do not change, there is no gain in transmitting the same information over and over. As briefly mentioned in Section II, the CACC we consider uses acceleration and speed of leader and front vehicle, and the distance ahead to compute the acceleration to apply. If the acceleration of a vehicle $\ddot{x}$ is constant and we assume that such a vehicle sent its acceleration $\ddot{x}(t_s)$ and speed $\dot{x}(t_s)$ at time t_s, then the other vehicles can estimate its speed at time t by simply computing

$$\dot{x}(t) = \dot{x}(t_s) + \ddot{x}(t_s) \cdot (t - t_s). \tag{3}$$

From a purely theoretical point of view, each vehicle can then "feed" the controller with this information with no performance loss. In practice, however, we have two problems. The first one

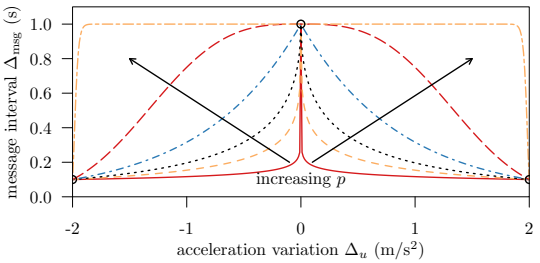


Figure 1: Δ_{msg} function for $\max_{\text{bi}} = 1\,\text{s}$, $\min_{\text{bi}} = 0.1\,\text{s}$, $\Delta_{u_{\max}} = 2\,\text{m/s}^2$, and different values of p (0.1, 0.3, 0.5, 1, 3, and 100).

is awareness, i.e., if we completely stop information sharing we cannot know if this is due to a network failure or to the fact that the state of a vehicle did not change. This can easily be solved by introducing a minimum beaconing rate. The second problem is that we need to decide what it means to have a constant acceleration. In the real world there exists no constant acceleration, and sending a beacon for every small variation is worse than using static beaconing. We thus need to find the bound that defines when acceleration changes are simply noise and can be neglected.

We approach the problem by using the concept of jerk. The jerk, denoted with $\dddot{x}$, is a physical quantity that measures the variation of acceleration over time. Exploiting a discrete estimation of the jerk, the beaconing controller decides whether to send a new beacon or not. The decision variable is the difference between the current value of the acceleration command computed by the controller and the value sent in the last message: $\Delta_u = u - u_{\text{sent}}$, with u and u_{sent} being the current value and the value sent in the last message respectively. To map this difference to a target beacon interval, we use the following formula:

$$\Delta_{\text{msg}}(\Delta_u) = \max\left(e^{-a|\Delta_u|^p} \cdot b, \min_{\text{bi}}\right). \tag{4}$$

Notice that we use the control input u (i.e., the desired acceleration) instead of the actual acceleration $\ddot{x}$ because u is what is sent in the beacons. The a, b, and p parameters change the behavior of the function. a and b control the maximum ($\max_{\text{bi}}$) and minimum ($\min_{\text{bi}}$) desired beacon interval. In particular, by setting

$$b = \max_{\text{bi}}, \quad a = -\ln\left(\frac{\min_{\text{bi}}}{\max_{\text{bi}}}\right) \cdot \Delta_{u_{\max}}^{-p} \tag{5}$$

we obtain $\Delta_{\text{msg}}(0) = \max_{\text{bi}}$ and $\Delta_{\text{msg}}(\Delta_{u_{\max}}) = \min_{\text{bi}}$. The p parameter controls the reactivity of the protocol. For example, for $p \to 0$, the minimum change in the desired acceleration causes the function to return the minimum beacon interval. On the contrary, for $p \to \infty$, the function returns the maximum beacon interval for any Δ_u such that $-\Delta_{u_{\max}} < \Delta_u < \Delta_{u_{\max}}$. To better understand the behavior of the function, refer to Figure 1.

The performance of the protocol, both from a network and a vehicular perspective, depends on the parameter p. A low value of p should improve the performance in term of safety but might overload the network. Conversely, by increasing p we save network resources, but vehicles may get too close or,

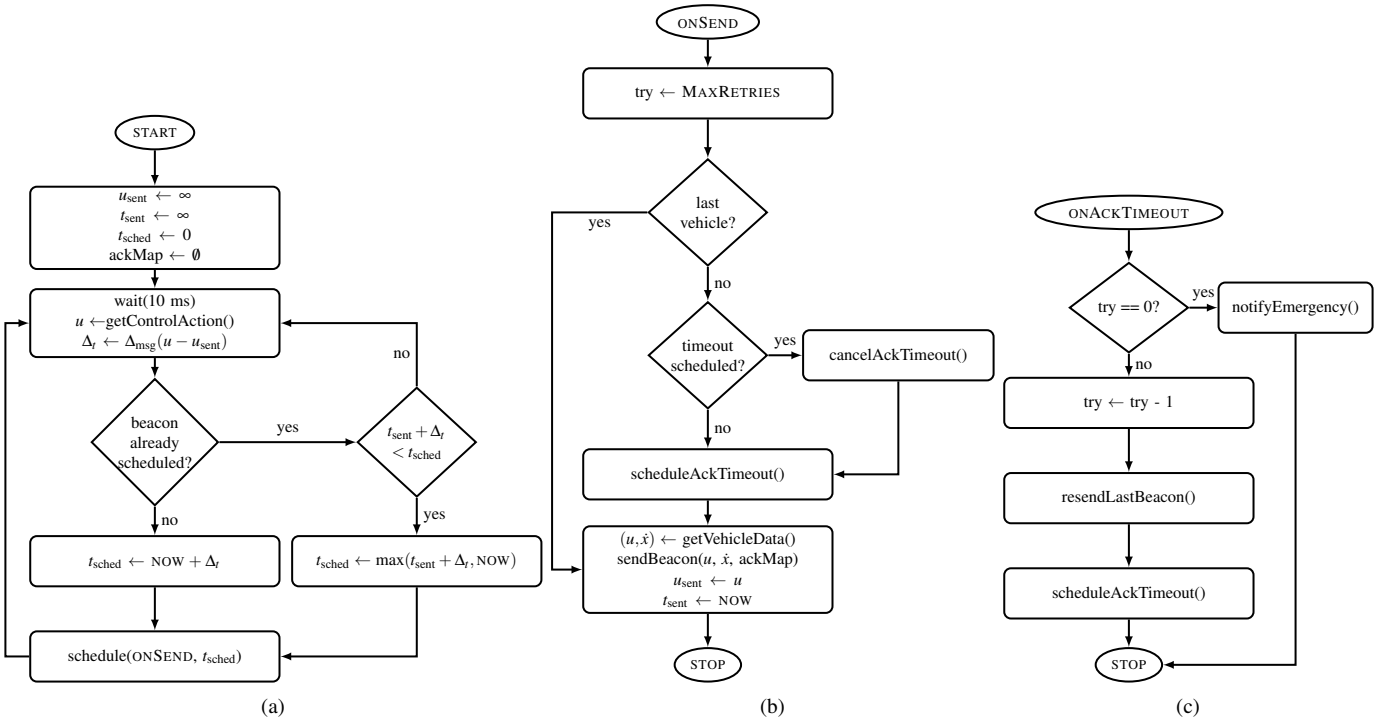


Figure 2: Flow charts for the main protocol loop, sending, and ack timeout procedures.

in the worst case, crash into each other. We thus need to find a good compromise that guarantees a minimum safety distance while using as few network resources as possible.

A. Lower Layer Reliability

We need to consider that, from a control theory point of view, the protocol tries to dynamically change the sampling frequency of the system to the minimum allowable. This means that lost packets can significantly harm system's safety. For this reason, we need to couple *Jerk Beaconing* with a lower layer protocol that ensures message delivery or informs the application that the network failed in doing so.

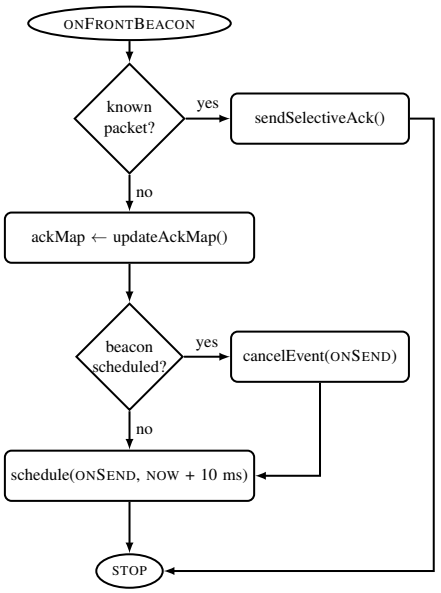
We start by exploiting the findings in our previous work [6]. In particular, we use a slotted approach which, besides reducing random channel contention among vehicles in the same platoon, permits us to use *chain acknowledgements*. In particular, assume that vehicle 0 sends a beacon scheduled by *Jerk Beaconing*. Upon reception of such message, vehicle's 1 CACC will compute a new control action which its follower will be interested in. Vehicle 1 can thus schedule a beacon which includes the updated information plus a piggybacked ack that vehicle 0 can overhear. This mechanism can be propagated towards the tail of the platoon in a chained fashion.

We further improve the protocol by adding an acknowledgement map. Each vehicle maintains a vector of values $\mathcal{A} = (a_1, \ldots, a_{N-1}) \in \mathbb{N}^{N-1}$, where a_i is the sequence number of the last packet of vehicle $i-1$ acknowledged by vehicle i. The map $\mathcal{A}$ is included in each beacon. This way, for example, vehicle 0 can get missed acknowledgements of vehicle 1 from vehicles further behind. Moreover, each vehicle includes the latest leader information received, because leader data is crucial for stability reasons [25]. To sum up, each beacon of vehicle i contains $(u_i, \dot{x}_i, s_i, \mathcal{A}_i, u_0, \dot{x}_0, s_0)$, with s_i being the sequence number associated to vehicle data. By considering 64-bit floats (for u and $\dot{x}$) and 64-bit integers (for sequence numbers), the MSDU size would be 200 B for a platoon of 20 cars. 32-bit numbers, however, have enough precision for the purpose, so with the same MSDU size we can support up to 45 vehicles.

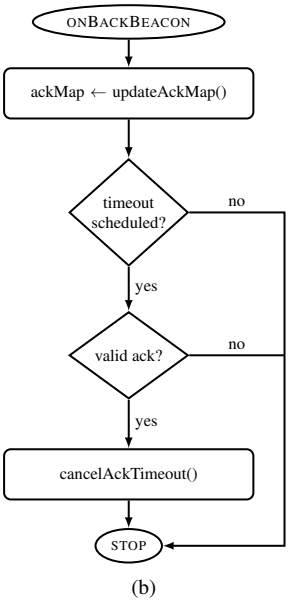
Finally, we exploit transmit power control for non-leading vehicles [6]. The followers use a reduced transmission power as they need to reach only the vehicle immediately behind.

Figures 2 and 3 show the flow diagrams of the protocol. Figure 2a shows the main loop of the protocol. Every 10 ms (the CACC sampling rate) the protocol invokes the Δ_{msg} function (Equation (4)) and decides whether to schedule a beacon to be sent or not. When a beacon needs to be sent (Figure 2b), the protocol sets the number of maximum retries and schedules an ack timeout. The last vehicle does not need to do so, as there is no platooning vehicle behind it. Still, other vehicles behind might be interested in receiving its data for cooperative awareness reasons.

In the case a vehicle does not receive an ack from vehicles behind within the timeout (Figure 2c), the number of tries is decremented and the beacon is re-sent. If the number of tries reaches zero, it means that the network is not properly working, and the protocol notifies the emergency to the application layer. Deciding what is the best action to take in such a situation is a non trivial task and we explicitly disregard this issue in the paper. In our evaluation, if a vehicle declares the emergency state we simply stop the simulation and log the outcome.



(a)



(b)

Figure 3: Flow charts for message reception handling.

Upon reception of a beacon from the vehicle directly in front (Figure 3a), we check whether this packet is already known. This might happen if a vehicle correctly receives a beacon, but the ack gets lost. In this case we simply send a selective ack. If the packet is new, we update the ack map and we schedule a new beacon to be sent in 10 ms. This beacon will also acknowledge the one we have just received.

Finally, when we receive a beacon from any of the vehicles behind (Figure 3b), we first update the ack map and then, if the packet acknowledges a previously sent beacon, we cancel the ack timeout.

IV. Evaluation

We test our proposal in a crowded traffic jam scenario implemented in the Plexe platooning simulation framework [10].

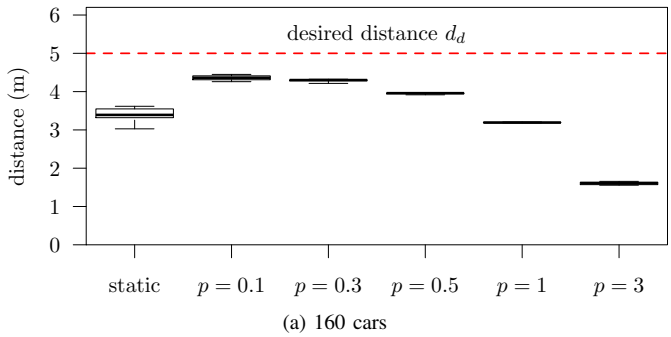
Table I: Network and road traffic simulation parameters.

	Parameter	Value
communication	Path loss model	Free space ($\alpha = 2.0$)
	Fading model	Nakagami ($m = 3$)
	PHY model	IEEE 802.11p
	MAC model	1609.4 single channel (CCH)
	Frequency	5.89 GHz
	Bitrate	6 Mbit/s (QPSK R = 1/2)
	Access category	AC_VI
	MSDU size	200 B
	Transmit power	20 dBm and 0 dBm
	$\max_{bi}$, $\min_{bi}$, $\Delta_{u_{max}}$	1 s, 0.01 s, 2 m/s^2
	p	0.1, 0.3, 0.5, 1, and 3
mobility	Max speed	130 km/h
	Min speed (harsh/gentle)	30 km/h and 110 km/h
	Deceleration (harsh/gentle)	7 m/s^2 and 3 m/s^2
	Acceleration	1.5 m/s^2
	Platoon size	20 cars
	Number of cars	160, 320, and 640
(C)ACC	Engine lag τ	0.5 s
	CACC's C_1, ω_n, ξ, d_d	0.5, 0.2 Hz, 1, 5 m
	ACC's T, λ	1.2 s, 0.1

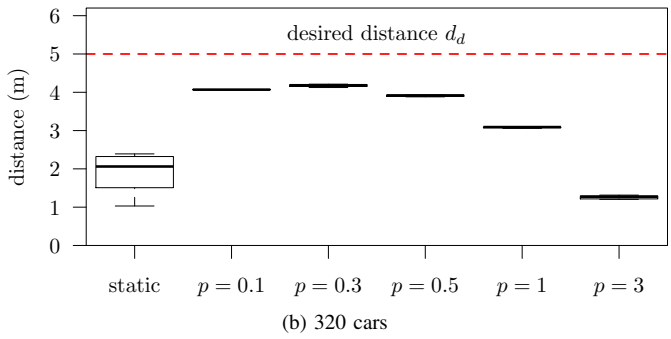
For coherence with our previous work [6], we setup a 4-lane freeway scenario with 160, 320, and 640 vehicles divided in platoons of 20 cars. At the head of each lane we add a vehicle which generates a traffic shockwave by continuously changing from a low to a high speed and vice versa every 30 s. Jamming vehicles are not perfectly synchronized, so platoons on different lanes are always close each other but they misalign over time. Platoon leaders are controlled by a standard ACC. We consider two jamming scenarios: A harsh and gentle one. In the harsh scenario, jamming vehicles switch from 130 km/h to 30 km/h and back with a deceleration of 7 m/s^2 and an acceleration of 1.5 m/s^2, simulating a very dangerous and demanding setup. In the gentle scenario, vehicles' speed changes between 130 km/h and 110 km/h with a deceleration of 3 m/s^2 and an acceleration of 1.5 m/s^2. Simulations run for 180 s (roughly three traffic jam cycles) and each configuration is repeated 10 times. Notice that even if the jamming vehicles have almost (due to actuation lag) "step-like" decelerations, the decelerations of the platoon leaders will be smoother, as they are computed by the ACC. Moreover, leaders at the back further attenuate the deceleration, thus the scenario considers inhomogeneous braking efforts.

We test our approach against periodic cooperative awareness beacons sent with a frequency of 10 Hz. *Jerk Beaconing* is tested for $\max_{bi} = 1$ s, $\min_{bi} = 0.01$ s, $\Delta_{u_{max}} = 2$ m/s^2, and for different values of p, i.e., 0.1, 0.3, 0.5, 1, and 3. The maximum number of retries and the ack timeout are set to 5 and 50 ms respectively. When using *Jerk Beaconing*, vehicles predict leader and front car speed using Equation (3). Finally, both static and *Jerk Beaconing* use transmit power control, i.e., followers use a transmit power of 0 dBm; Leaders, instead, use 20 dBm. Table I summarizes all simulation parameters.

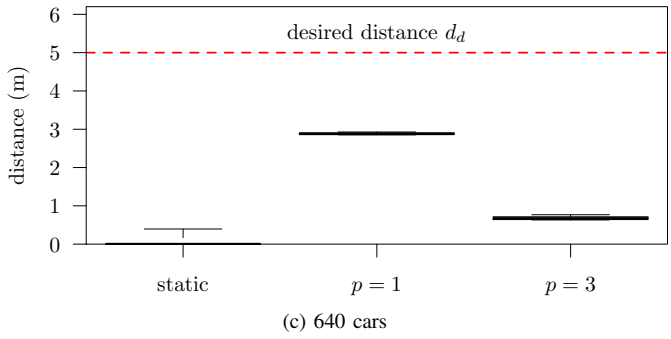
We evaluate the protocols using three different metrics. The first one is minimum distance: For all repetitions, we compute the minimum distance between any pair of vehicles in the simulation. This gives a measure of the safety of the protocol. The second metric is the channel busy ratio, i.e., we measure how much time the channel was sensed busy by all vehicles during the simulation, thus showing how much the network is



(a) 160 cars

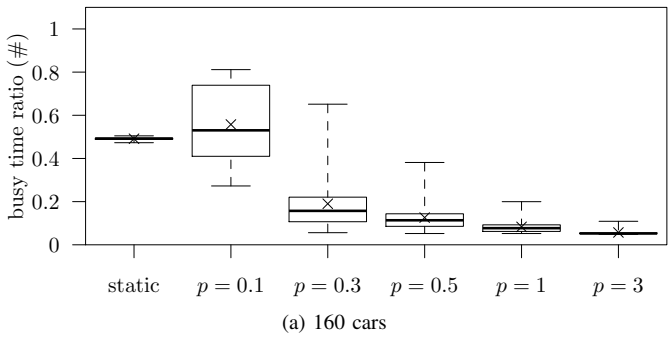


(b) 320 cars

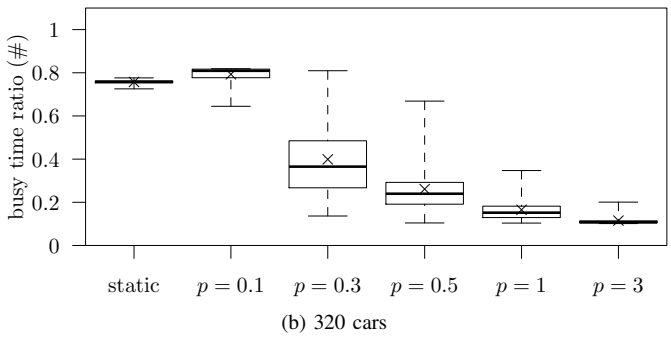


(c) 640 cars

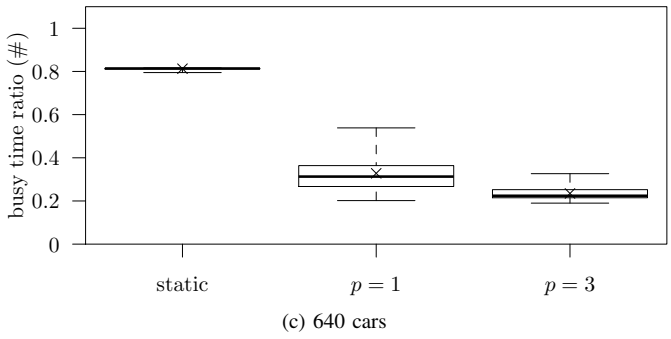
Figure 4: Minimum distances for the different protocols when transmit power control is not employed, harsh scenario.



(a) 160 cars



(b) 320 cars

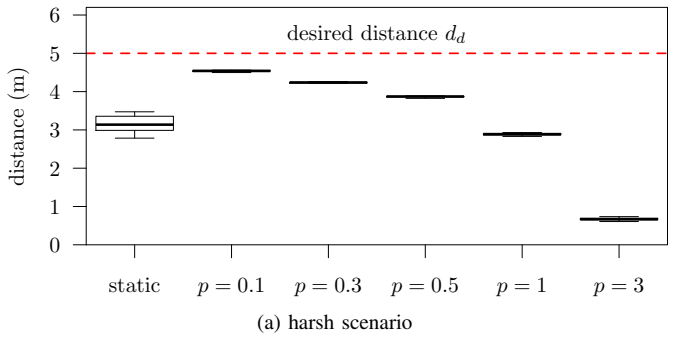


(c) 640 cars

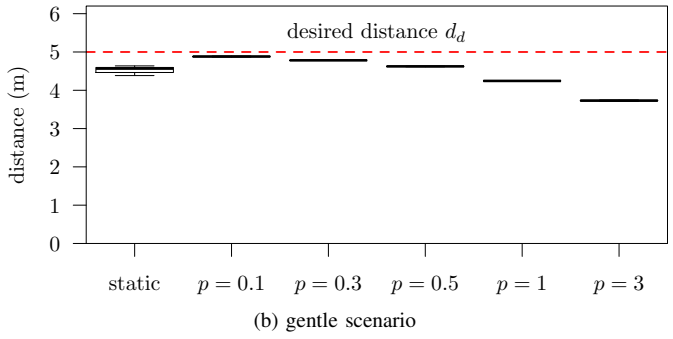
Figure 5: Channel busy time for the different protocols when transmit power control is not employed, harsh scenario.

overloaded. To compute this metric, each vehicle samples busy ratio with a 1 Hz frequency throughout the entire simulation: In the post-processing phase we take all samples from all cars and all repetitions and generate the boxplots. Finally, we compute the ECDF of beacon inter-arrival times, to show how many resources each protocol saves. This clearly needs to be read together with previous metrics, because a high beacon inter-arrival time might also indicate severe packet losses.

For the sake of completeness, we also evaluated the protocols when transmit power control is not employed (i.e., all vehicles use 20 dBm). Figure 4 shows the minimum distance results for the harsh scenario grouped in boxplots, together with a dashed line showing the desired distance d_d. The plots show a minimum distance strictly lower than d_d because it is reached in the braking phase, while d_d is the steady-state cruising distance. As highlighted in [6], platooning in large scenarios cannot be supported by static beaconing if transmit power is not adapted, and the simulations we performed further confirm this thesis. For the 160 cars scenario, the network still properly works. Neither static nor *Jerk Beaconing* suffer from a vehicle dynamics point of view. For 320 cars, static beaconing starts to suffer: Some simulations reach a minimum distance smaller than 2 m. *Jerk Beaconing* never results in collisions, but for $p = 0.1$, the majority of the simulations declared network failure due to ack timeout and were stopped. For higher values of p, *Jerk Beaconing* never led to a crash, and in particular, for $p = 0.5$, the minimum distance was always around 4 m. The issues can also be seen in Figure 5, were we plot the channel busy time. The network overload caused by static beaconing and by *Jerk Beaconing* for $p = 0.1$ is evident, and explains why the performance is unacceptable. The 640 vehicles scenario, by completely saturating the channel, makes all protocols unusable. Static beaconing almost always results in a vehicle collision. The packet loss rate caused by the network overload (Figure 5c) is non sustainable. *Jerk Beaconing* works only for $p = 1, 3$ but, from a vehicle-dynamics point of view, the results are not satisfying. For $p = 1$ the minimum distance is always around 3 m, while for $p = 3$ it goes down to less than 1 m. For $p = 3$, the minimum distance worsen with the number of cars in the scenario even if the network is not overloaded. This is due to

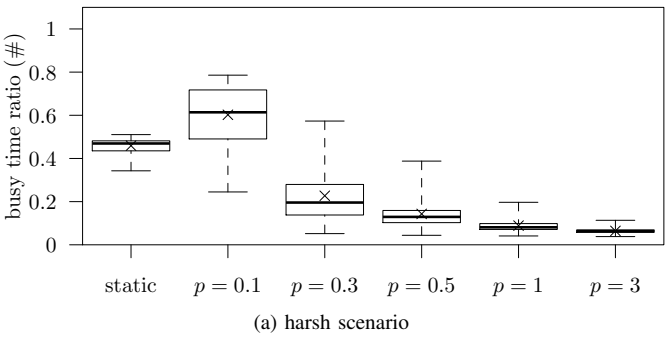


(a) harsh scenario

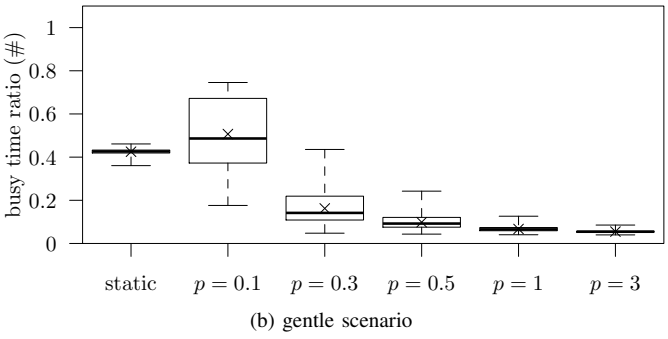


(b) gentle scenario

Figure 6: Minimum distances for the different protocols, 640 cars.



(a) harsh scenario



(b) gentle scenario

Figure 7: Channel busy time for the different protocols, 640 cars.

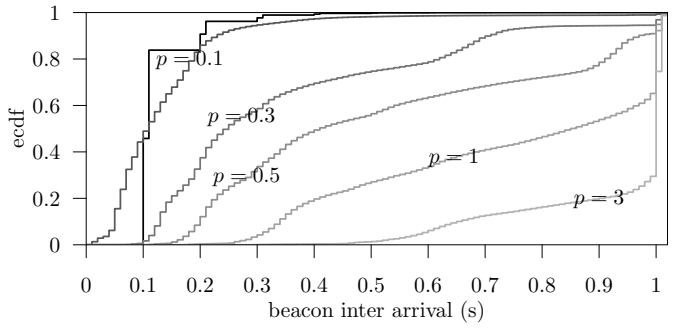
the high CCA-threshold which affects the computation of the channel busy metric, i.e., farther vehicles have few impact on the metric. Still, the interference caused by the high density of vehicles impacts packet reception rate which, in turn, highly affects *Jerk Beaconing* in a low reactivity setting.

When using TPC, independently from the number of vehicles (160, 320, and 640), for the same scenario (harsh or gentle) the results are almost equal. This is the case thanks to the use of TPC which reduces the interference range and makes the network capable of supporting very dense scenarios. For this reason we only report the results for 640 cars (Figure 6). Moreover, thanks to TPC coupled with the reliability protocol, the timeout mechanism declared network failure only in some rare cases. Figure 6a shows minimum distances for the harsh scenario. The plot confirms the benefits of using TPC: All protocols, including static beaconing, never cause an accident. *Jerk Beaconing* for values of p equal to 0.1, 0.3, and 0.5 shows better performance than static beaconing which, in turn, performs better than *Jerk Beaconing* with $p = 1$ and 3. In particular, for $p = 3$, *Jerk Beaconing* behaves too conservatively in terms of network resource usage, making the system unsafe. In the gentle scenario (Figure 6b), instead, the lower reactivity demand improves the performance of all approaches.

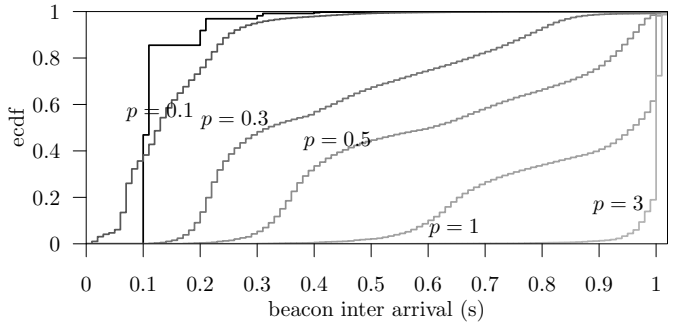
To compare the protocols from the point of view of network resources, Figure 7 shows the channel busy ratio statistics. The high performance in terms of minimum distance of *Jerk Beaconing* for $p = 0.1$ comes at a price, i.e., network overload. In this particular case, network usage is higher than for static beaconing. For other values of p, instead, *Jerk Beaconing* uses on average at most half of the resources, leaving the channel free for other possible applications. The results are qualitatively equal between harsh and gentle scenario, with the gentle scenario showing a slightly lower channel usage. This holds for static beaconing as well because of different vehicle densities in the scenario. The ACC used by the leaders results in an inter-platoon gap of only 10 m when driving at 30 km/h. Because the minimum speed in the gentle scenario is 110 km/h, instead, the inter-platoon distance is never lower than 36 m. This difference in density also affects the variance of the load, which is smaller in the gentle scenario.

As a final comparison metric, we consider beacon inter-arrival distribution for leader messages (Figure 8). The black, steep line represents static beaconing which, in the majority of the cases, delivers the packets every 100 ms, except in case of packet losses. Figure 8 also shows why *Jerk Beaconing* for $p = 0.1$ overloads the networks: The beacon inter-arrival time in the harsh scenario is smaller than 100 ms in 50 % of the cases, and 40 % of the cases in the gentle scenario. For other values of p, it is evident how effective *Jerk Beaconing* is in saving resources, both with respect to static beaconing and to different scenarios. Considering $p = 0.5$, in 50 % of the cases, *Jerk Beaconing* sends beacons no faster than every 400 ms in the harsh scenario, and 650 ms in the gentle scenario.

In conclusion, by observing Figures 6 to 8, *Jerk Beaconing* with $p = 0.5$ shows better performance in terms of both safety and resource saving compared to standard CAM or BSM beaconing. This shows the potential of dynamic approaches for platooning and poses a new interesting question: Can we find a theoretical link between vehicle dynamics, required sampling (i.e., beaconing) frequency, and performance in terms of minimum distance bound? This would help in designing an optimal networking protocol in terms of safety guarantees.



(a) harsh scenario



(b) gentle scenario

Figure 8: Empirical CDF of beacon inter-arrival times for the different protocols, 640 cars.

V. Conclusion

This work proposed *Jerk Beaconing*, a dynamic information dissemination protocol for platooning that exploits vehicle dynamics to send beacons only when needed. The protocol, coupled with a reliable delivery mechanism, showed improved performance both in terms of safety and network resource saving. This approach showed that it is not necessary to have a periodic 10 Hz beaconing to support platooning, but that by sending beacons at the right moment we can spare channel load and even improve passengers' safety. The results in this paper raise a further question: Can we find a theoretical link between beacon interval and performance of the controller? Answering this question would permit us to develop an optimal algorithm that could be tuned depending on the desired performance.

References

[1] S. Shladover, "PATH at 20 – History and Major Milestones," in *IEEE Intelligent Transportation Systems Conference (ITSC 2006)*, Toronto, Canada, September 2006, pp. 22–29.

[2] C. Bergenhem, Q. Huang, A. Benmimoun, and T. Robinson, "Challenges of Platooning on Public Motorways," in *17th World Congress on Intelligent Transport Systems (ITS 2010)*, Busan, Korea, October 2010.

[3] B. van Arem, C. van Driel, and R. Visser, "The Impact of Cooperative Adaptive Cruise Control on Traffic-Flow Characteristics," *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 4, pp. 429–436, December 2006.

[4] P. S. Jootel, "SAfe Road TRains for the Environment," SARTRE Project, Final Project Report, October 2012.

[5] J. Ploeg, B. Scheepers, E. van Nunen, N. van de Wouw, and H. Nijmeijer, "Design and Experimental Evaluation of Cooperative Adaptive Cruise Control," in *IEEE International Conference on Intelligent Transportation Systems (ITSC 2011)*, Washington, DC, October 2011, pp. 260–265.

[6] M. Segata, B. Bloessl, S. Joerer, C. Sommer, M. Gerla, R. Lo Cigno, and F. Dressler, "Towards Communication Strategies for Platooning: Simulative and Experimental Evaluation," *IEEE Transactions on Vehicular Technology*, 2015, in print, available online doi:10.1109/TVT.2015.2489459.

[7] R. Rajamani, H.-S. Tan, B. K. Law, and W.-B. Zhang, "Demonstration of Integrated Longitudinal and Lateral Control for the Operation of Automated Vehicles in Platoons," *IEEE Transactions on Control Systems Technology*, vol. 8, no. 4, pp. 695–708, July 2000.

[8] S. Santini, A. Salvi, A. S. Valente, A. Pescapè, M. Segata, and R. Lo Cigno, "A Consensus-based Approach for Platooning with Inter-Vehicular Communications," in *34th IEEE Conference on Computer Communications (INFOCOM 2015)*, Hong Kong, 2015, pp. 1158–1166.

[9] A. Ali, G. Garcia, and P. Martinet, "The Flatbed Platoon Towing Model for Safe and Dense Platooning on Highways," *Intelligent Transportation Systems Magazine, IEEE*, vol. 7, no. 1, pp. 58–68, January 2015.

[10] M. Segata, S. Joerer, B. Bloessl, C. Sommer, F. Dressler, and R. Lo Cigno, "PLEXE: A Platooning Extension for Veins," in *6th IEEE Vehicular Networking Conference (VNC 2014)*, Paderborn, Germany, December 2014, pp. 53–60.

[11] O. Tonguz, N. Wisitpongphan, J. Parikh, F. Bai, P. Mudalige, and V. Sadekar, "On the Broadcast Storm Problem in Ad hoc Wireless Networks," in *3rd International Conference on Broadband Communications, Networks, and Systems (BROADNETS)*, San Jose, CA, 2006.

[12] M. Segata and R. Lo Cigno, "Automatic Emergency Braking - Realistic Analysis of Car Dynamics and Network Performance," *IEEE Transactions on Vehicular Technology*, vol. 62, no. 9, pp. 4150–4161, October 2013.

[13] European Telecommunications Standards Institute, "Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 3: Specifications of Decentralized Environmental Notification Basic Service," ETSI, TS 102 637-3 V1.1.1, September 2010.

[14] M. Torrent-Moreno, J. Mittag, P. Santi, and H. Hartenstein, "Vehicle-to-Vehicle Communication: Fair Transmit Power Control for Safety-Critical Information," *IEEE Transactions on Vehicular Technology*, vol. 58, no. 7, pp. 3684–3703, September 2009.

[15] C. Sommer, S. Joerer, M. Segata, O. K. Tonguz, R. Lo Cigno, and F. Dressler, "How Shadowing Hurts Vehicular Communications and How Dynamic Beaconing Can Help," *IEEE Transactions on Mobile Computing*, vol. 14, no. 7, pp. 1411–1421, July 2015.

[16] European Telecommunications Standards Institute, "Intelligent Transport Systems (ITS); Decentralized Congestion Control Mechanisms for Intelligent Transport Systems operating in the 5 GHz range; Access layer part," ETSI, TS 102 687 V1.1.1, July 2011.

[17] G. Bansal, J. Kenney, and C. Rohrs, "LIMERIC: A Linear Adaptive Message Rate Algorithm for DSRC Congestion Control," *IEEE Transactions on Vehicular Technology*, vol. 62, no. 9, pp. 4182–4197, 2013.

[18] G. Bansal, B. Cheng, A. Rostami, K. Sjoberg, J. B. Kenney, and M. Gruteser, "Comparing LIMERIC and DCC Approaches for VANET Channel Congestion Control," in *6th IEEE International Symposium on Wireless Vehicular Communications (WiVec 2014)*, Vancouver, Canada, September 2014, pp. 1–7.

[19] ETSI, "Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 2: Specification of Cooperative Awareness Basic Service," ETSI, Tech. Rep. 302 637-2 V1.3.1, September 2014.

[20] C.-L. Huang, Y. P. Fallah, R. Sengupta, and H. Krishnan, "Adaptive Intervehicle Communication Control for Cooperative Safety Systems," *IEEE Network*, vol. 24, no. 1, pp. 6–13, January 2010.

[21] G. Bansal, H. Lu, J. B. Kenney, and C. Poellabauer, "EMBARC: Error Model based Adaptive Rate Control for Vehicle-to-Vehicle Communications," in *10th ACM International Workshop on Vehicular Internetworking (VANET 2013)*, Taipei, Taiwan, June 2013, pp. 41–50.

[22] S. Zemouri, S. Djahel, and J. Murphy, "Smart Adaptation of Beacons Transmission Rate and Power for Enhanced Vehicular Awareness in VANETs," in *17th IEEE International Conference on Intelligent Transportation Systems (ITSC 2014)*, Qingdao, China, October 2014, pp. 739–746.

[23] M. Sepulcre, J. Gozalvez, O. Altintas, and H. Kremo, "Adaptive Beaconing for Congestion and Awareness Control in Vehicular Networks," in *Vehicular Networking Conference (VNC), 2014 IEEE*, Paderborn, Germany, December 2014, pp. 81–88.

[24] L. Zhang and S. Valaee, "Safety Context-aware Congestion Control for Vehicular Broadcast Networks," in *15th IEEE International Symposium on Signal Processing Advances in Wireless Communications (SPAWC 2014)*, Toronto, Canada, June 2014, pp. 399–403.

[25] R. Rajamani, *Vehicle Dynamics and Control*, 2nd ed., 2012.