2023 Design, Automation & Test in Europe Conference (DATE 2023)
Special Initiative "Autonomous Systems Design"

Phalanx: Failure-Resilient Truck Platooning System

Changjin Koo[1], Jaegeun Park[2], Taewook Ahn[1], Hongsuk Kim[1], Jong-Chan Kim[1], and Yongsoon Eun[2]
[1]Graduate School of Automotive Engineering, Kookmin University, Seoul, Korea
[2]Department of Electrical Engineering and Computer Science, DGIST, Daegu, Korea
Correspondence: jongchank@kookmin.ac.kr

***Abstract*—We introduce Phalanx, a failure-resilient truck platooning system, where trucks in a platoon protect each other from sensor failures despite the lack of redundant sensors. For that, we first emulate the failed sensors by collectively utilizing other sensors across the platoon. If the failed sensor cannot be emulated, the control system is instantaneously reconfigured to a cooperative protection mode using only the live sensors. We take a scenario-based approach considering six scenarios with single and dual failures of the essential sensors (i.e., lidar, encoder, and camera) for platooning control. For each scenario, we present a protection method that enables the safe maneuvering of platoons. For the evaluation, Phalanx is implemented using our scale truck testbed instrumented with fault injection modules, demonstrating safe platooning controls for the failure scenarios.**

I. Introduction

Truck platooning is an automation technology that maintains close gaps between trucks at high speeds for enhanced fuel efficiency and safety [1]–[3]. The *leading vehicle* (LV) is driven by a human driver, whereas the *following vehicles* (FVs) are autonomous with various sensors. Due to the strong dynamics of such heavy-duty vehicles, even a single sensor failure in a platooning truck can cause a catastrophic disaster. Thus, the platooning system should be *resilient* to such failures by retaining its safe maneuvering capability.

Most failure-resilient systems are based on *redundant sensors* [4], [5]. However, making every sensor redundant leads to undesirable complexity in terms of system architecture. In contrast, we make the trucks protect each other without sensor redundancy by exploiting platooning trucks' unique driving patterns and connectivity. Using the vehicle-to-vehicle (V2V) communication between trucks, even *remote* sensors in other trucks can be utilized to mitigate *local* sensor failures. Let us take an example of a velocity sensor failure. Then we can estimate the velocity by combining the preceding truck's remote velocity and the in-between gap distance from a local sensor.

Platooning trucks are equipped with various sensors [2], [3], where the essential ones are

- Range sensor (e.g., lidar or radar),
- Velocity sensor (e.g., encoder), and
- Vision sensor (e.g., camera).

The range and velocity sensors are for the longitudinal control, whose primary objective is to maintain the string stability with a constant gap between trucks. The vision sensor is primarily for the lateral control by detecting lanes. Besides, trucks are connected through the V2V network to exchange real-time information for the platooning control.

With the above motivation, we present a failure-resilient truck platooning system, which is named *Phalanx* since the trucks remind us of the ancient soldiers protecting each other by tactical movements with spears and shields. We implemented a testbed with three 1/14 scale platooning trucks (Fig. 1) that are introduced in [6], equipped with three essential sensors (i.e., lidar, encoder, and camera) and computers running perception and control algorithms. In our testbed, the LV is also autonomous, which is the only difference from the real-world platooning with a human driver in charge of the LV. Thus, this study focuses on the FVs since they have to be fully autonomous even in emergencies.

Fig. 1: Scale truck platooning testbed equipped with lidar, encoder, front- and rear-facing camera sensors.

With our scale trucks, we take a scenario-based approach, where six sensor failure scenarios are considered: three *single* sensor failures and three *dual* sensor failures. The case of triple sensor failures is not considered. In such an unrecoverable situation, an immediate safe stop should be enforced instead of letting the failed trucks run on highways. Under the six scenarios, we aim to preserve the platoon's minimum safety driving capability such that the trucks can be mobilized to a safe location.

Our initial approach is to *emulate* failed sensors by utilizing other (possibly remote) sensors. For example, if the lidar sensor fails, the front-facing camera can be utilized to *estimate* the gap by locating the preceding trailer in the camera image. If the encoder fails, the preceding truck's velocity and the gap distance can be utilized to estimate the velocity, as previously presented. However, we need a different approach for the front-facing camera failure since emulating a camera is not viable in our configuration. As an alternative approach, the control system is reconfigured from lane-keeping control to target tracking control upon a camera failure. Under the new control mode, the control objective is to follow the preceding trailer's track by the lidar sensor instead of following lanes.

For dual sensor failures, one approach is to combine two individual protection methods, each of which developed for single sensor failures. This approach works well for the lidar

978-3-9819263-7-8/DATE23/© 2023 EDAA

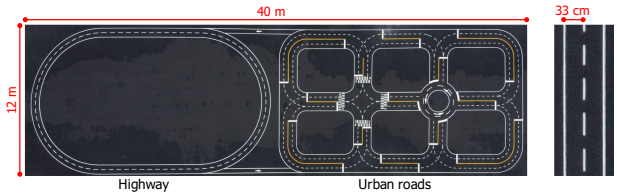


Fig. 2: Scale truck proving ground.

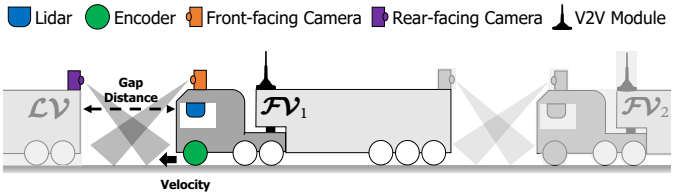


Fig. 3: System model of platooning trucks.

and encoder sensor failures and the encoder and camera sensor failures. However, it is not applicable to the camera and lidar sensor failures because the lidar failure protection requires the camera while the camera failure protection requires the lidar, leading to a mutual contradiction. For that, our solution is to utilize the preceding truck's *rear-facing* camera. In many countries, a rear-facing camera is mandated in heavy-duty trucks for the safety monitoring purpose [7]. Then the troubled truck can observe its own state through the preceding truck's rear-facing camera to make control decisions.

For the evaluation, we instrumented the platooning system with fault injection modules that artificially trigger sensor failures. Faults are injected based on the failure scenarios while the trucks are running on our scale proving ground (Fig. 2). The rigorous fault injection tests demonstrate that Phalanx can safely protect the trucks against the six failure scenarios.

The contributions of this study can be summarized as:

- We propose a resilient truck platooning architecture, where the trucks protect each other from sensor failures without the need for redundant sensors.
- The proposed architecture is implemented based on our scale truck testbed, where our failure protection methods are evaluated for single and dual sensor failures.

II. Background

A. System Model

Fig. 3 shows our system model for a truck platoon. There are N identical trucks, where the LV is denoted by $\mathcal{LV}$ and the FVs are denoted by $\mathcal{FV}_i \in \{\mathcal{FV}_1, \mathcal{FV}_2, \cdots, \mathcal{FV}_{N-1}\}$, in the platooning order. The set of FVs is denoted by $\mathcal{FV}_*$. Although $\mathcal{LV}$ is driven by a human driver in the real-world platooning, we assume that all the trucks are autonomous. Each truck is equipped with four sensors: (i) lidar, (ii) encoder, (iii) front-facing camera, and (iv) rear-facing camera. The first three are *essential* because they provide indispensable data for the platooning control. The lidar measures the gap distance. The encoder measures the velocity. The front-facing camera detects the lane. In contrast, the rear-facing camera at each trailer's rear end is not essential since it is for safety monitoring. However, even the rear-facing camera has to be utilized in an emergency. Besides, the trucks are connected through a wireless V2V network.

For the platooning control, $\mathcal{LV}$ is given a reference *velocity*, while $\mathcal{FV}_*$ is given a reference *gap distance*. Thus, they have different control objectives (i.e., velocity control and gap control, respectively), letting the velocity of the platoon be governed by $\mathcal{LV}$'s velocity. For $\mathcal{FV}_*$'s longitudinal control, their main control objective is to maintain the string stability between trucks with a constant gap. For the lateral control, both *lane-keeping* control and *target tracking* control can be applied, where our primary choice is the lane-keeping control, by which each truck follows the center of its perceived lanes.

B. Failure Scenarios

With the above system model, the following six failure scenarios are considered:

- Lidar failure, denoted by Ⓛ.
- Encoder failure, denoted by Ⓔ.
- Camera failure, denoted by Ⓒ.
- Lidar and Encoder failures, denoted by Ⓛ + Ⓔ.
- Encoder and Camera failures, denoted by Ⓔ + Ⓒ.
- Lidar and Camera failures, denoted by Ⓛ + Ⓒ.

For the simplicity of explanation, the term "camera" means a "front-facing camera" by default throughout this study. When referring to a "rear-facing camera", its full title is always used. We do not consider the rear-facing camera's failure since it is not an essential sensor, meaning that the rear-facing camera is not used for the platooning control in normal situations. Among the scenarios, the first three are single sensor failures, whereas the remaining three are more complex scenarios with dual sensor failures. We do not consider the case of triple essential sensor failures in this study.

Regarding the failure modes of sensors, we assume the sensors are fail-silent, meaning that if there is a sensor failure, the sensor becomes silent and does not provide incorrect data. Besides, we assume that the sensor diagnostic test interval, as defined in ISO 26262 (Part 3), is small enough to detect the fault without noticeable delays. Thus, our focus is not on detecting failures but on the protection methods after failure detection.

C. Problem Description

With the above system model and failure scenarios, our problem is to develop sensor failure protection methods for the six scenarios. To be precise, we do not expect to maintain the optimal control performance in such abnormal situations. Instead, we try to maintain a minimum safety driving capability such that the platooning trucks do not pose threats to road safety by moving themselves to a safe place. By the minimum safety driving capability, we expect the platoon to slow down to a safe velocity while maintaining a somewhat relaxed gap distance and keeping themselves at least within the lane. Throughout this study, we assume sensor failures only in $\mathcal{FV}_1$. However, without loss of generality, the developed methods are equally applicable to other FVs.

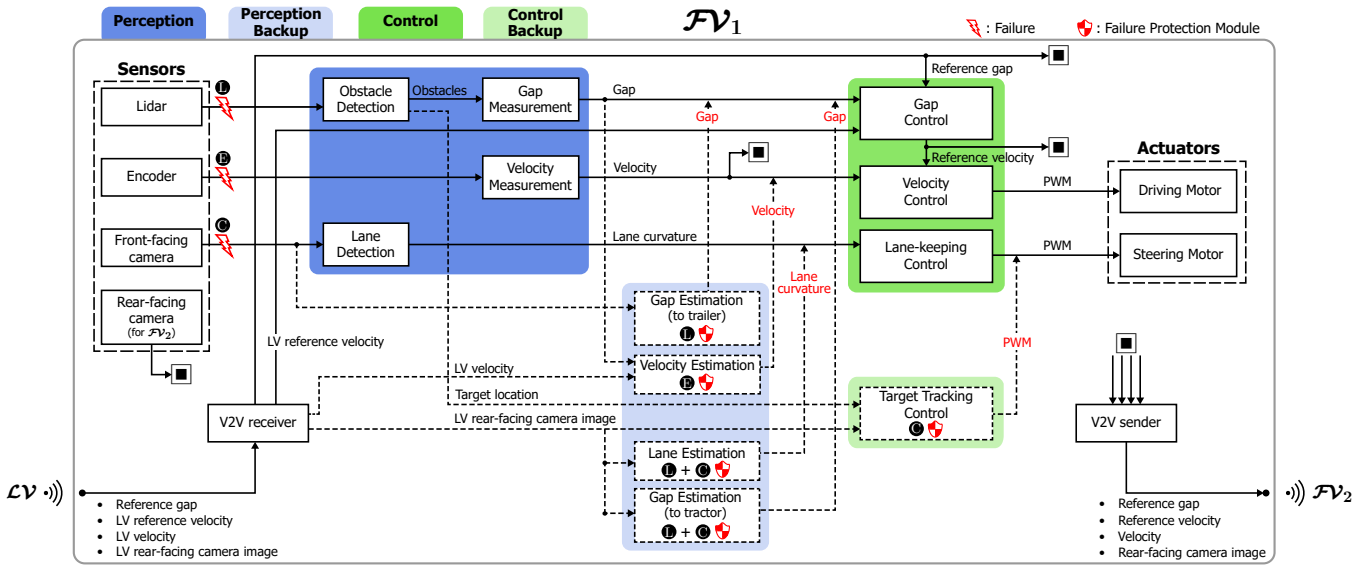


Fig. 4: Phalanx architecture for the failure-resilient truck platooning.

III. Resilient Truck Platooning Architecture

Fig. 4 shows our failure-resilient truck platooning architecture both in normal operations and failure protection operations, assuming the ego truck is $\mathcal{FV}_1$ with its preceding truck $\mathcal{LV}$.

In normal driving situations, data flow through the solid lines. For example, the lidar data go through the obstacle detection module to the gap measurement module to the gap control module. The encoder data go through the velocity measurement module to the velocity control module that produces actuation signals (PWM) to the driving motor. Meanwhile, the lane detection module uses the front-facing camera's image to produce the lane curvature, which in turn is used by the lane-keeping control module to make steering signals (PWM) to the steering motor. Even in normal driving situations, the gap control module requires the preceding truck's reference velocity and reference gap distance through the V2V receiver module. For more about normal platooning operations, refer to [6].

Upon sensor failures, data flow from the failed sensors stop, as depicted by the red lightning icons with circled letters. In such cases, since the original perception modules can no longer produce outputs, the backup modules become active, enabling new data flow depicted by dashed lines. For example, for Ⓛ, the gap distance is no longer available, then the gap estimation (to trailer) module utilizes the front-facing camera to estimate the gap. For Ⓔ, the velocity is not available. Then our velocity estimation module utilizes the preceding truck's velocity (from the V2V receiver module) and the gap to the preceding truck (from the gap measurement module) to estimate the ego truck's velocity. For Ⓒ, the original lane-keeping control is no longer available. Since there is no other way of detecting lanes without the camera, our solution is to switch the lateral control mode from lane-keeping control to target tracking control. Then the target tracking control module uses the preceding truck's location from the obstacle detection module as its target. The ego truck can now follow the preceding truck using the pure pursuit algorithm [8] based on the lidar sensor. However, the bent angle between the tractor and the trailer in semi-trailer trucks may significantly mislead the ego truck on curved roads. To solve this problem, the preceding truck's rear-facing camera image (from the V2V receiver module) is utilized to correct the target point by estimating the lateral offset of the trailer from the lane center.

For the dual failures of Ⓛ + Ⓔ and Ⓔ + Ⓒ, the individually developed protection methods can be used in combination. However, in Ⓛ + Ⓒ, we cannot reuse the individual failure protection methods since the lidar failure protection method requires the camera, while the camera failure protection method requires the lidar. In that case, our solution is to switch to an emergency control mode that depends on the preceding truck's rear-facing camera. Then the rear-view image is delivered to the lane estimation module and the gap estimation (to tractor) module, where the gap and lane information is extracted from the image for the gap and lane-keeping controls.

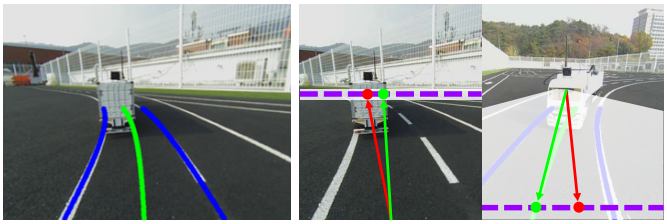
IV. Failure Protection Methods

A. Gap Estimation

To estimate the gap to the preceding trailer in the case of Ⓛ, we use a 2D object detection module that finds relevant objects in the camera image. Since we are interested in the location of the preceding trailer, the object detection network is retrained by our custom-collected images of trailers. Fig. 5a shows an example image that detects a trailer. Since this image is in the camera coordinate, it should be transformed into a bird's-eye view (BEV) coordinate as in Fig. 5b. Then we count the pixels between the bottom line and the trailer's rear end. During the camera calibration, we found that the 1.0 m distance in the real world corresponds to 490 vertical pixels in the BEV coordinate, by which the gap distance can be precisely estimated.

(a) Detecting the trailer. (b) Estimating the gap.

Fig. 5: Camera-based gap estimation.



(a) Lane keeping. (b) Target tracking.

Fig. 6: Lane-keeping control vs. target tracking control.

B. Velocity Estimation

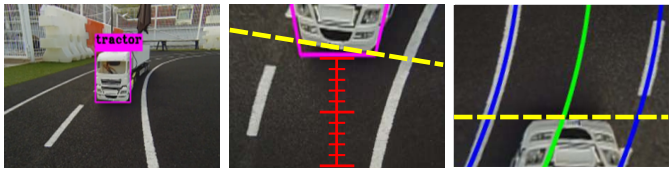
The encoder measures velocity by counting the rotation of the driving motor. In the case of E, we can estimate the ego truck's velocity (v_i) by combining the preceding truck's velocity (v_{i-1}) and the measured gap distance (d_i) between them. For the notational simplicity, the index i is assumed to be 0 for $\mathcal{LV}$, while it corresponds to the index in $\mathcal{FV}_*$ for the remaining trucks. Since v_{i-1} and d_i are measured at discrete time points with its sampling index k, they are denoted as functions of k (i.e., $v_{i-1}(k)$ and $d_i(k)$). Then the estimated velocity $\hat{v}_i(k)$ can be calculated as

$$\hat{v}_i(k) = v_{i-1}(k) - \frac{d_i(k) - d_i(k-1)}{T_s}, \tag{1}$$

where T_s is the sampling period for the gap measurement. The second term on the right-hand side represents the instantaneous velocity change between the $(k-1)$-th and k-th sampling points. To be precise, the transmitted velocity $v_{i-1}(k)$ can be a few steps behind due to the wireless network latency, causing slight delay errors. A first-order low-pass filter is used to minimize estimation errors caused by delays and also lidar sensor noises.

C. Target Tracking Control

In the case of C, we can no longer use *lane-keeping control* that solely depends on the camera as in Fig. 6a. Then our lateral control module is reconfigured to use *target tracking control* that follows the preceding trailer based on the pure pursuit algorithm [8] as in Fig. 6b. At first, the preceding trailer can be located by the lidar sensor, and its center point, depicted by a red dot, looks like a reasonable target point for the target tracking control. However, this naïve approach does not always work well due to the off-tracking phenomenon in semi-trailer trucks [9], where curved roads make significant bent angles between the tractor and the trailer. As illustrated in Fig. 6b, with a curvature to the left-hand side, the trailer's rear end naturally approaches the left lane. Thus, blindly following the center point will invade the ego truck's left lane. For this off-tracking problem, our solution is to utilize the preceding truck's rear-facing camera. By detecting lanes in the rear-view image as in the right-hand side of Fig. 6b, the offset between the trailer's center point (i.e., red dot) and the lane's center point (i.e., green dot) can be estimated. Then the offset can help correct the incorrect target angle (i.e., red arrows) toward the lane center (i.e., green arrows).



(a) Detecting the tractor. (b) Estimating the gap. (c) Estimating the lane.

Fig. 7: Rear-facing camera-based gap and lane estimation.

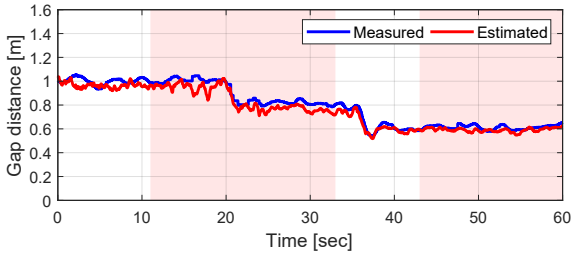
D. Combining Individual Protection Methods for Dual Failures

For the dual failures of L + E and E + C, their respective failure protection methods can be simply combined without any modification. However, for L + C, combining their individual protection methods presents a conflict that can never be resolved. Recall that the protection method for L utilizes the *camera* to estimate the gap distance, while the protection method for C requires the *lidar* to enable the target tracking control. Due to this deadlock-like situation, we cannot simply combine the two protection methods. Instead, we need to devise a different method, which will be presented in the next section.

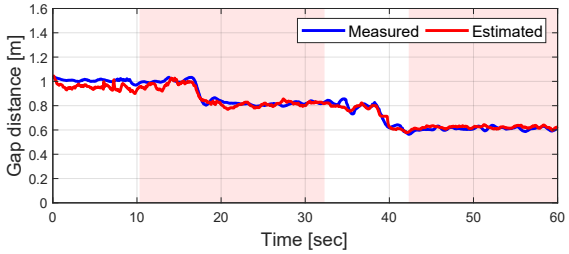
E. Rear-facing Camera-based Lane and Gap Estimation

For the dual failure of L + C, our solution is to switch to an emergency control mode that primarily depends on the preceding truck's rear-facing camera. Fig. 7a shows a rear-view image that detects the ego truck's tractor, where we can estimate the gap distance and lane information. Fig. 7b shows a BEV image that illustrates the gap estimation method, which is similar to the method in Section IV-A. The figure shows a slight discrepancy between the tractor and the bounding box caused by the road curvature. We found that this discrepancy is negligible for the gap estimation. However, it cannot be neglected when deciding the steering angle for the lateral control. Thus, the BEV image is rotated to make the tractor's face (depicted by the yellow dashed line) horizontal as in Fig. 7c. Then the left and right lanes can be estimated, producing the desired path depicted by the green line.

For the implementation, the preceding truck sends raw rear-view images to the ego truck through the V2V wireless network, and the ego truck processes the gap and lane estimation from the image. It would be interesting to compare this section's rear-facing camera-based lane-keeping control with the lidar-based target tracking control in Section IV-C in terms of the lateral control performance, which will be presented in the experimental section.



(a) Front-facing camera-based (to trailer).



(b) Rear-facing camera-based (to tractor).

Fig. 8: Evaluation of gap estimation.

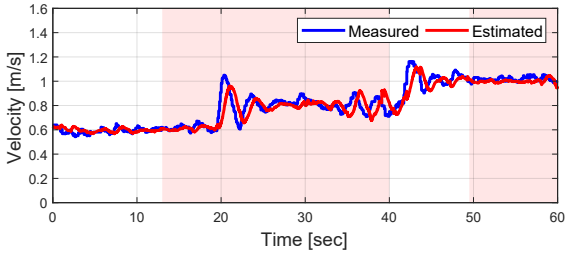


Fig. 9: Evaluation of velocity estimation.

V. Experiments

A. Implementation

Three 1/14 scale trucks are developed with a lidar (RPLidar A3), a magnetic encoder, and two USB cameras with 180 degrees field of view (FoV). The trucks are denoted by $\{\mathcal{LV}, \mathcal{FV}_1, \mathcal{FV}_2\}$. Among them, $\mathcal{FV}_1$ is the major experimental platform where faults are injected. As computing platforms, an Nvidia Jetson AGX Xavier platform (high-level controller) and an OpenCR embedded board (low-level controller) are used. For the software platform, we use the robot operating system (ROS), where perception and control modules are developed as ROS nodes. Since an object detection module is required, a YOLO-based object detection node is added to the ROS network. More specifically, a YOLOv3-tiny network is trained using our custom dataset of 2551 images captured by $\mathcal{FV}_1$'s front-facing camera and 1676 images captured by $\mathcal{LV}$'s rear-facing camera for detecting the trailer and the tractor, respectively. For the V2V communication, we use the 802.11ac WIFI network.

B. Evaluation

Fig. 8 compares the camera-based gap estimation method (Section IV-A and Section IV-E) with the lidar-based gap measurement. In the figure, white areas denote the straight road segments while red areas denote the left curved road segments in the proving ground (Fig. 2). Both experiments are conducted by ramping down the reference gap from 1 m to 0.8 m to 0.6 m while the velocity is fixed at 0.8 m/s. The average error is 3.86 cm in Fig. 8a and 2.54 cm in Fig. 8b, respectively, having no meaningful impact on the longitudinal control. One observation is that the estimation becomes more accurate with shorter gaps. It is due to the disproportionate pixel-to-pixel conversion ratio from the camera coordinate to the BEV coordinate, which depends on the distance. It is a positive property considering the close gaps between platooning trucks.

Fig. 9 compares the velocity estimation method (Section IV-B) with the encoder-based velocity measurement. The reference velocity is ramped up from 0.6 m/s to 0.8 m/s to 1.0 m/s while the reference gap is fixed as 0.8 m. The figure shows that the estimated velocity closely follows the measured velocity, however, with noticeable time lags, which are vivid around the time points 20 and 43. The time lag is caused by the wireless network delay between $\mathcal{LV}$ and $\mathcal{FV}_1$ and the processing delay of the low-pass filter in $\mathcal{FV}_1$. Its impact will be evaluated later with scenario-based experiments.

Fig. 10 compares the lateral error of the lane-keeping control and the target tracking control (Section IV-C) with and without the rear-facing camera-based correction while the gap is 0.8 m and the velocity is 0.6 m/s. In the figure, the lateral error of the target tracking control without our correction method invades the left lane on the curved road segments (red areas) due to the off-tracking problem. In contrast, with our correction method, the target tracking control ties with the lane-keeping control without noticeable performance degradation.

Fig. 11 evaluates the rear-facing camera-based lane-keeping control (Section IV-E) at the velocity of 0.6 m/s and the gap of 0.8 m. The figure compares the lateral error of the front-facing camera-based and the rear-facing camera-based lane-keeping

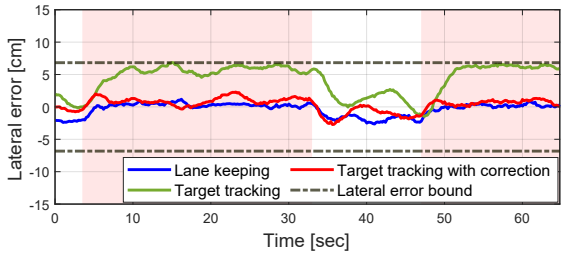


Fig. 10: Evaluation of lidar-based target tracking.

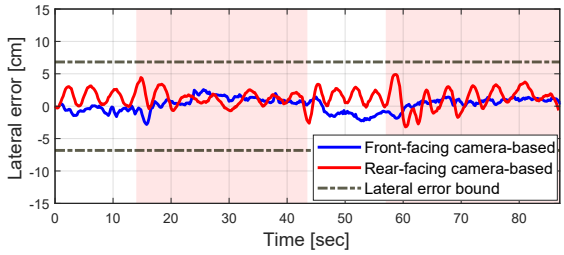
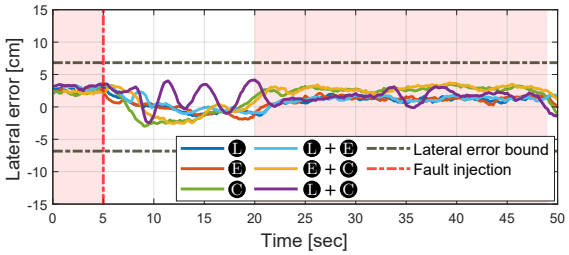
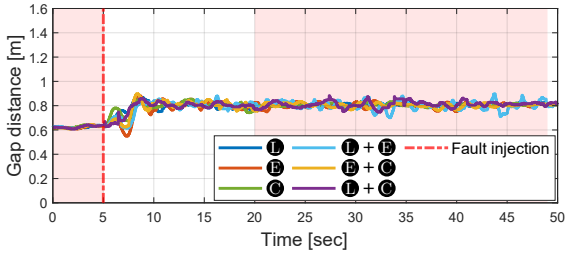


Fig. 11: Evaluation of rear-facing camera-based lane keeping.



(a) Lateral control performance.



(b) Longitudinal control performance.

Fig. 12: Failure scenario-based fault injection tests.

methods. Although both methods satisfy the given lateral error bound without invading lanes, the rear-facing camera-based control suffers significant fluctuations caused by the estimation delay noticed in Fig. 9.

Fig. 12 shows the results of the fault injection tests with the six scenarios. The platoon initially maneuvers in the normal mode with the gap of 0.6 m and the velocity of 1.0 m/s. Then faults are injected at time 5, and the platoon goes into an emergency mode of a reduced velocity (0.6 m/s) and an increased gap distance (0.8 m). Fig. 12a shows that the lateral errors are maintained close to zero except the Ⓛ + Ⓒ case that shows significant fluctuations caused by the limited performance of the rear-facing camera-based control. However, it is still within the lateral error bound. Fig. 12b shows stable gap control performance across all the scenarios. One interesting observation is the undershoot by Ⓔ right after the fault injection, which happens to be intensified by the sudden slow down of $\mathcal{LV}$ and its velocity's delayed transmission, making $\mathcal{FV}_1$ mistakenly speed up briefly.

VI. Related Work

Many fail-operational autonomous systems employ redundancy architectures [4], [5], which have inherent cost issues. However, safety is an issue of life and death rather than an issue of cost. In this regard, we are not blindly against the redundancy architecture due to the cost reason. Rather, our proposed method further enhances the safety of heavy-duty platooning trucks, regardless of their redundancy architecture. Similarly, there are many studies on platooning trucks from various aspects, for example, in terms of network issues [10], [11], actuator faults [12], [13], and redundancy and diversity [14]. In [15], a sensor fault detection and mitigation method is presented assuming platooning vehicles, however, within a limited scope of the longitudinal control.

VII. Conclusion

This study presents Phalanx, a failure-resilient truck platooning system that exploits platooning trucks' unique driving patterns and connectivity. Six sensor failure scenarios are considered and mitigated by emulating failed sensors and switching control modes through cooperation between platooning trucks. Phalanx is implemented on our scale trucks, which successfully protect each other from sensor failures.

Acknowledgment

The authors would like to thank the late Prof. Kihong Park, who motivated us to begin our truck platooning research. This work was supported partially by the BK21 Four Program (5199990814084) of NRF funded by the Ministry of Education, Korea, partially by an NRF grant funded by the Korean government (MSIT) (2022R1A2C1013197), and partially by the DGIST R&D Program of the Ministry of Science and ICT (22-Bridge-01). J.-C. Kim is the corresponding author.

References

[1] C. Bergenhem, S. Shladover, E. Coelingh, C. Englund, and S. Tsugawa, "Overview of platooning systems," in *Proc. 19th World Congress on Intelligent Transport Systems (ITS)*, 2012, pp. 1–8.

[2] S. Tsugawa, S. Jeschke, and S. E. Shladover, "A review of truck platooning projects for energy savings," *IEEE Transactions on Intelligent Vehicles*, vol. 1, no. 1, pp. 68–77, Mar. 2016.

[3] T.-W. Kim, W.-S. Jang, J. Jang, and J.-C. Kim, "Camera and radar-based perception system for truck platooning," in *Proc. 20th International Conference on Control, Automation and Systems (ICCAS)*, 2020, pp. 950–955.

[4] E. van Nunen, R. Koch, L. Elshof, and B. Krosse, "Sensor safety for the european truck platooning challenge," in *Proc. 23rd World Congress on Intelligent Transport Systems (ITS)*, 2016, pp. 306–311.

[5] J. Axelsson, "Safety in vehicle platooning: A systematic literature review," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 5, pp. 1033–1045, May. 2017.

[6] H. Lee, J. Park, C. Koo, J.-C. Kim, and Y. Eun, "Cyclops: Open platform for scale truck platooning," in *Proc. 39th IEEE International Conference on Robotics and Automation (ICRA)*, 2022.

[7] M. Esser, "Standardization and vehicle regulation aspects of camera monitor systems," in *Handbook of camera monitor systems*, 2016, pp. 51–100.

[8] R. C. Coulter, "Implementation of the pure pursuit path tracking algorithm," Carnegie-Mellon University Pittsburgh PA Robotics Institute, Tech. Rep., 1992.

[9] Y. Lee, T. Ahn, C. Lee, S. Kim, and K. Park, "A novel path planning algorithm for truck platooning using V2V communication," *Sensors*, vol. 20, no. 24, p. 7022, Dec. 2020.

[10] J. Ploeg, N. van de Wouw, and H. Nijmeijer, "Fault tolerance of cooperative vehicle platoons subject to communication delay," *IFAC-PapersOnLine*, vol. 48, no. 12, pp. 352–357, 2015.

[11] A. Khalil, M. Al Janaideh, K. F. Aljanaideh, and D. Kundur, "Fault detection, localization, and mitigation of a network of connected autonomous vehicles using transmissibility identification," in *Proc. 69th American Control Conference (ACC)*. IEEE, 2020, pp. 386–391.

[12] W. Wang, B. Han, Y. Guo, X. Luo, and M. Yuan, "Fault-tolerant platoon control of autonomous vehicles based on event-triggered control strategy," *IEEE Access*, vol. 8, pp. 25 122–25 134, 2020.

[13] C. Pan, Y. Chen, Y. Liu, and I. Ali, "Adaptive resilient control for interconnected vehicular platoon with fault and saturation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 8, pp. 10 210–10 222, Aug. 2022.

[14] T. Bijlsma and T. Hendriks, "A fail-operational truck platooning architecture," in *Proc. 28th IEEE Intelligent Vehicles Symposium (IV)*, 2017, pp. 1819–1826.

[15] M. R. Hidavatullah, J.-C. Juang, Z.-H. Fang, and W.-H. Chang, "Heterogeneous platooning vehicle with robust sensor fault detection and estimation," in *Proc. 5th IEEE International Symposium on Computer, Consumer and Control (IS3C)*, 2020, pp. 436–439.