Consensus-Based Fault-Tolerant Platooning for Connected and Autonomous Vehicles

ABSTRACT

Platooning is a representative application of connected and autonomous vehicles. The information exchanged between connected functions and the precise control of autonomous functions provide great safety and traffic capacity. Santini *et al.* [11] proposed a consensus-based approach for platooning, but it does not exploit the full strength of consensus to protect platooning against faulty information. Therefore, in this paper, we develop advanced consensus-based approaches for platooning. By applying consensus-based fault detection and adaptive gains to controllers, we can detect faulty position and speed information from vehicles and reinstate the normal behavior of the platooning. Experimental results show the effectiveness (small steady state errors) and efficiency (small settling times) of the developed approaches under scenarios with faults.

ACM Reference Format:
. 2020. Consensus-Based Fault-Tolerant Platooning for Connected and Autonomous Vehicles. In *Proceedings of International Conference On Computer Aided Design (ICCAD'20).* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Platooning is a representative application of connected and autonomous vehicles. Usually supported by the Cooperative Adaptive Cruise Control (CACC) on each vehicle, the information exchanged between vehicles and the precise control provide great safety and traffic capacity. As the technology of connectivity and autonomy advanced in recent years, platooning has been studied from many different perspectives [4–6, 11]. Especially, Santini *et al.* [11] proposed a state-of-the-art consensus-based approach for platooning. The main goal of the consensus-based approach is to construct a control system which establishes consensus between vehicles and forms the platoon. Though the proposed approach introduces the concept of consensus into platooning, it does not exploit the full strength of consensus to protect platooning against faulty information (the details are explained in Section 3.1). As a result, a platoon may converge to a wrong state if some vehicles send faulty messages, no matter they are general network faults or malicious attacks.

To address this problem, in this paper, we develop an advanced consensus-based approach consisting of *faulty-vehicle detection* and *controller-gain adjustment.* The faulty-vehicle detection analyzes the messages from vehicles and detects faulty position information from vehicles; the controller-gain adjustment eliminates the impacts from faulty *position* information and reinstates the normal behavior of the platooning. Based on the approach, we then develop another two-phase approach to detect faulty *speed* information from the leading vehicle and also reinstate the normal behavior of the platooning. Experimental results show that the developed approaches outperform the approach in [11] and achieve small steady state errors and small settling times.

The rest of this paper is organized as follows: Section 2 presents the system model and the problem formulation. Section 3 describes the developed consensus-based fault-tolerant controller which detects faulty position information and reinstates the platooning. Section 4 describes the two-phase approach which detects faulty speed information and reinstates the platooning. Section 5 provides experimental results, Section 6 reviews related work, and Section 7 concludes this paper.

ICCAD'20, November 2020, San Diego, CA
© 2020 Association for Computing Machinery.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

2 SYSTEM MODEL AND PROBLEM FORMULATION

The notation throughout this paper is listed in Table 1, and the system model and problem formulation are provided in this section.

Vehicle. There are N vehicles indexed from 0. All vehicles are connected and autonomous, and they are on the same lane to form a platoon. Each vehicle v_i periodically broadcasts its position, speed, acceleration, and other relevant information.

- The (true) position, speed, and acceleration of v_i at time t are $x_i(t)$, $v_i(t)$, and $a_i(t)$, respectively.
- The position, speed, and acceleration of v_i at t, as messages received by other vehicles, are $x'_i(t)$, $v'_i(t)$, and $a'_i(t)$, respectively, *i.e.*, if there is no fault, $x'_i(t) = x_i(t)$, $v'_i(t) = v_i(t)$, and $a'_i(t) = a_i(t)$.
- The initial position, speed, and acceleration of v_i are X_i, V_i, and A_i, respectively, *i.e.*, $x_i(0) = X_i$, $v_i(0) = V_i$, and $a_i(0) = A_i$.

The system model in [11] assumes that each vehicle in the platoon can receive the messages from the leading vehicle v_0. Here we assume that each vehicle in the platoon can receive the messages from all other vehicles in the platoon.

Controller. The controller C_i of v_i decides the acceleration $u_i(t)$ of v_i at t. For each activation period at t, C_i collects the messages from other vehicles and computes $u_i(t)$, where the computation involves the controller gains toward the messages from other vehicles. Note that v_i has its maximum acceleration U_i^+ and its maximum deceleration U_i^-. If the computed acceleration is larger (smaller) than U_i^+ (U_i^-), $u_i(t)$ will be set to U_i^+ (U_i^-).

Target States. The target state of v_i includes the *position target state* and the *speed target state.* The position target state of v_i ($i \geq 1$) is the safety distance between v_{i-1} and v_i, denoted as D_i, *i.e.*,

$$x_i(t) - x_{i-1}(t) = D_i, \tag{1}$$

Table 1: Notation throughout this paper.

Index	i, j	the index of a vehicle
	t	the time or the time step
Set	$\mathcal{I}^*$	the index set of faulty vehicles
Element	v_i	the i-th vehicle in the platoon (from 0)
	C_i	the controller of v_i
Given Parameter	N	the number of vehicles in the platoon
	X_i	the initial position of v_i
	V_i	the initial speed of v_i
	A_i	the initial acceleration of v_i
	U_i^+	the maximum acceleration of v_i
	U_i^-	the maximum deceleration of v_i
	D_i	the safety distance between v_{i-1} and v_i
	F_i	the difference from the true position of v_i
	F	the difference from the true speed of v_0
Decision Variable	$g_i(t)$	the controller gain of v_i toward the speed reference from v_0 at t
	$g_{i,j}(t)$	the controller gain of v_i toward the position reference from v_j at t
Dependent Variable	$x_i(t)$	the (true) position of v_i at t
	$v_i(t)$	the (true) speed of v_i at t
	$a_i(t)$	the (true) acceleration of v_i at t
	$x_i'(t)$	the (received) position of v_i at t
	$v_i'(t)$	the (received) speed of v_i at t
	$a_i'(t)$	the (received) acceleration of v_i at t
	$u_i(t)$	the controlled acceleration from C_i at t
	s_i	the settling time of v_i
	$\delta x_i(t)$	the position error of v_i at t
	$\delta v_i(t)$	the speed error of v_i at t

and the speed target state of v_i is the speed of v_0, *i.e.*,

$$v_i(t) = v_0(t). \tag{2}$$

Settling Time. The settling time s_i of v_i is the time for v_i to reach and stay within 5% of its final states (state values), *i.e.*,

$$s_i = \min \left\{ t \mid \forall t' > t, \left| 1 - \frac{x_i(t') - x_{i-1}(t')}{\lim\limits_{t'' \to \infty} (x_i(t'') - x_{i-1}(t''))} \right| < 5\% \right. \\ \left. \wedge \left| 1 - \frac{v_i(t')}{\lim\limits_{t'' \to \infty} v_i(t'')} \right| < 5\% \right\}. \tag{3}$$

The system settling time is the maximum settling time of all vehicles.

State Errors and Steady State Errors. The state errors of v_i at t includes the *position-state error* and the *speed-state error*. The position-state error of v_i at t is

$$\delta x_i(t) = x_i(t) - x_{i-1}(t) - D_i, \tag{4}$$

and the speed-state error of v_i at t is

$$\delta v_i(t) = v_i(t) - v_0(t). \tag{5}$$

The steady state errors of v_i are the corresponding state errors of v_i when $t \geq s_i$.

Fault Model. In this paper, we consider *faulty positions* and a *faulty speed.* A faulty position of v_i indicates

$$x_i'(t) = x_i(t) + F_i, \tag{6}$$

where F_i is the difference from the true value. A faulty speed of v_o indicates

$$v_0'(t) = v_0(t) + F, \tag{7}$$

where F is the difference from the true value. We assume that the index set of faulty vehicles $\mathcal{I}^*$ is *unknown* (not given) in advance.

Problem Formulation. Given the system model, decide

- The controller gain of v_i toward the speed reference from v_0 at t, $g_i(t)$, and
- The controller gain of v_i toward the position reference from v_j at t, $g_{i,j}(t)$,

to minimize

- The average steady state errors of all vehicles (primary objective), and
- The system settling time (secondary objective).

3 CONSENSUS-BASED FAULT-TOLERANT CONTROLLER

In this section, we first introduce the state-of-the-art approach [11] in detail (integrated with the notation of this paper) and explain the motivations in Section 3.1. We then describe our approach based on faulty-vehicle detection and controller-gain adjustment in Sections 3.2 and 3.3, respectively.

3.1 Motivations

Santini *et al.* [11] proposed a consensus-based approach for platooning. They observed that the previous research lacks the consideration of changing communication delays and thus proposed a controller which takes communication delays into account. The leading vehicle v_0 is a reference for platooning and is assumed globally reachable, meaning that $v_1, v_2, \ldots, v_{N-1}$ can receive the information sent from v_0. The objective of the controller C_i is to fix a desired safety distance D_i between v_{i-1} and v_i and set the speed $v_i(t)$ to approach $v_0(t)$ by computing

$$u_i(t) = - g_i(t)(v_i(t) - v_0(t)) \\ - \sum_{j=0}^{N-1} g_{i,j}(t) \cdot f(x_i(t), x_j(t), v_0(t)), \tag{8}$$

where there are other parameters and variables involved in the function f. Please refer to [11] for details.

After formulating the problem, $u_i(t)$ is rewritten with the terms of the state errors $\delta x_i(t)$ and $\delta v_i(t)$, recasting the problem to be a closed-loop platooning dynamics. Through some lemmas and theorems, the controller is theoretically proven to be asymptotically stable. The controller gains $g_i(t)$ and $g_{i,j}(t)$ also guarantee string stability if there is an upper bound of the communication delay. Compared with the classic CACC algorithm, the proposed consensus-based approach can maintain stable (the state errors $\delta x_i(t)$ and $\delta v_i(t)$ converge to 0), even with presence of communication delays.

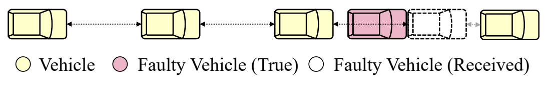


Figure 1: Faulty-vehicle detection.

Starting from the same controller in [11], we develop a more advanced approach with faulty-vehicle detection and controller-gain adjustment. There major motivations and differences are as follows:

- The approach in [11] does not consider faulty messages (no matter they are general network faults or malicious attacks), *i.e.*, $F_i = 0$ for each vehicle v_i. As a result, the corresponding platoon may converge to a wrong state. In our approach, we can deal with faulty positions (as messages), *i.e.*, $F_i \neq 0$.
- The implemented controller C_i of the approach in [11] only considers the speed of v_0 and the positions of v_0 and v_{i-1}, which means that the positions of other vehicles are not considered, *i.e.*, the controller gain $g_{i,j}(t) = 0$ if $j \neq 0$ and $j \neq i-1$. This setting is reasonable when there is no faulty message because the speed of v_0 and the positions of v_0 and v_{i-1} are the most relevant information for C_i. However, when there are faulty messages, C_i which relies on only few specific vehicles may converge to a wrong state. In our approach, we exploit the strength of consensus and consider the positions of other vehicles, *i.e.*, $g_{i,j}(t)$ is not necessarily zero if $j \neq 0$ and $j \neq i-1$.
- Also because there is no faulty message, the control gain $g_{i,j}(t)$ is set to be constant in the approach in [11]. In our approach, we further adaptively adjust the control gain $g_{i,j}(t)$ to eliminate the impacts on the platoon from faulty positions.
- Our approach can be extended to deal with the faulty speed of the leading vehicle v_0, which will be described in Section 4.

3.2 Faulty-Vehicle Detection

For v_i to detect a faulty vehicle v_j, the intuitive solution is to check the inconsistency of the true dynamics $x_j(t), v_j(t), a_j(t)$ and the received dynamics $x'_j(t), v'_j(t), a'_j(t)$, as illustrated in Figure 1. However, given that a vehicle can only know the dynamics of non-adjacent vehicles through received messages, the position-state error of v_j, *observed by other vehicles*, can be written as

$$x'_j(t) - x'_{j-1}(t) - D_j. \tag{9}$$

If there v_j is not faulty, the position-state error of v_j should converge to zero, *i.e.*,

$$x'_j(t) - x'_{j-1}(t) - D_j \longrightarrow 0, \tag{10}$$

when all vehicles become steady. However, if v_j is faulty, the position-state error of v_j should converge to F_j, *i.e.*,

$$x'_j(t) - x'_{j-1}(t) - D_j \longrightarrow F_j, \tag{11}$$

when all vehicles become steady. We would like to point out that whether v_{j-1} is faulty or not does not change Equations (10) and (11) because v_j behaves according to $x'_{j-1}(t)$, not $x_{j-1}(t)$. Based on Equations (10) and (11), the faulty-vehicle detection checks the position-state errors when all vehicles become steady. If the gap between v_j and v_{j-1} is abnormally larger than or small than D_j, the faulty-vehicle detection decides that v_j is a faulty vehicle.

Similar to [11], we assume that D_j is the same for each vehicle so that we can *compare* the position-state errors of gaps. The detailed design is described as follows. We set up an error count for each gap. For every pair of gaps, if they are 5% larger than or smaller than their average gap value (*i.e.*, very different from the other), the error counts of both gaps will increase by one. After comparing all pairs of gaps, we check the error count of each gap. If the error count of a gap between v_j and v_{j-1} is larger than or equal to $\frac{N}{2}$, the gap is regarded as an abnormal one, and v_j is regarded as a faulty vehicle. This leads to two properties as follows:

- The detection is not affected by an extremely large F_j — if F_j is extremely large, it is still counted once when comparing each pair of gaps. This property is not true if we use the average of all position-state errors for faulty-vehicle detection.
- If $F_j \geq \frac{D_j}{10}$ for each $j \in \mathcal{I}^*$ and $|\mathcal{I}^*| < \frac{N}{2}$, then the faulty-vehicle detection can detect these faults[1] — conceptually, it is a voting mechanism where $|\mathcal{I}^*| < \frac{N}{2}$ guarantees that there are sufficient non-faulty votes.

The faulty-vehicle detection will lead to the controller-gain adjustment (which is described in Section 3.3). In practice, when to trigger the detection is still an issue. If it is triggered when all vehicles become steady, then we can achieve high fault-tolerance, but the system settling time may become much larger; if it is triggered too early, then the dynamics of the most vehicles may have not converged, and thus many vehicles will be detected as faulty vehicles. To address this issue, we also propose to trigger the detection and the following adjustment when $\frac{N}{2}$ vehicles become steady, which can improve (shorten) the system settling time. It should be noted that this *fast fault-tolerant* approach does not have any additional communication overhead as each vehicle can decides the stability of each other vehicle by the received information.

Note that, if the assumption that D_j is the same for each vehicle is not held, we can assume that each vehicle knows the safety distances for all vehicles in front of it, which can be achieved by additional information exchanges between vehicles and/or roadside units. Similarly, if the gap between v_j and v_{j-1} is 5% larger than or smaller than the safety distance, v_j is regarded as a faulty vehicle.

3.3 Controller-Gain Adjustment

Since we cannot find the faulty vehicles in the beginning, all controllers are first set to be the same as that in [11], meaning that the controller C_i of v_i only considers the positions sent from v_0 and v_{i-1}, *i.e.*, $g_{i,0}(0) > 0$, $g_{i,i-1}(0) > 0$, and $g_{i,j}(0) = 0$ for $j \neq 0$ and $j \neq i-1$. With this setting, the system settling time can be small, but the steady state errors will be affected by faulty positions. The goal of the controller-gain adjustment here is to minimize the average steady state errors by adaptively adjust the control gains.

When to trigger the detection has been described in Section 3.2. Once it is triggered, it starts from the last vehicle v_{N-1} to the leading vehicle v_0. If v_j is detected as a faulty vehicle, then the controller

[1] $F_j \geq \frac{D_j}{10}$ comes from "5% larger than or smaller than their average gap value" in the detection design. "5%" is adjustable, depending on the fault-tolerant level decided by system designers.

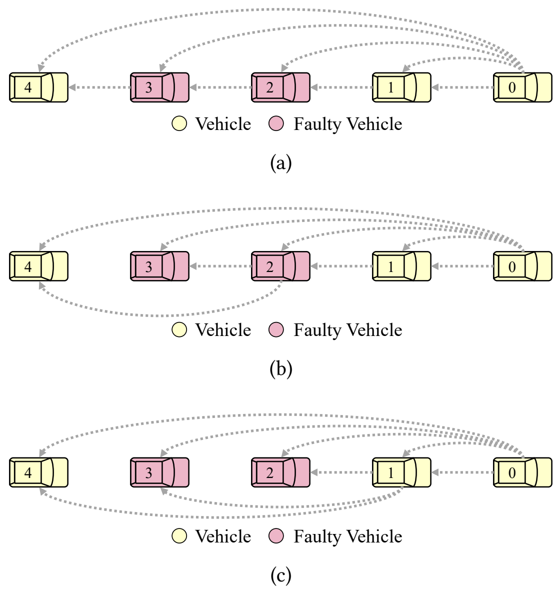


Figure 2: Controller-gain adjustment.

of v_i will adjust the controller gain by the following rule: if

$$g_{i,j}(t) \neq 0, \tag{12}$$

then

$$g_{i,j-1}(t) \leftarrow g_{i,j}(t) \text{ and } g_{i,j}(t) \leftarrow 0. \tag{13}$$

Following the properties in Section 3.2, if $F_j \geq \frac{D_j}{10}$ for each $j \in \mathcal{I}^*$ and $|\mathcal{I}^*| < \frac{N}{2}$, then the controller-gain adjustment can reinstate the normal behavior of the platooning.

An example with five vehicles is shown in Figure 2. In the example, v_2 and v_3 are faulty vehicles. In the beginning, all controllers are set to be the same, meaning that v_i only considers the positions of v_0 and v_{i-1}, and the logical communication topology is shown in Figure 2(a). Each vehicle triggers the detection and the following adjustment are triggered at the same time. As the rule above, they start from the last vehicle v_4 to the leading vehicle v_0. Once v_4 detects v_3 as a faulty vehicle, $g_{4,2}(t)$ is set to $g_{4,3}(t)$, and then $g_{4,3}(t)$ is set to zero. At this point, the logical communication topology is shown in Figure 2(b). Next, once v_4 and v_3 detects v_2 as a faulty vehicle, $g_{4,1}(t)$ and $g_{3,1}(t)$ are set to $g_{4,2}(t)$ and $g_{3,2}(t)$, respectively, and $g_{4,2}(t)$ and $g_{3,2}(t)$ are set to zero. At this point, the logical communication topology is shown in Figure 2(c). Since there is no other faulty vehicle, the logical communication topology will stay as Figure 2(c).

4 EXTENSION TO FAULTY SPEED

We have developed the consensus-based fault-tolerant approach to deal with faulty positions in Section 3. In this section, we extend the approach and develop a two-phase approach to deal with a faulty speed of the leading vehicle v_0. It should be emphasized that the first phase in the two-phase approach is exactly the approach in Section 3, meaning that we do not need to know the types (positions or speed) of faults in advance. We will also show that the developed approaches can deal with the combination of faults in Section 5.

The model of the faulty speed of v_0 is defined as Equation (7). Each vehicle v_i in the platoon only considers the positions of other vehicles and the speed of v_0, so the faulty speeds of vehicles, except of v_0, do not cause steady state errors or break the platoon. With the controller defined in Section 3, we observe that the faulty speed of v_0 results in position-state errors, while all vehicles still converge to the true speed of v_0. This is because the controller tends to stabilize the corresponding vehicle. If the vehicles in the platoon have different speeds from the true speed of v_0, the platoon will not become stable as the distances between vehicles constantly change. As a result, the controller follows Equation (8) and generates position-state errors.

Based on the observation, we develop a two-phase approach to deal with a faulty speed of v_0. In the first phase, we apply our fast fault-tolerant approach in Section 3. If the position-state errors result from the faulty speed of v_0, the controller-gain adjustment can still be triggered. However, adjusting $g_{i,j}$ does not eliminate the position-state errors as the errors result from the faulty speed of v_0. Therefore, if the position-state errors are not eliminated after the controller-gain adjustment, we know that the errors result from the faulty speed of v_0. In the second phase, since the position-state errors are not eliminated, all vehicles except v_0 give up the speed of v_0 and regard v_1 as the dummy leading vehicle, meaning that $g_i(t)$ becomes the controller gain of v_i toward the speed reference from v_1 at t. Then, $g_{i,0}$ is set to 0, and the other controller gains are reset to the initial values because the controller-gain adjustment in the first phase is misled by the faulty speed of v_0.

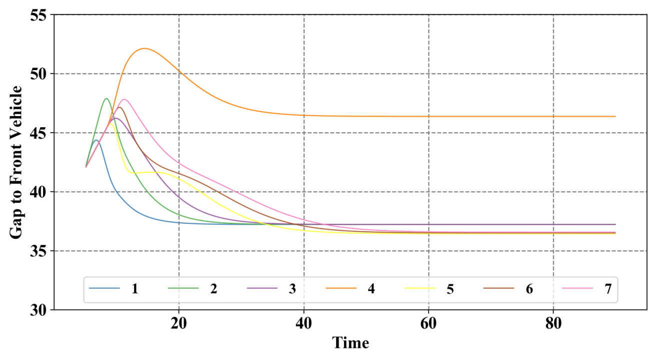
5 EXPERIMENTAL RESULTS

The whole experiments were simulated with Plexe [12] and run on a macOS mojave notebook with 2.7 GHz Intel CPU and 8 GB memory. As the approach in [11] is open-sourced and also experimented with Plexe, we have successfully reproduced the results by plugging in the same parameters except some missing initial states. There are five scenarios, all with 8 vehicles ($N = 8$), in our experiments:

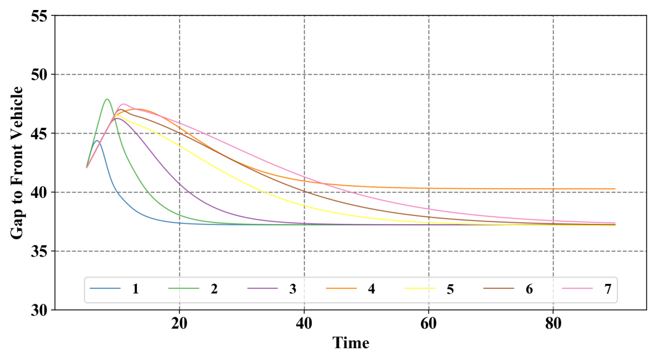
- **Scenario 1: Single Faulty Position**, where the position of one vehicle is faulty.
- **Scenario 2: Multiple Faulty Positions**, where the positions of multiple vehicles are faulty.
- **Scenario 3: Different Safety Distances**, where the position of one vehicle is faulty, and there are different safety distances between vehicles.
- **Scenario 4: Faulty Speed of Leading Vehicle**, where the speed of v_0 is faulty.
- **Scenario 5: Combination of Faults**, where the positions of multiple vehicles are faulty, and the speed of v_0 is faulty.

There are four experimented approaches for Scenarios 1–3:

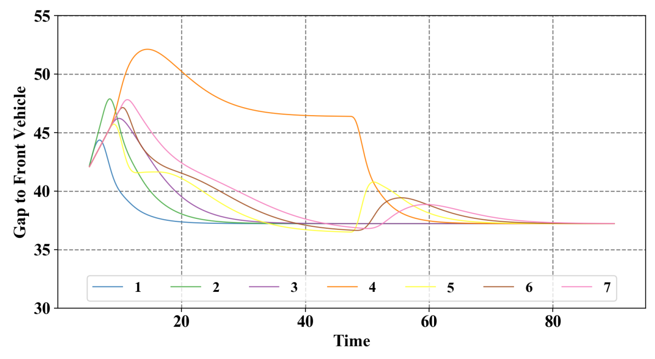
- **The Approach in [11]**, where v_i only considers the positions of v_0 and v_{i-1}, and the controller gain $g_{i,j}(t)$ maintains as a constant.
- **The "All-Front" Approach**, where v_i considers the positions of $v_0, v_1, \ldots, v_{i-1}$, and the controller gain $g_{i,j}(t)$ maintains as a constant.



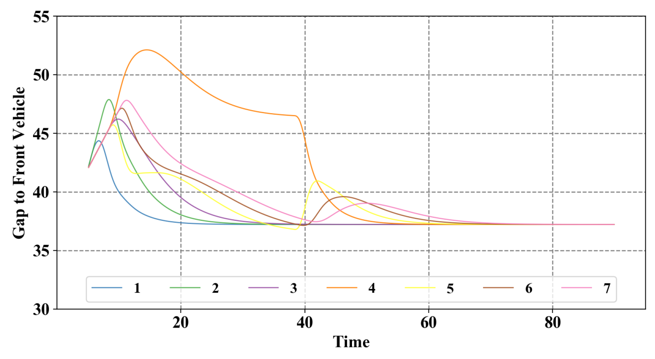
(a) The approach in [11].



(b) The all-front approach.



(c) Our fault-tolerant approach.



(d) Our fast fault-tolerant approach.

Figure 3: The gaps (m) to the front vehicles of $v_1, v_2, \ldots, v_{N-1}$ along the time (s) in Scenario 1 (single faulty position).

Table 2: Experimental results of Scenario 1.

Approach	System Settling Time (s)	Avg. Steady Position-State Error (m)
[11]	35.6	1.61
All-Front	54.9	0.46
Fault-Tolerant	58.5	< 0.01
Fast Fault-Tolerant	49.9	< 0.01

- **Our Fault-Tolerant approach** which starts the faulty-vehicle detection when all vehicles are steady and executes the controller-gain adjustment.
- **Our Fast Fault-Tolerant Approach** which starts the faulty-vehicle detection when $\frac{N}{2} = 4$ vehicles are steady and executes the controller-gain adjustment.

For Scenarios 4–5, the approach in [11] and our two-phase approach are experimented.

5.1 Scenario 1: Single Faulty Position

The given parameters are listed as follows (the units are in meter (m), second (s), or meter per second (m/s)):

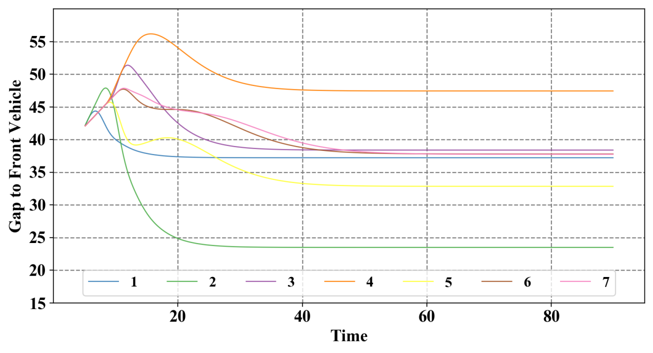
- $X_i = 46 \cdot (8 - i)$ for $i = 0, 1, \ldots, 7$.
- $V_i = (27, 25, 23, 22, 21, 20, 19, 18)$ for $i = 0, 1, \ldots, 7$.
- $A_i = 0$, $U_i^+ = 2.5$, and $U_i^- = 9$ for all i [11].
- $D_i = 37.2$ for $i = 1, 2, \ldots, 7$. [11].
- $g_{1,0}(0) = 460$, $g_{i,0}(0) = 80$ for $i = 2, 3, \ldots, 7$, $g_{i,i-1}(0) = 860$ for $i = 1, 2, \ldots, 7$, and $g_{i,j}(0) = 0$ for the remaining cases.
- $F_3 = -10$, and each other F_i is 0.
- $F = 0$.

The experimental results with single faulty position are listed in Table 2, and the gaps to the front vehicles along the time are shown in Figure 3. All vehicles controlled by all approaches have almost zero steady speed-state errors, so we only report and discuss steady position-state errors. As the approach in [11] only considers the positions of v_0 and v_{i-1}, v_4 is affected by the faulty positions sent from v_3, resulting a large average steady position-state error. On the other hand, as the all-front approach considers the positions of $v_0, v_1, \ldots, v_{i-1}$, v_4 is still affected by the faulty positions sent from v_3 but mitigated by the correct positions sent from v_0, v_1, v_2, resulting a smaller average steady position-state error, compared with the approach in [11]. Our fault-tolerant approach applies the faulty-vehicle detection and the controller-gain adjustment and has an almost zero average steady position-state error. The system settling time is shortened by the fast fault-tolerant approach — it is even smaller than that of the all-front approach.

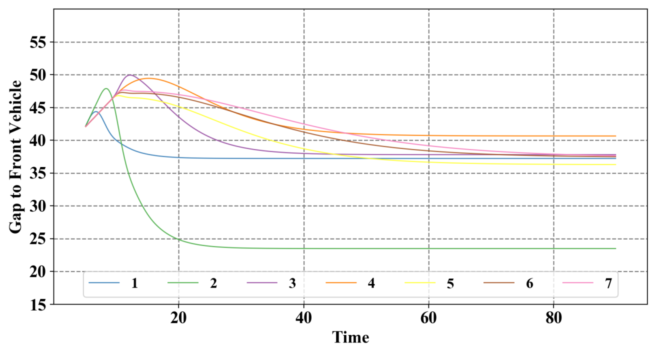
5.2 Scenario 2: Multiple Faulty Positions

The given parameters are the same as Scenario 1, except the following parameters (the units are in meter (m)):

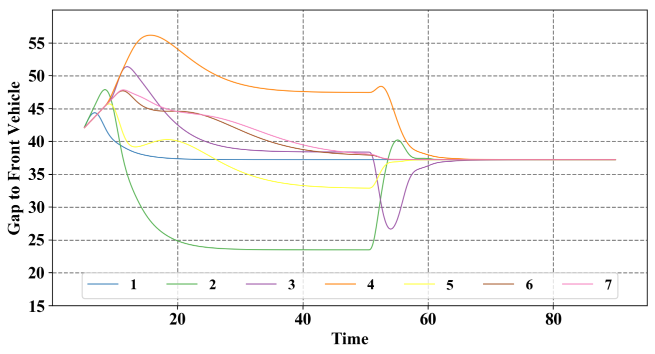
- $F_1 = 15$, $F_3 = -10$, $F_4 = 5$, and each other F_i is 0.



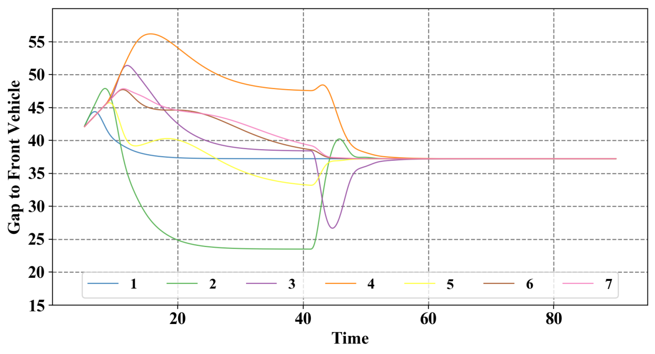
(a) The approach in [11].



(b) The all-front approach.



(c) Our fault-tolerant approach.



(d) Our fast fault-tolerant approach.

Figure 4: The gaps (m) to the front vehicles of $v_1, v_2, \ldots, v_{N-1}$ along the time (s) in Scenario 2 (multiple faulty positions).

Table 3: Experimental results of Scenario 2.

Approach	System Settling Time (s)	Avg. Steady Position-State Error (m)
[11]	39.3	4.37
All-Front	59.2	2.72
Fault-Tolerant	57.7	< 0.01
Fast Fault-Tolerant	48.3	< 0.01

Table 4: Experimental results of Scenario 3.

Approach	System Settling Time (s)	Avg. Steady Position-State Error (m)
[11]	24.5	1.61
All-Front	39.3	0.46
Fault-Tolerant	41.2	< 0.01
Fast Fault-Tolerant	32.3	< 0.01

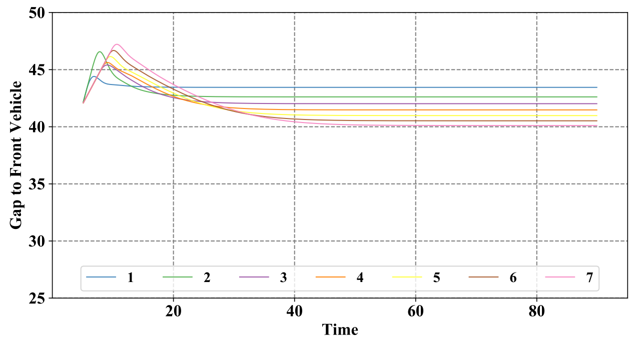
The experimental results with multiple faulty positions are listed in Table 3, and the gaps to the front vehicles along the time are shown in Figure 4. Similarly, all vehicles controlled by all approaches have almost zero steady speed-state errors, so we only report and discuss steady position-state errors. The results are similar to those in Scenario 1. The approach in [11] has a large average steady position-state error, and the all-front approach has a smaller average steady position-state error, compared with the approach in [11]. Again, our fault-tolerant approach has an almost zero average steady position-state error, and the system settling time is shortened by the fast fault-tolerant approach. We would like to point out that there is actually a trade-off between the fault-tolerant approach and the fast fault-tolerant approach. In some other scenarios, it is possible that the fault-tolerant approach can have a smaller average steady position-state error than the fast fault-tolerant approach, especially when there are more faulty vehicles.

5.3 Scenario 3: Different Safety Distances

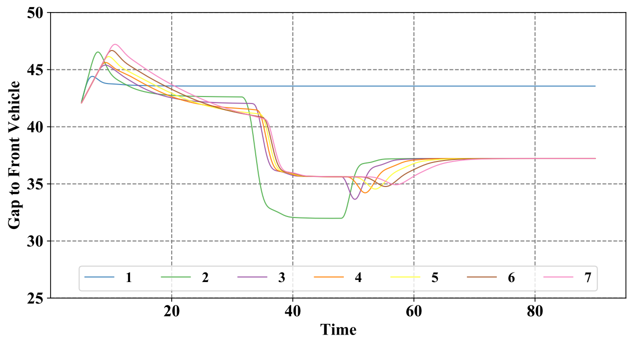
The given parameters are the same as Scenario 1, except the following parameters (the units are in meter (m)):

- $D_1 = 52.2$, $D_7 = 72.2$, and each other D_i is 37.2.

The experimental results with different safety distances are listed in Table 4, and the gaps to the front vehicles along the time are omitted due to the limited space. With the assumption that each vehicle knows the safety distances for all vehicles in front of it, the four experimented approaches have the same advantages and disadvantages as described in Scenarios 1 and 2. Our fault-tolerant approach has an almost zero average steady position-state error, and the system settling time is shortened by the fast fault-tolerant approach. The advantages and disadvantages are also the same if we consider multiple faulty positions in this scenario.

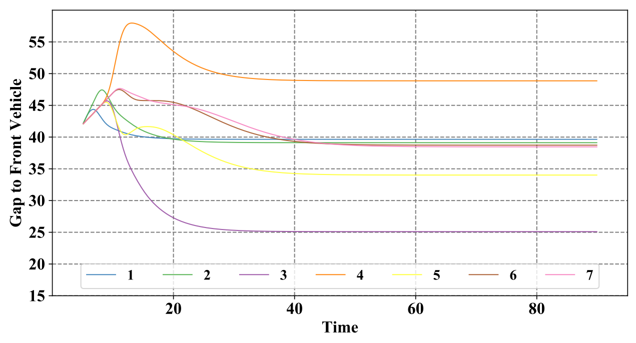


(a) The approach in [11].

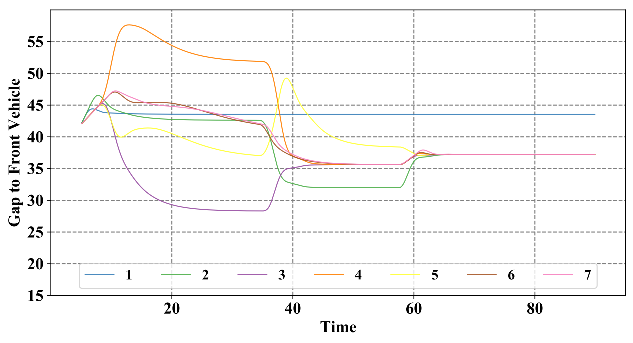


(b) Our two-phase approach.

Figure 5: The gaps (m) to the front vehicles of $v_1, v_2, \ldots, v_{N-1}$ along the time (s) in Scenario 4 (faulty speed of v_0).



(a) The approach in [11].



(b) Our two-phase approach.

Figure 6: The gaps (m) to the front vehicles of $v_1, v_2, \ldots, v_{N-1}$ along the time (s) in Scenario 5 (combination of faults).

Table 5: Experimental results of Scenario 4.

Approach	System Settling Time (s)	Avg. Steady Position-State Error (m)
[11]	26.4	4.06
Two-Phase	59.0	< 0.01

Table 6: Experimental results of Scenario 5.

Approach	System Settling Time (s)	Avg. Steady Position-State Error (m)
[11]	36.4	5.27
Two-Phase	60.7	< 0.01

5.4 Scenario 4: Faulty Speed of Leading Vehicle

The given parameters are the same as Scenario 1, except the following parameters (the units are in meter (m) or meter per second (m/s)):

- $F_i = 0$ for all i.
- $F = -2$.

The experimental results of faulty speed of v_0 are listed in Table 5, and the gaps to the front vehicles along the time are shown in Figure 5. Note that we do not include the steady state errors of the second vehicle in the calculation because v_0 is like leaving the platoon, and the second vehicle becomes the dummy leading vehicle after applying the two-phase approach. The approach in [11] has a large average steady position-state error, and the two-phase approach has an almost zero average steady position-state error. In the first phase of the two-phase approach, due to the position-state errors result from the faulty speed of v_0, the controller-gain adjustment is triggered, and we can observe a significant turning point at $t = 31.3$ in Figure 5(b). At $t = 47.8$, all vehicles become stable, but the position-state errors are not eliminated, which implies that the errors result from the faulty speed of v_0. Therefore, in the second phase of the two-phase approach, the vehicles behave as described in Section 4 and achieves an almost zero average steady position-state error.

5.5 Scenario 5: Combination of Faults

The given parameters are the same as Scenario 1, except the following parameters (the units are in meter (m) or meter per second (m/s)):

- $F_2 = 15$, $F_3 = -10$, $F_4 = 5$, and each other F_i is 0.
- $F = -2$.

The experimental results of faulty speed of v_0 are listed in Table 6, and the gaps to the front vehicles along the time are shown in Figure 6. In this scenario, the two-phase approach detects the faulty speed of v_0 and regards the second vehicle as the dummy leading

Table 7: Experimental results of different numbers of vehicles.

N	System Settling Time (s)	Avg. Steady Position-State Error (m)
8	48.3	< 0.01
12	61.6	< 0.01
16	66.3	< 0.02

vehicle. Then, the faulty-vehicle detection (which can be regarded as a routine function) detects faulty positions, and the controller-gain adjustment reinstates the platooning. As a result, we have an almost zero average steady position-state error, showing the capability of dealing with the combinations of faults.

5.6 Discussion

The two-phase approaches are also experimented with different numbers of vehicles. We set N as 12 and 16 and have the same faulty positions as Scenario 2. The experimental results are listed in Table 7. When N is larger, we can observe that the system settling time is longer, and the average steady position-state error is larger. However, by the observation, the system settling time is sublinear to N, and the steady position-state error is still close to zero.

6 RELATED WORK

Collaborative Adaptive Cruise Control (CACC) is an emerging system for platooning connected and autonomous vehicles. However, the CACC algorithm might be vulnerable to external malicious attacks, causing safety and security threats to connected and autonomous vehicles. Previous work has been done to defense the attacks through the implementation of fault-tolerance or intrusion detection. van Nunen *et al.* [13] focused on sensor faults and proposed an algorithm to calculate safe distances and guarantee the safety for CACC. Jagielski *et al.* [7] considered attacks that modify the position, speed, or acceleration information in messages and affect a victim vehicle and proposed approaches using physical constraints and machine-learning-based intrusion detection algorithms. Alotibi and Abdelhakim [1] proposed another intrusion detection algorithm, which fuses the information of the leading vehicle from other vehicles and fixed infrastructure and precludes the deviated data to detect attacks in CACC.

On the other hand, consensus algorithms have been well studied. Lamport [8] proposed the Paxos algorithm, which can be implemented for a fault-tolerant distributed system. Cavin *et al.* [3] reformed the traditional consensus algorithms to handle the dynamic in mobile ad hoc networks. Ongaro and Ousterhout [9] developed the Raft algorithm, which simplified and optimized the design of Paxos algorithm. Benchi *et al.* [2] presented a new consensus algorithm that can be applied to opportunistic networks. Some research specifically concentrated on the automotive applications of consensus algorithms. Zhang *et al.* [14] proposed a consensus tracking algorithm for a multi-vehicle system with a dynamic topology. Petit and Mammeri [10] presented a consensus application in vehicular ad hoc networks (VANET) with dynamic inputs according to uncertainty of surrounding environments.

There is also research about optimization of CACC or platooning design from different perspectives. Dao *et al.* [4] formulated the platooning vehicle assignment problem with the implementation of a linear model. Through solving the linear model, the platooning assignment problem is optimized. Hao *et al.* [5] optimized the sequence of platooning vehicles considering vehicles joining or leaving the CACC system. Their objective of the formulation is to find the minimum energy consumption. Heinovski and Dressler [6] further took the individual behaviors of each vehicle into account in a platooning assignment problem. With different preferences or requirements of each vehicle, the optimal vehicles platooning assignment could be determined by the proposed algorithm.

7 CONCLUSION

we developed advanced consensus-based approaches for platooning. By applying consensus-based fault detection and adaptive gains to controllers, we can detect faulty position and speed information from vehicles and reinstate the normal behavior of the platooning. Experimental results showed the effectiveness (small steady state errors) and efficiency (small settling times) of the developed approaches under scenarios with faults. Potential future work includes different types of faults, malicious attacks including collusion between vehicles, and consensus problems in other applications of connected and autonomous vehicles.

REFERENCES

[1] F. Alotibi and M. Abdelhakim. Anomaly detection in cooperative adaptive cruise control using physics laws and data fusion. In *IEEE Vehicular Technology Conference (VTC-Fall)*, pages 1–7, Sep 2019.

[2] A. Benchi, P. Launay, and F. Guidec. Solving consensus in opportunistic networks. In *International Conference on Distributed Computing and Networking*, pages 1:1–1:10, 2015.

[3] D. Cavin, Y. Sasson, and A. Schiper. Consensus with unknown participants or fundamental self-organization. *Lecture Notes in Computer Science*, 3158, Jul 2004.

[4] T.-S. Dao, J. P. Huissoon, and C. Clark. A strategy for optimisation of cooperative platoon formation. *International Journal of Vehicle Information and Communication Systems*, 3, Aug 2013.

[5] P. Hao, Z. Wang, G. Wu, K. Boriboonsomsin, and M. Barth. Intra-platoon vehicle sequence optimization for eco-cooperative adaptive cruise control. In *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–6, Oct 2017.

[6] J. Heinovski and F. Dressler. Platoon formation: Optimized car to platoon assignment strategies and protocols. In *IEEE Vehicular Networking Conference (VNC)*, pages 1–8, Dec 2018.

[7] M. Jagielski, N. Jones, C.-W. Lin, and C. Nita-Rotaruand S. Shiraishi. Threat detection for collaborative adaptive cruise control in connected cars. In *ACM Conference on Security & Privacy in Wireless and Mobile Networks (WiSec)*, pages 184–189, 2018.

[8] L. Lamport. Paxos made simple. *ACM SIGACT News (Distributed Computing Column) 32, 4 (Whole Number 121)*, pages 51–58, Dec 2001.

[9] D. Ongaro and J. Ousterhout. In search of an understandable consensus algorithm. In *USENIX Annual Technical Conference (ATC)*, pages 305–319, Jun 2014.

[10] J. Petit and Z. Mammeri. Dynamic consensus for secured vehicular ad hoc networks. In *IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pages 1–8, Oct 2011.

[11] S. Santini, A. Salvi, A. S. Valente, A. PescapÃĺ, M. Segata, and R. L. Cigno. A consensus-based approach for platooning with inter-vehicular communications. In *IEEE Conference on Computer Communications (INFOCOM)*, pages 1158–1166, April 2015.

[12] M. Segata, S. Joerer, B. Bloessl, C. Sommer, F. Dressler, and R. L. Cigno. Plexe: A platooning extension for Veins. In *IEEE Vehicular Networking Conference (VNC)*, pages 53–60, Dec 2014.

[13] E. van Nunen, J. Ploeg, A. M. Medina, and H. Nijmeijer. Fault tolerancy in cooperative adaptive cruise control. In *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, pages 1184–1189, Oct 2013.

[14] J. Zhang, C. Sun, L. Wang, and J. Xia. Consensus tracking for multi-vehicle system with a well-informed leader: Adaptive control method. In *Chinese Control and Decision Conference (CCDC)*, pages 1050–1054, May 2011.