Improving the Jamming Resilience of a Cooperative Adaptive Cruise Controller

Mateen Malik$^{1,2[0000-0003-0148-533X]}$, Ludvig Ohlsson$^{2[0009-0000-2972-7949]}$, Karthik Sharma$^{2[0009-0008-7110-2211]}$, Behrooz Sangchoolie$^{1[0000-0001-9536-4269]}$, and Johan Karlsson$^{2[0000-0002-5542-2011]}$

[1] Dependable Transport Systems, RISE Research Institutes of Sweden, Sweden
`{mateen.malik,behrooz.sangchoolie}@ri.se`
[2] Chalmers University of Technology, Sweden
`{mateenma,ludohl,ravika,johan}@chalmers.se`

Abstract. This paper presents the results of extensive simulations assessing the impact of barrage jamming attacks on various Cooperative Adaptive Cruise Control (CACC) algorithms. CACC is an extension to Adaptive Cruise Control (ACC) using Vehicle-to-Vehicle (V2V) communication to maintain inter-vehicle spacing and achieve string stability, e.g., in automotive platooning systems. To ensure safety, CACC algorithms must be resilient to jamming attacks and other types of communication failures. In this study, we assess an existing CACC algorithm and extend it with two fallback mechanisms in order to improve its resilience to jamming attacks. Our simulations show that both proposed mechanisms substantially improve jamming resilience, with one being slightly more effective than the other. To allow a comparison with other CACC algorithms, we also conduct jamming attack simulations with three other previously published CACC algorithms. Our results confirm that CACC algorithms primarily relying on local sensor data exhibit high resilience to jamming attacks at the cost of increased inter-vehicle spacing. In contrast, CACC algorithms that depend on external sensor data achieve smaller inter-vehicle spacing but are significantly more vulnerable to jamming attacks. However, this vulnerability can be mitigated by carefully designed fallback mechanisms.

Keywords: Simulation-based testing · Platooning · Jamming attacks · Jamming resilience · Cooperative Adaptive Cruise Control (CACC).

1 Introduction

In an era of rapid advancements in vehicular technology, the concept of Cooperative Driving Automation (CDA) has gained significant attention as a transformative approach to modern transportation. CDA relies on Vehicle-to-Vehicle (V2V) wireless communication for an exchange of parameters such as acceleration, speed, direction and location between vehicles. While the exchange of sensor data allows for the implementation of advanced autonomous or semi-autonomous driver support systems, it also makes these systems potentially vulnerable to communication failures and jamming attacks.

This paper presents the results of extensive simulations conducted to investigate the impact of jamming attacks on six cooperative adaptive cruise control (CACC) algorithms designed to provide longitudinal control of vehicles in platooning applications. The objective of these algorithms is to maintain short inter-vehicle distances while enforcing string stability [1]. To conduct the simulations, we have used a simulation engine known as ComFASE[2], which automates jamming attack simulations conducted with the Plexe simulation environment [3].

The goal of this paper is to present and evaluate two extensions of an existing CACC algorithm, which we denote as P1 (Plexe 1) [3]. P1 is one of several CACC algorithms available in the Plexe simulations. The control laws of P1, which use a *constant spacing policy* for longitudinal control, have been developed by Rajamani et al. [4]. The extensions we develop is to enhance the P1's jamming resilience by incorporating fallback mechanisms that leverages vehicles' onboard radars and using Adaptive Cruise Control (ACC) algorithm to maintain safe spacing between the vehicles, when communication failures occur. These modifications significantly strengthen the P1, ensuring greater string stability and safety under jamming attacks. To enable a comparative analysis, we also conducted jamming attack simulations on three other CACC algorithms available in Plexe.

Previous research by van der Heijden et al. [5] has shown that the P1 algorithm is highly sensitive to jamming attacks. There are two reasons for this: *(i)* the implementation of the constant spacing policy is highly dependent on sensor information received via V2V communication, and *(ii)* the P1 algorithm is not equipped with any fallback mechanisms that can mitigate the impact of communication failures. In their paper, van der Heijden et al. conducted jamming attack simulations not only with P1 but also with two other CACC algorithms available in Plexe called Ploeg [6] and Consensus [7]. The results showed that Ploeg and Consensus were much more resilient to jamming attacks compared to P1. However, the Ploeg and Consensus implement a *variable spacing policy* that largely depends on the vehicle's own sensors and less so on sensor data from other vehicles, which makes them intrinsically resilient to communication failures.

According to Ali et al., [8], constant spacing algorithms can maintain string stability at higher traffic flows than their variable spacing counterparts. Motivated by this observation, this paper aims to address the following research

[1] String stability implies that changes in the speed of the platoon leader should not result in unbounded variations in the inter-vehicle distances as such speed changes propagate towards the end of the platoon.

[2] ComFASE [1] is developed within our research group and is available for download at [2].

[3] Segata et al. [3] call this algorithm CACC. We call it P1 to avoid confusion with the general meaning of the acronym CACC.

question: *Can we design a fallback mechanism that makes P1 - a constant spacing algorithm - highly resilient to jamming attacks?*

Providing a general answer to this research question is, however, a challenging task given that jamming attacks can be conducted with a variety of jamming techniques ranging from barrage jamming to sophisticated destructive interference attacks. The communication jamming taxonomy presented by Lichtman et al. [9] provides a good overview of different jamming techniques. In this paper, we focus our simulations using barrage jamming attacks.

Barrage jamming attacks introduce high-powered noise-like energy over the entire portion of the frequency spectrum used by the target signal. These attacks are relatively easy to execute as they require minimal knowledge of the target system and the communication protocol. Lichtman et al. [9] therefore classify them as non-protocol aware and non-time correlated attacks in their jamming taxonomy. We model these attacks by modifying the Signal to Interference & Noise Ratio (SINR) parameter, which is available in the physical layer simulator for the IEEE 802.11p protocol in Plexe.

In summary, the paper makes the following contributions:

1. A comprehensive study of the impact of barrage jamming attacks on a platooning system using the P1 algorithm without any fallback mechanisms.
2. Design and evaluation of two CACC algorithms, which extends P1 improving its resilience to barrage jamming attacks. of P1 offering strong resilience to barrage jamming attacks.
3. Barrage jamming attack simulations with three other CACC algorithms available in Plexe for comparative analysis.

The remainder of the paper is organized as follows. We present related work in **Section 2**. In **Section 3**, we describe the implementation of the P1 extensions: P1A and P1B. **Section 4** presents the experimental setup. **Section 5** presents the experimental results and evaluation. Finally, **Section 6** has the summary, conclusions, and future work.

2 Related Work

2.1 CACC Algorithms

Rajamani et al. [4] established the foundational equations in the design of the P1 algorithm implemented by Segata et al. [3]. P1 employs a constant spacing policy to maintain the inter-vehicle spacing, achieving string stability by relying on inter-vehicle communication. However, it assumes ideal communication conditions as it has no fallback mechanisms to cope with communication failures.

Santini et al. [7] propose a CACC algorithm called Consensus. Consensus eliminates the need for a designated leader vehicle. Instead, each vehicle receives data from all others and performs a verification step before using the information to control its speed and acceleration. While this consensus-based model inherently resists many types of communication failures, the authors note that its reliability deteriorates beyond a 60% Packet Error Rate (PER).

Ploeg et al. [6] introduce a CACC algorithm called Ploeg that ensures string stability while incorporating a fallback mechanism. Their system allows vehicles to switch from CACC to Adaptive Cruise Control (ACC) using a custom hardware gateway called MOVE, eliminating reliance on wireless communication.

The Consensus and Ploeg models use a variable spacing policy for inter-vehicle distance management. Unlike constant spacing policies, this approach reduces the need for extensive inter-vehicle communication. Instead, these models achieve string stability using onboard sensor data, with the drawback that inter-vehicle distances vary with speed and can become quite large. The P1B algorithm we developed has fallback mechanisms similar to the one proposed by Ploeg et al. However, the P1B automatically switches to ACC upon experiencing poor communication.

Ali et al. [8] present the Flatbed CACC model, which conceptualizes a stationary vehicle placed on a moving flatbed truck traveling at velocity V. Each vehicle adjusts its speed relative to the truck, allowing for closer inter-vehicle spacing while maintaining platoon stability.

Part of our study is closely related to van der Heijden et al. [5], where the authors evaluate the jamming and cyberattack resilience [9] of three CACC algorithms (P1, Consensus, and Ploeg) implemented in Plexe. van der Heijden et al. [5] look into the impact of denial-of-service (DoS) attacks on CACC algorithms and compare them side by side with respect to collisions and spacing errors. Moreovoer, they did not propose any fallback mechanisms for P1.

2.2 Jamming Techniques

Lichtman et al. [10] categorize jamming attacks based on the jammer's capabilities, which include protocol awareness, time correlation, learning ability, and signal spoofing. Protocol awareness refers to the jammer's knowledge of the communication protocol. Time correlation indicates how well the attacker's signals are synchronized with legitimate signals. Learning ability allows a jammer to adapt by analyzing communication patterns over time. Lastly, signal spoofing involves faking or altering legitimate signals to mislead receivers.

Barrage jamming is the simplest form of jamming attacks that is usually defined as the jammer continuously transmitting noise energy across the entire portion of the frequency spectrum [9]. Barrage jamming is non-protocol-aware and not time-correlated. In contrast, pilot jamming and nulling precisely target synchronization of attacker's and legitimate signals in wireless networks, which make them more efficient while requiring knowledge of the protocols involved. Repeater jamming is time-correlated, where the jammer listens to the target's signal and transmits a modified version to disrupt the communication.

Previous research has investigated various jamming techniques that can interfere with wireless communication by partially degrading signals or completely blocking transmission. Moser et al. [11] developed a jamming method to simulate the effects of interference and signal nulling on the Cyclic Prefix (CP), a technique used to enhance signal integrity by duplicating the end portion of a signal at its beginning. These attacks aim to weaken or eliminate legitimate signals, effectively causing a denial of service.

3 Extentions of the P1 Control Algorithm

This section provides a detailed overview of the extensions of the P1 algorithm: P1A and P1B. Before going into the details of these extensions in Sections 3.2 and 3.3, we present the P1 algorithm in Section 3.1. We chose P1 as our baseline CACC algorithm for evaluating cooperative driving behaviors under barrage jamming attacks. This CACC algorithm has no fallback mechanism, thus to enhance its resilience against barrage jamming attacks, we introduce P1A and P1B as extensions of P1.

3.1 P1 Control Algorithm

P1 is the default CACC algorithm available in the Plexe framework. P1 follows a constant spacing policy, which relies heavily on inter-vehicle communication to maintain a fixed distance between vehicles. The desired spacing value can be configured before simulations, allowing flexibility in different testing scenarios. The P1 algorithm consists of an upper-level controller and a lower-level controller. The upper-level controller calculates the desired acceleration for each vehicle to maintain the predefined spacing, while the lower-level controller converts this acceleration into throttle and brake commands.

In the P1 algorithm, each vehicle's upper-level controller receives real-time data from the lead and preceding vehicles through the wireless network. This data includes the controller's desired acceleration (m/s^2), the vehicle's actual acceleration (m/s^2), speed (m/s), position (m), and the timestamp in seconds (s) indicating when the measurements were taken. The controller processes this information to compute the desired acceleration for each vehicle using Eq. 1 [3].

$$\begin{aligned} &\ddot{x}_{i\_des} = \alpha_1 \ddot{x}_{i-1} + \alpha_2 \ddot{x}_0 + \alpha_3 \dot{\varepsilon}_i + \alpha_4 (\dot{x}_i - \dot{x}_0) + \alpha_5 \varepsilon_i \\ &\text{where,} \\ &\dot{\varepsilon}_i = \dot{x}_i - \dot{x}_{i-1} \\ &\varepsilon_i = x_i - x_{i-1} + l_{i-1} + \text{gap}_{\text{des}} \end{aligned} \tag{1}$$

In this equation, $\ddot{x}_{i\_des}$ is the desired acceleration. $\ddot{x}_0$ and $\dot{x}_0$ are the acceleration and speed of the leader, respectively, while $\ddot{x}_{i-1}$ is the acceleration of the preceding vehicle. The distance error ε_i is based on a constant desired distance [3], while gap_{des} is the desired spacing gap between the vehicles. The length of the preceding vehicle is denoted by l_{i-1}, x_i is the position of the ith vehicle, and finally, x_{i-1} is the position of the preceding vehicle. The alpha terms in Eq. 1 are defined in Eq. 2 [3].

$$\begin{aligned}
&\alpha_1 = 1 - C_1, \alpha_2 = C_1, \alpha_5 = -\omega_n^2 \\
&\alpha_3 = -(2\xi - C_1(\xi + \sqrt{\xi^2 - 1}))\omega_n \\
&\alpha_4 = -C_1(\xi + \sqrt{\xi^2 - 1})\omega_n
\end{aligned} \tag{2}$$

In Eq. 2, C_1 is a weight factor between the accelerations of the leader and the preceding vehicles (default set to 0.5), ξ is the damping ratio (default set to 1), and ω_n is the bandwidth of the controller (default set to 0.2 Hz). Except from the constant alpha factors, the first four variables in Eq. 1, $\ddot{x}_{i-1}$, $\ddot{x}_0$, $\dot{\varepsilon}_i$ and $(\dot{x}_i - \dot{x}_0)$ mainly depend on the information received from the V2V communication.

The gap error ϵ_i (see Eq. 1) relies mostly on sensor data obtained from each vehicle's onboard radar, making it immune to communication losses. In P1, the radar is primarily used to maintain a constant spacing between vehicles (`gap2pred`) that is based on the 'position of the ith vehicle and preceding vehicle' and the desired spacing (gap_{des}) as shown in Eq. 1. The radar also calculates the preceding vehicle's speed (`predSpeed`). However, unlike standard ACC systems, P1 takes the speed value from the communication network and does not utilize the radar speed for collision avoidance. This design choice of P1 allows us to assess the impact of communication failures on string stability without interference from collision avoidance mechanisms.

3.2 P1A Control Algorithm

In the P1A algorithm, the controller switches to a degraded mode of operation (degCACC) when a communication failure is detected.

In case of complete communication failures, the detection is based on the calculated time difference between the current time and the time when the latest message from the front vehicle was received. For every vehicle in the platoon, P1A operates by regularly comparing the current time with the timestamp of the most recently received network packet. If the time difference is smaller than 100 ms, representing the beaconing interval at which the leader vehicle transmits information, the vehicle assumes that the communication link to other vehicles is still up and continues to use the P1 algorithm. However, if the time difference is larger than or equal to 100 ms, signaling a potential communication loss, the vehicle switches to a fallback mechanism.

In case of partial communication failure, P1A checks every 100 ms if a message has been received within that time frame from the front or the leader vehicle. The fallback mechanism activates if either of the messages is lost. In both cases, the transition occurs only for vehicles that experience communication failures while the rest of the platoon continues to operate under the P1 algorithm. When a message loss is detected, the system transitions [3] to retrieving the value

[3] We set the fallback trigger time to 100 ms, matching the beaconing interval at which the leader vehicle updates and transmits information.

of `predSpeed` from onboard radar instead of relying on external sensor data received from wireless communication. Additionally, the desired inter-vehicle spacing, $\mathrm{gap}_{\mathrm{des}}$, is increased tenfold. These modifications significantly improve the algorithm's ability to maintain string stability despite communication failures.

In summary, P1A can be triggered on complete or partial communication failures[4]. The implementation specifically affects two key terms in Eq. 1. First, the $\dot{\epsilon}_i$ is modified to rely solely on speed data obtained from radar. Second, the $\mathrm{gap}_{\mathrm{des}}$ is scaled up by a factor of 10, ensuring a longer vehicle spacing only for the time period in which the communication is disrupted.

3.3 P1B Control Algorithm

In the P1B algorithm, the resilience of P1 is enhanced by integrating a fallback mechanism that switches to the standard Adaptive Cruise Control (ACC) algorithm upon detection of a communication failure. The switching mechanism is the same as P1A (see Section 3.2).

Vehicles equipped with ACC automatically decelerate to maintain a safe distance from a preceding vehicle driving at a lower speed. This ACC implementation is available in SUMO. In our approach, the switchover from P1 to ACC occurs only for vehicles experiencing communication loss while the rest of the platoon continues operating under P1. Once communication is restored, the affected vehicles seamlessly switch back to P1. The ACC determines the desired acceleration $\ddot{x}_{i\_des}$ for vehicle i by using Eq. 3 [3].

$$\begin{aligned} \ddot{x}_{i\_des} &= -\frac{1}{T}(\dot{\epsilon}_i + \lambda\delta_i) \\ where& \\ \delta_i &= x_i - x_{i-1} + l_{i-1} + T\dot{x}_i \\ and& \\ \dot{\epsilon}_i &= \dot{x}_i - \dot{x}_{i-1} \end{aligned} \tag{3}$$

In this equation, i represents the controlled vehicle, while T denotes the time headway in seconds. The term $\dot{\epsilon}_i$ captures the speed difference between the controlled vehicle and its preceding vehicle, which helps determine whether acceleration or deceleration is needed. The variable δ_i represents the deviation between the current distance to the preceding vehicle and the desired following distance. Lastly, λ is a design parameter that is always greater than zero (default value set to 0.1) [3].

4 Experimental Setup

4.1 Driving Scenario

We define a driving scenario as the specific driving pattern of the vehicles, such as acceleration and deceleration. The Plexe simulation framework provides various

[4] P1A, which triggers on partial communication failures, can handle both partial and complete communication failures.

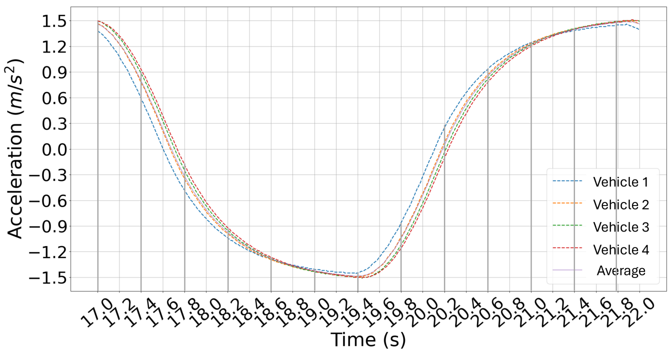


Fig. 1: A sample simulation period showing acceleration profiles of all vehicles in the platoon.

predefined platooning driving scenarios, including *sinusoidal scenario*, *braking scenario*, *platoon merge*, and *platoon split*. Our simulations use the *sinusoidal scenario* with a total simulation time of 45 s. In this driving scenario, the platoon drives in a sinusoidal driving pattern (see Fig. 1), where vehicles accelerate and decelerate at a fixed frequency of 0.2 Hz and an amplitude of 5.0 km/h. The lead vehicle's maximum speed is 100 km/h, and the inter-vehicle distance is kept to 5 meters. We chose this driving scenario as it effectively captures dynamic highway driving scenarios, including acceleration and deceleration phases, which are critical for evaluating the impact of barrage jamming attacks on CACC algorithms.

4.2 Attack Model

In our simulations, we use the *Barrage jamming* attack model from the ComFASE attack model library to evaluate the CACC algorithms. Lichtman et al. [9] classify *barrage jamming* as a *non–protocol-aware* attack since it does neither require prior knowledge of the communication protocol nor preciseness to be effective. In ComFASE, *barrage jamming* is modeled by modifying the noise power parameter used for SINR calculations in the Veins simulator. This parameter accounts for various noise sources, such as channel interference and noise.

The *barrage jamming* model injects equal noise across all vehicles simultaneously. However, its impact varies due to differences in legitimate signal strength, which depend on factors such as the distance between vehicles and interference levels. These variations can lead to symmetric or asymmetric message losses. Symmetric losses affect all vehicles equally, whereas asymmetric losses result in uneven message losses based on distance and noise intensity. Moreover, barrage jamming attacks often serve a more sinister purpose than just a denial of service. For example, a barrage jamming attack against an Advanced Driver Assistance System (ADAS) or CDA application, such as platooning, can cause vehicle collisions and traffic jams.

Table 1: Outcome categories defined to classify the simulation results.

Severe	Emergency braking activation (deceleration of higher than 5.0 m/s^2) or collisions recorded.
Benign	None of the vehicles perform an emergency braking, but at least one brakes with a rate higher than the highest deceleration recorded in the golden run corresponding to 1.53 m/s^2.
Negligible	The recorded maximum deceleration is less than or equal to 1.53 m/s^2.
Non-effective	The vehicles drive exactly as they did in the golden run.

Attack Model Parameters. In ComFASE, the *barrage jamming* attack model includes three key parameters: *attack start time*, *attack duration*, and *attack value*. The attack-start time specifies when the attack is initiated in the driving scenario. The attack duration determines how long the attack remains active. Finally, the attack value represents the severity of the attack by defining the amount of noise injected into the communication channel.

Attack Injection Configuration. To configure the attack injection campaigns, we vary the attack value from 0.04 to 1×10^{-5} mW, in steps of 0.04×10^{-5} mW, resulting in 25 experiments. We selected 13 attack-start times within the range of 17.0 s to 21.9 s with a step size of 0.4 s (see Fig. 1) to analyze the vehicle's response under different driving situations, such as acceleration and deceleration phases. We conducted simulations with 11 different attack durations for each attack start time. In total, we performed 3575 attack simulations for the sinusoidal scenario, calculated as: 13 (*attack-start times*)*11 (*attack durations*)*25 (*attack values*).

Attack Injection Outcome Classification. To classify the outcomes of the simulations, we define four outcome categories based on the vehicles' deceleration profiles and collision incidents under barrage jamming attacks. These categories are described in Table 1.

5 Experimental Results and Evaluation

We conducted simulations of the barrage jamming attacks, as described in Section 4.2, using the *sinusoidal* scenario test setup outlined in Section 4.1. In this section, we first present the outcome classification results of our simulations. We then analyze severe outcomes and their correlation with noise power levels. To do so, we evaluate and compare the jamming resilience of the P1 algorithm against its enhanced variants, P1A and P1B. Additionally, we extend our analysis by comparing the resilience of P1A and P1B with other CACC algorithms available in Plexe (Consensus, Ploeg, and Flatbed).

5.1 P1 Simulation Results

Results of the experiments conducted on the P1 algorithm show that this control algorithm is highly susceptible to jamming attacks causing communication failures (see Fig. 2); this confirms the results from the previous studies [1,5,12]. The benign outcomes (represented by the blue curve) and the negligible out-

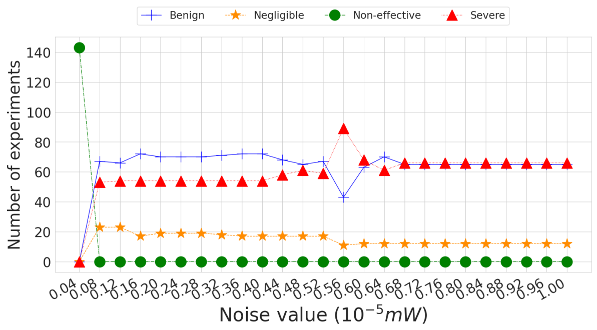


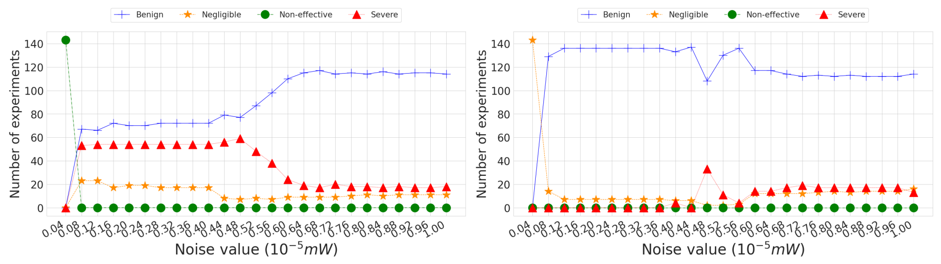
Fig. 2: Outcome distribution for the P1 algorithm w.r.t. the noise values.

comes (depicted by the orange curve) remain relatively stable across all attack values, indicating minimal variation in their occurrence. Additionally, for attack values at 0.04×10^{-5} mW, all attacks are classified as non-effective (represented by the green curve), implying that the message losses are not significant enough to deviate the trajectory of the vehicles. The proportion of severe outcomes remains relatively constant at around 42% across all noise values. When attacks occur during vehicle deceleration, the loss of communication does not necessarily lead to collisions, mitigating the overall impact of the jamming attack.

From 3575 simulation outcomes, 1475 results in collisions when P1 is subjected to barrage jamming attacks. This vulnerability of P1 is due to its lack of resilience against such attacks, as it does not incorporate any fallback mechanisms to mitigate communication failures. In the following sections, we analyze the severe outcomes of the two extensions to P1 that we presented in Section 3.

5.2 P1A Simulation Results

Fig. 3a illustrates the impact of barrage jamming on vehicle behavior when using the P1A algorithm. We observed that the severe outcomes are substantially reduced from 42% to 14% for noise values greater than 0.62×10^{-5} mW. This is due to the fact that the fallback mechanism implemented by P1A is triggered by a complete communication loss, which is much more likely to happen on higher noise values. For lower noise values, specifically the ones between 0.04×10^{-5} mW and 0.48×10^{-5} mW, certain vehicles experience partial communication failures due to the asymmetrical message losses, which do not meet the activation threshold for the P1A fallback mechanism, leading to an insignificant improvements in the severe results when compared with Those obtained for P1. The figure also shows a negative correlation between severe and benign outcomes. In other words, as the severe outcomes decrease, the benign outcomes increase.



(a) Results for when the fallback mechanism is activated on complete communication failure.

(b) Results for when the fallback mechanism is activated on partial communication failure.

Fig. 3: Outcome distribution for the P1A algorithm w.r.t. the noise values.

To further improve the jamming resilience of P1A, we decided to detect partial communication failures for every vehicle and activate the fallback mechanism upon such detections, in addition to those connected to complete communication failures. This enhancement ensures a more responsive CACC algorithm, reducing the likelihood of collisions under varying levels of communication failures.

The improved P1A version offers significant improvements in reducing collision incidents. As shown in Fig. 3b, no collisions are observed within the noise range of 0.04×10^{-5} mW to 0.36×10^{-5} mW. This highlights the effectiveness of the enhanced P1A fallback mechanism, which detects both complete and partial communication failures and proactively switches to fallback mode for as long as a vehicle experiences a communication failure.

In summary, P1A, with the ability to handle partial communication failures, can significantly reduce collision incidents. The collisions are reduced to zero in 0.04×10^{-5} mW to 0.36×10^{-5} mW range (see Fig. 3b). For noise values greater than 0.48×10^{-5} mW the collision incidents are reduced to 14% from 42% compared to the P1 algorithm. The remaining 14% collisions cannot be mitigated by the P1A as this algorithm is not a complete rewrite of the fallback activation mechanism but rather a change when the fallback activation condition is satisfied. Coming to why they occur in the first place in the P1A, we speculate that the degraded fallback mechanism itself is not sufficient enough to prevent collisions at such high noise values for long attack durations. To further enhance the jamming resilience against the remaining collisions, we extended P1 to P1B.

5.3 P1B Simulation Results

This section presents the outcomes of the P1B algorithm when subjected to barrage jamming attacks with varying noise levels. When the P1B algorithm is employed, the affected vehicle switches to ACC as a fallback mechanism upon detecting either partial or complete communication failure. As a result, the number of collisions is reduced to zero in both cases, as shown in Fig. 4. This significant improvement is due to ACC's sole reliance on onboard sensors for speed and

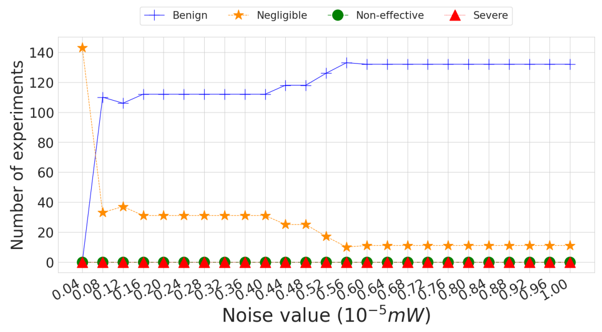


Fig. 4: Outcome distribution for when the fallback mechanism in the P1B algorithm is activated on partial communication failure.

distance measurement, eliminating dependence on V2V communication. These results highlight the effectiveness of using ACC as a fallback mechanism in enhancing vehicle resilience against barrage jamming attacks.

5.4 Comparison of Simulation Results with those obtained for the Consensus, Ploeg, and Flatbed CACC algorithms

To assess the jamming resilience of our enhanced P1A and P1B algorithms compared to other widely used CACC algorithms, we conducted targeted barrage jamming attack injection campaigns for the Consensus, Ploeg, and Flatbed. Our analysis revealed that both Consensus and Ploeg algorithms demonstrate high resilience against barrage jamming attacks, successfully avoiding collisions compared to the P1. This suggests that these control algorithms incorporate robust mechanisms connected to the usage of onboard sensors to maintain stability and safety even during communication failures. While both Consensus and Ploeg effectively mitigated severe outcomes, Consensus recorded a higher number of benign outcomes than P1B, whereas Ploeg experienced more severe outcomes than P1B. In contrast, the Flatbed algorithm demonstrated significant sensitivity to barrage jamming attacks, with 1764 out of 3575 simulations resulting in severe outcomes, out of which 1238 are collisions and 526 are emergency braking. Flatbed performs slightly better than the P1 in handling the impact of barrage jamming attacks. A ranking of the CACCs based on their resilience to barrage jamming attacks is provided in Table 2, offering a comparative assessment of their performance under adversarial conditions.

6 Summary and Future Work

This paper investigates the vulnerability of CACC algorithms to barrage jamming attacks, a prevalent cybersecurity threat in V2V communication [9]. Barrage jamming disrupts V2V communication by injecting noise-like energy across

Table 2: Ranking of CACC algorithms w.r.t the outcome classification.

Rank	**CACC**	**Severe**	**Benign**	**Negligible**	**Non-effective**
1	P1B	0	2947 (82.4%)	628 (17.6%)	0
2	Consensus	0	3551 (99.3%)	0	24 (0.7%)
3	Ploeg	9 (0.3%)	2840 (79.4%)	711 (19.9%)	15 (0.4%)
4	P1A	231 (6.5%)	2973 (83.2%)	371 (10.3%)	0
5	Flatbed	1764 (49.3%)	1530 (42.8%)	281 (7.9%)	0
6	P1	1475 (41.3%)	1591 (44.5%)	366 (10.2%)	143 (4%)

the entire portion of the spectrum occupied by the vehicle with 100% duty cycle in time [9], effectively hindering the reliable exchange of information critical for CACC functionality.

As our focus is to assess the jamming resilience of CACC algorithms, we chose the P1 (Plexe 1) algorithm for its heavy reliance on V2V communication to maintain close inter-vehicle spacing and platoon string stability. In contrast, other CACC algorithms, such as Consensus and Ploeg, are more jamming resilient but use onboard sensors and maintain larger inter-vehicle distances based on the speed of the platoon. P1 depends on the V2V communication to obtain the distance and speed of the predecessor and the leader vehicle, which makes it particularly susceptible to jamming attacks. To enhance the jamming resilience of P1, we extended it to P1A and P1B.

P1A incorporates the vehicle's onboard radar and increases inter-vehicle spacing to mitigate the impact of communication failures. P1B implements the ACC algorithm as a fallback mechanism, which triggers in case of V2V communication failures. P1A significantly improves jamming resilience, reducing the number of severe outcomes. P1A also mitigates collisions caused by partial communication failures. However, while significantly improved, P1A cannot entirely prevent collisions during complete communication failures. P1B, by leveraging the ACC as a fallback mechanism, achieves the most robust performance, eliminating severe incidents in both partial and complete communication failures.

Our study shows that improving the jamming resilience of the CACC algorithms that use a constant spacing policy is possible by carefully designing the fallback mechanisms. We also subjected the Consensus, Ploeg, and Flatbed to barrage jamming attacks to provide a comparative analysis. Our findings reveal that Consensus and Ploeg algorithms exhibit greater jamming resilience than Flatbed and the P1. However, the P1B is the best-performing algorithm against barrage jamming attacks in terms of collision mitigation and reduced benign outcomes. Moreover, basing P1B on P1 gives us closer spacing and string stability, and therefore traffic density compared to Consensus and Ploeg.

As part of our future work, we design and implement a physical testing methodology for barrage jamming attacks to validate the attack model we used in the simulations. This validation will help bridge the gap between simulation-based and physical testing environments, ensuring that our findings provide insights for enhancing the resilience of CACC algorithms against jamming attacks.

Acknowledgments. The work of this paper has been done in the context of the SUNRISE project, funded by the European Union's Horizon Europe Research and Innovation Actions under grant agreement no.101069573.

References

1. M. Malik, M. Maleki, P. Folkesson, B. Sangchoolie, and J. Karlsson. Comfase: A tool for evaluating the effects of v2v communication faults and attacks on automated vehicles. In *52nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN2022)*, pages 1–12. IEEE, 2022.
2. RISE Dependable Transport Systems. Comfase github repository for code access. `https://github.com/RISE-Dependable-Transport-Systems/ComFASE/`. Accessed: 27.07.2022.
3. M. Segata, S. Joerer, B. Bloessl, C. Sommer, F. Dressler, and R. L. Cigno. Plexe: A platooning extension for veins. In *2014 IEEE Vehicular Networking Conference (VNC)*, pages 53–60, 2014.
4. 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*, 8(4):695–708, 2000.
5. R. Van der Heijden, T. Lukaseder, and F. Kargl. Analyzing attacks on cooperative adaptive cruise control (cacc). In *2017 IEEE Vehicular Networking Conference (VNC)*, pages 45–52, 2017.
6. J. Ploeg, B.T.M. Scheepers, E. van Nunen, N. van de Wouw, and H. Nijmeijer. Design and experimental evaluation of cooperative adaptive cruise control. In *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 260–265, 2011.
7. 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)*, pages 1158–1166. IEEE, 2015.
8. A. Ali, G. Garcia, and P. Martinet. The flatbed platoon towing model for safe and dense platooning on highways. *IEEE Intelligent Transportation Systems Magazine*, 7(1):58–68, 2015.
9. M. Lichtman, J. D. Poston, S. D. Amuru, C. Shahriar, T. C. Clancy, R. M. Buehrer, and J. H. Reed. A communications jamming taxonomy. *IEEE Security and Privacy*, 14(1):47–54, 2016.
10. T. C. Clancy. Efficient ofdm denial: Pilot jamming and pilot nulling. In *2011 IEEE International Conference on Communications (ICC)*, pages 1–5, 2011.
11. J. A. Mahal, C. Shahriar, and T. C. Clancy. Emulated cp jamming and nulling attacks on sc-fdma and two novel countermeasures. In *MILCOM 2015 - 2015 IEEE Military Communications Conference*, pages 275–280, 2015.
12. M. Maleki, M. Malik, P. Folkesson, B. Sangchoolie, and J. Karlsson. Modeling and evaluating the effects of jamming attacks on connected automated road vehicles. In *2022 IEEE 27th Pacific Rim International Symposium on Dependable Computing (PRDC)*, pages 12–23. IEEE Computer Society, 2022.