IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

IEEE VEHICULAR TECHNOLOGY SOCIETY SECTION

Received 16 November 2023, accepted 12 January 2024, date of publication 22 January 2024, date of current version 9 February 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3357357

RESEARCH ARTICLE

A Testing and Verification Approach to Tune Control Parameters of Cooperative Driving Automation Under False Data Injection Attacks

JAMES C. HOLLAND[1], (Graduate Student Member, IEEE), FARAHNAZ JAVIDI-NIROUMAND[1], ALA' J. ALNASER[2], AND ARMAN SARGOLZAEI[1], (Senior Member, IEEE)

[1]Department of Mechanical Engineering, University of South Florida, Tampa, FL 33620, USA
[2]Department of Applied Mathematics, Florida Polytechnic University, Lakeland, FL 33805, USA

Corresponding author: Farahnaz Javidi-Niroumand (farahnaz.javidi@gmail.com)

This work was supported in part by the National Science Foundation under Grant ECCS-EPCN-2241718 and Grant CNS-1919855.

ABSTRACT Control systems are used in safety-critical applications where tuning the system parameters is required to ensure safe and secure operation. The process of tuning these parameters can be arduous, time-consuming, and unreliable as they are dependent on the operating environment. In this paper, we discuss a testing and verification approach for tuning the control parameters of a secure cooperative adaptive cruise controller (CACC) while simultaneously testing the safety of the algorithm under false data injection (FDI) attacks. In our approach, we use particle swarm optimization (PSO) to tune the parameters of the controller and observer. The performance of the controller will be evaluated before and after the optimization of control and detection parameters. After employing several swarms, it was noticed that the global optimal solution is reached within 74 iterations, on average. In summary, the configurations found by each swarm ensured that a safe following distance was achieved throughout testing. In terms of FDI estimation, however, the more conservative configuration with the minimum optimal parameter values performed the best.

INDEX TERMS Cooperative driving automation, false data injection attack, Lyapunov stability, parameter tuning, particle swarm optimization, testing, verification.

I. INTRODUCTION

Connected autonomous vehicles (CAVs) are an active area of research that seeks to develop safe and efficacious self-driving vehicles that communicate with one another. The benefits of this technology is innumerable, including improvements to traffic management and congestion, reducing vehicle emissions, and – most importantly – reducing traffic fatalities. CAVs utilize vast systems of sensors and controllers that rely on tunable parameters to enable decision-making and piloting, using cooperative driving automation (CDA). Determining the optimal values for these parameters is an arduous task and is heavily dependent on the operating environment. In order to achieve ideal results, the controller and observer gains should be tuned to implement the optimal solution. This ensures that CAVs and CDAs deliver on their promises of safe, efficient, and stress-free transit and services. Cooperative adaptive cruise control (CACC) is one of the many popular CDA systems. CACC is similar to adaptive cruise control (ACC) but leverages wireless connectivity to enable vehicle-to-everything (V2X) communication. As such, CACC can enhance the capabilities of ACC systems with cooperative maneuvers and increase proliferation by reducing the sensors required to safely operate. CACC improves the reaction time to potential threats and traffic conditions which could result in increased roadway utilization while reducing traffic congestion [1], [2], [3].

The associate editor coordinating the review of this manuscript and approving it for publication was Hassan Omar.

Designing a safe CACC algorithm is a daunting task due to two major factors. Firstly, a controller can only be designed for the platform and environment that it is expected to operate within. Traditionally, this was accomplished by either directly measuring the environment or by making assumptions about the operating conditions. While this does indeed suffice in some situations, the mission of CACC is too critical to cut such corners. Second, the wireless connectivity of CACC introduces inherent vulnerabilities to system stability that can be exploited by adversarial factors or even by the stochastic nature of reality.

To further improve controller safety and reliability are resilient controllers. Resilient controller design is an area within the field of control engineering that seeks to establish designs and principles that guarantee a system will behave as designed, even in contested environments. Traditionally, this involved the utilization of observers and Lyapunov design principles [4]. In recent years, researchers have investigated utilizing techniques leveraged in other fields, such as neural-networks and online predictive algorithms for attack detection and mitigation [5], [6], [7]. Just as with any other controller or algorithm, resilient controllers require adequate testing and tuning to ensure the system operates as desired. However, there is a big gap in a mathematical-based approach to address this issue.

Optimal control is an active research area that seeks to solve exactly this problem. The objective of an optimal controller is to operate in a manner that minimizes a cost function [8]. There exists a myriad of techniques and algorithms that may be applied, depending on the system and use case. One of the classical methods is gradient descent which operates by iteratively tuning the control parameters to ''driving'' the cost function to a minima value, without guaranteeing that this is the global minima.Through our survey of the literature, particle swarm optimization (PSO) was selected for use in tuning the observer and controller parameters of a resilient CACC. The justification for this will be discussed further in Section II.

Testing CACC and CAVs, as a whole, is an exceedingly difficult task. Firstly, enumerating all the potential test cases that a given system could encounter under ideal circumstances is challenging. This issue is exacerbated when considering less-than-ideal operating conditions or adversarial actors, as would be necessary for testing CAVs and CDA. There exist two projects of note that have undertaken this endeavor, these being the Adaptive and Pegasus projects [9], [10]. These projects are interdisciplinary, leveraging insight and guidance from the fields of engineering as well as legal professionals and policymakers to identify the fundamental aspects of autonomous driving that need to be tested. To address this challenge, we have proposed a framework for dynamically generating test cases based on fundamental requirements and the performance of a given system while under testing.

In addition to test generation is the requirement for defining a means for test convergence. Currently, there is no defined standard for accomplishing this, however, there are many methods being employed. The first of these is through testing CAVs with real-world driving. Using this approach, approximately 80%–90% of the verification process has been accomplished. What remains, however, are the challenging and dangerous edge-case scenarios [11]. Furthermore, these scenarios are also very rare and often occur once every 100,000 miles [12]. Even at the current, unprecedented rate of testing, verification by this method alone is expected to take at least 200 years [13], [14]. To address this challenge, our proposed framework incorporates a means for verifying the performance and safety of the vehicle's decision-making algorithms.

This research will contribute to our other, ongoing work in CAV testing and verification [15]. The first novelty of this paper is in developing a secure CACC algorithm which can estimate and mitigate FDI attacks in real-time. Since there is no systematic technique to tune the parameters of the developed algorithm for its safe operation under FDI attacks, the second novelty of this work is in the utilization of PSO as an optimization approach in the context of our proposed testing and verification framework. To achieve this, we developed a novel verification cost function to measure the safety of CACC algorithm. This creates the foundation for our testing and verification framework to perform future controller tuning. The aforementioned aspects of this work have not been previously presented in the literature. Furthermore, a threat model and risk analysis was conducted for, booth, the untuned and tuned CACC proposed in this paper to guide the discussion and outline the goals of this paper.

The remainder of this paper is organized as follows. Section II discusses existing work related to this project and provides justifications for the selection of PSO. In Section III, the threat model and risk analysis of a CACC system is presented. Section IV discusses the dynamic model of CACC under FDI attack, along with the controller, observer, and attack estimation design. Furthermore, a stability analysis has been done in this section to ensure the overall system's stability. Section VI introduces the testing and verification approach. The PSO algorithm is defined in Section VII as a method to find the optimal parameters for the controller and observer. A detailed description of the experimental setup is given in Section VIII, and Section IX presents the results of this study. Lastly, Section X concludes that the proposed method is effective for tuning control parameters under FDI attack.

II. LITERATURE REVIEW

There exist a myriad of approaches for both the upper and lower level of CACC implementations. Many of the proposed upper-level control schemes utilize fuzzy logic, model predictive controls, and reinforcement-learning [16], [17], [18], [19]. Research improving upper-level controls often seek to increase safety, traffic flow, and efficiency through cooperative traffic management [20], [21]. The proposed low-level control methods primarily determine the

actuator inputs to achieve a necessary drivetrain output for maintaining speed. In [22], a soft actor-critic (SAC) based controller was leveraged to calculate the necessary engine torque to maintain desired vehicle speed. In [23], a PID control scheme was proposed for maintaining speed through adjusting of the engine's throttle position. Another PID control approach that manages both acceleration and deceleration using a logic switch was proposed in [24]. In [25], an efficiency-focused robust controller for electric vehicles was proposed.

To accomplish the task of CAV testing and verification, there are several approaches. Each of these considers either an individual vehicle subsystem or combinations of subsystems and occurring either in the real-world or simulated environments. Within simulated testing, there exist offline and real-time methods. Offline simulations seek to maximize the computation speed of testing while real-time simulations prioritize testing accuracy in a bounded response time [26]. Furthermore, real-time simulations maintain data accessibility and accelerate testing while ensuring certainty throughout the development process [27]. Contained within real-time simulations are the following methods: Simulation-in-the-Loop (SiL), Vehicle-in-the-Loop (ViL), and Hardware-in-the-Loop (HiL).

Within the area of optimal control, there exists a myriad of techniques and algorithms that may be applied, depending on the system and use case. A popular subset of optimization algorithms is metaheuristic algorithms which are capable of solving many nonlinear problems in a reasonable time [28], [29], [30]. These algorithms also benefit from faster convergence time compared to gradient descent, Newton method, and other derivative-based approaches while also being less susceptible to converging on local minima [31]. This is accomplished by using a two-phase approach, referred to as exploration and exploitation. Exploration is where the algorithm searches the problem space to discover the most ideal search area. During exploitation, the best solution for the current local area is found.

Metaheuristic optimization algorithms can be grouped into two main groups: genetic algorithms and swarm-based. Genetic algorithms are another branch of optimization techniques [32]. These are iterative algorithms that take inspiration from natural selection. Typically, these algorithms can be thought of as a distributed implementation of gradient descent where a population of many points searches the solution space for the ideal parameter solutions. At each iteration, the best solution found by the population, as a whole, is used to ''mutate'' the other members of the population such that the most globally optimal solution is converged on.

There exist many metaheuristic algorithms, such as MOEA/D [33], AMOEA/D [34], adaptive recurrent fuzzy algorithms [35], as well as plant-inspired approaches [36]. The authors in [33] present a method for adaptive control of wastewater treatment using a Multiobjective Evolutionary Algorithm Based on Decomposition (MOEA/D). In [34], the authors propose an adaptive recursive fuzzy neural network with Gustafson-Kessel (GK) clustering and a hierarchical adaptive second-order optimization algorithm (HAS). Proposed in [35] is an enhanced MOEA/D that incorporates a self-organizing collaborative scheme. In [36], a review of plant-based approaches the utilize recent developments in the understanding of plant decision-making.

Within the branch of genetic algorithms is the particle swarm optimization (PSO) algorithm [37]. PSO simulates the behavior of a particle swarm exploring the solution space. At each iteration, each point's velocity is altered based on the best solution found by each particle and the swarm, as a whole, such that the swarm converges on a finite area of the space. An advantage of PSO is the ability for widespread exploration of the solution space to prevent conversion on a local minima. In addition, PSO is an intuitive algorithm that is very easy to understand and implement [38]. As such, PSO was selected for tuning the control parameters of the resilient CACC presented in this paper.

III. THREAT MODEL AND RISK ANALYSIS

In this section, a threat model will be created and evaluated based on the approach laid out in ISO/SAE 21434: Road Vehicles - Cybersecurity Engineering. These steps are applied to our use case of a resilient CACC algorithm deployed on two vehicles traveling along a straight highway.

There exist several approaches for performing threat modeling and risk analysis. Three of which are appropriate for automotive applications are STRIDE [39], STPA-Sec [40], and ISO/SAE 21434:2021 [41].

A new cybersecurity analysis framework that aligns with the ISO/SAE 21434:2021 standard, using threat models and vulnerability scoring is presented in [42]. The framework was applied to real-life scenarios, revealing 199 potential cyber threats in Advanced Driver-Assistance Systems (ADAS) and the need for specific security measures in modern vehicles to address vulnerabilities to cyberattacks.

In another recent publication, the authors emphasize the importance of a robust threat modeling framework for autonomous vehicles (AVs), proposing a comprehensive framework to identify cyber-physical threats to AV perception systems based on mathematical modeling and a comparative analysis with the ISO/SAE 21434 standard [43].

Discussed in [44] is the need for advanced techniques in assessing the cybersecurity of cyber-physical systems and introduces the System-Theoretic Process Analysis for Security (STPA-Sec) model, emphasizing its potential for (semi-)quantitative analysis through the newly proposed System-Theoretic Process Analysis for Security with Simulations (STPA-Sec/S) approach, illustrated in a water treatment plant case study.

A. DEFINE THE ITEM

The item is the CACC system that consists of a leader vehicle and a follower vehicle that communicate through wireless channels. The item decomposition includes the

sensors, actuators, controllers, and communication modules of each vehicle. The interfaces include the physical and logical connections between the components and the external environment. The data flows include the information exchanged between the components and the external sources. The assets include the safety, security, and performance of the CACC system.

B. IDENTIFY THREAT

The threats to the item can be identified using the STRIDE method, which considers six categories of threats: spoofing, tampering, repudiation, information disclosure, denial of service, and elevation of privilege. For example, a spoofing threat could be an attacker impersonating the leader vehicle and sending false acceleration commands to the follower vehicle. A tampering threat could be an attacker modifying the sensor data or the control signal of the follower vehicle. A repudiation threat could be an attacker denying their involvement in an attack or a vehicle denying its responsibility for an accident. An information disclosure threat could be an attacker eavesdropping on the wireless communication and obtaining sensitive data. A denial of service threat could be an attacker jamming the wireless channel and preventing the communication between the vehicles. An elevation of privilege threat could be an attacker gaining unauthorized access to the CACC system and compromising its functionality. For each threat, a threat level can be assigned based on the likelihood and severity of the threat.

C. ANALYZE RISKS

The risks associated with the identified threats can be analyzed using the DREAD method [45], which considers five criteria of risks: damage potential, reproducibility, exploitability, affected users, and discoverability. For example, a spoofing threat could have a high damage potential if it causes a collision or a loss of control, a high reproducibility if it can be easily repeated by the attacker, a high exploitability if it does not require sophisticated skills or tools, a high affected users if it impacts all the vehicles in the platoon, and a high discoverability if it can be easily detected by the system or the users. For each risk, a risk level can be assigned based on the DREAD criteria. In the case of this paper, risk was quantified from [46] as

$$R = 1 + F \times I \tag{1}$$

where R represents the risk value, F quantifies aggregated attack feasibility rating (Very Low $= 0$, Low $= 1$, Medium $=$ 1.5, High $= 2$), and I denotes impact rating (Negligible $= 0$, Moderate $= 1$, Major $= 1.5$, Severe $= 2$). In the case of this work, impact will be quantified and as shown below

$$2 \times \frac{C}{A} \tag{2}$$

where C and A denote the number of crashes and attacks, respectively, with the whole ratio scaled by two such that the impact is mapped to the same number space as the traditional, statically defined quantity.

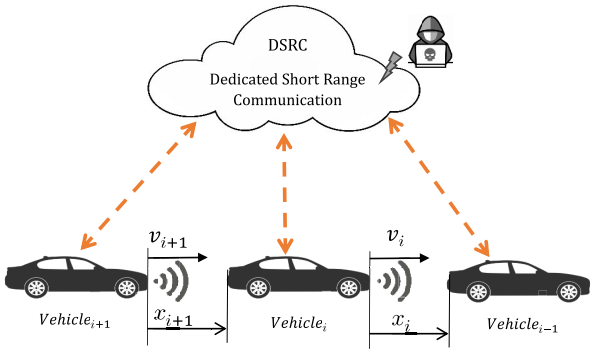


FIGURE 1. CACC schematic of leader-follower structure.

Given the criticality of a CACC system, the impact I rating could considered to be severe as interruption of this feature could result in injury or the loss of life. However, for the purposes of this research, a static I rating would be less helpful for tuning the proposed controller. As such, I was defined as the number of collisions per FDI attack. Attack feasibility is heavily dependent on several factors. In the case of this study, however, it is assumed to be high. This is due to the fact that the test scenarios to be used during validation possess a faulty leader vehicle reference signal.

D. DEFINE SECURITY CONCEPT

The security concept should describe the security mechanisms, security architecture, and security verification and validation methods for the item. For example, the security mechanisms could include encryption, authentication, error detection, frequency hopping, etc. The security architecture could include the security modules, security protocols, security policies, security interfaces, etc. The security verification and validation methods could include security testing, security analysis, security evaluation, security certification, etc. One of the advantages of the novel controller proposed in this paper is streamlining the process of securing the system by preventing erroneous control signal generation by using a Lyapunov-based algorithm that leverages fault detection and estimation. This enables the CACC control policy to focus on maintaining a safe following distance to prevent collision rather than attempting to prevent an attack from occurring.

IV. DYNAMIC MODEL OF CACC UNDER FDI ATTACK

A. DYNAMIC MODEL REPRESENTATION

Figure 1 illustrates a diagram of the CACC string of vehicles. The leader is denoted by $i - 1$, and the n follower vehicles in the platoon are described as $i \in \{1, \ldots, n\}$. The leader vehicle receives the control input as an acceleration command transmitted through the wireless communication channel. Other parameters such as velocity and position of the leader are measured using onboard sensors, such as Radar and GPS. The dynamic model of the vehicle in the platoon

is described as

$$\begin{cases} \dot{x}_i(t) = v_i(t), \\ \dot{v}_i(t) = -\gamma_{1_i} v_i(t) + \gamma_{2_i} u_i(t), \end{cases} \tag{3}$$

where $x_i \in \mathbb{R}$ is the position, $v_i \in \mathbb{R}$ is the velocity, and $u_i \in \mathbb{R}$ is the control input of the follower vehicle. $\gamma_{1_i} \in \mathbb{R}_{>0}$ and $\gamma_{2_i} \in \mathbb{R}_{>0}$ denotes the constant parameters that describe the follower vehicle engine dynamics as derived experimentally.

The Dynamic model of the leader vehicle is described as

$$\begin{cases} \dot{x}_{i-1}(t) = v_{i-1}(t), \\ \dot{v}_{i-1}(t) = -\gamma_{1_{i-1}} v_{i-1}(t) + \gamma_{2_{i-1}} u_{i-1}(t), \end{cases} \tag{4}$$

where $x_{i-1} \in \mathbb{R}$, $v_{i-1} \in \mathbb{R}$, and $u_{i-1} \in \mathbb{R}$ denotes the position, velocity, and control input of the leader vehicle, respectively. The leader vehicle dynamics are defined as $\gamma_{1_{i-1}} \in \mathbb{R}_{>0}$ and $\gamma_{2_{i-1}} \in \mathbb{R}_{>0}$. Since the scenario generator platoon consists of homogeneous vehicles, these parameters are similar to the follower vehicles, γ_{1_i} and γ_{2_i}.

B. FDI ATTACK REPRESENTATION

In FDI attacks, erroneous data is injected into the connected vehicles' communication network to disrupt the performance of the whole system, potentially leading to vehicle collisions. This study focuses on acceleration as the parameter affected by the FDI attack, which is defined as

$$\Pi_i(u_{i-1}(t)) \triangleq u_{i-1}(t) + \beta_i(t), \tag{5}$$

where $\Pi_i \in \mathbb{R}$ is the FDI attack function, $\beta_i(t) \in \mathbb{R}$ is an unknown, bounded, and continuous signal.

Assumption 1: The FDI attack is assumed to be bounded and differentiable such that $\|\beta_i(t)\| \leq \bar{\beta}_i \ \forall t \geq t_0$ where $\bar{\beta}_i$ is a known positive constant [4].

V. CONTROL, OBSERVER, AND ATTACK ESTIMATION DESIGN

A. ERROR SIGNALS

To quantify the performance of the designed controller, let $e_i : [t_0, \infty) \to \mathbb{R}$ be the tracking error between the leader and follower.

$$e_i(t) \triangleq x_i(t) - x_{i-1}(t) + D_i + x_{d_i}(t), \tag{6}$$

where $D_i \in \mathbb{R}$ is the length of vehicle i, and $x_{d_i} : [t_0, \infty) \to \mathbb{R}$ is the desired distance between leader and follower.

Assumption 2: The desired distance, its first, and second derivatives are assumed to be bounded by positive known constants, $x_{d_i}, \dot{x}_{d_i}, \ddot{x}_{d_i} \in \mathcal{L}_\infty$ [47].

To facilitate the design process and stability analysis, an auxiliary error signal is defined as

$$r_i(t) \triangleq \dot{e}_i(t) + \alpha_i e_i(t), \tag{7}$$

such that $\alpha_i \in \mathbb{R}_{>0}$, is a user-specified known gain.

Since the CACC is under FDI attack, there is a need to design an observer. To quantify the accuracy of the observer, a state estimation error $\tilde{x}_{i-1} : [t_0, \infty) \to \mathbb{R}$, is described as

$$\tilde{x}_{i-1}(t) \triangleq x_{i-1}(t) - \hat{x}_{i-1}(t), \tag{8}$$

where $\hat{x}_{i-1} \in \mathbb{R}$ denotes the estimated position of the lead vehicle.

An estimation of the auxiliary error signal $\tilde{r}_{i-1} : [t_0, \infty) \to \mathbb{R}$ is needed for stability analysis and can be defined as

$$\tilde{r}_{i-1}(t) \triangleq \dot{\tilde{x}}_{i-1}(t) + \alpha_{i-1}\tilde{x}_{i-1}(t), \tag{9}$$

where $\alpha_{i-1} \in \mathbb{R}_{>0}$ is a user-defined gain.

We use the observer to estimate the control signal of the leader. To evaluate its accuracy, an estimation error signal $\tilde{u}_{i-1} : [t_0, \infty) \to \mathbb{R}$, is defined as

$$\tilde{u}_{i-1}(t) \triangleq u_{i-1}(t) - \hat{u}_{i-1}(t), \tag{10}$$

where $\hat{u}_{i-1} \in \mathbb{R}$ is the estimated control signal of the leader.

Defining $\bar{u}_{i-1}(t) \triangleq u_{i-1}(t) + \beta_i$ and $\hat{u}_{i-1}(t) \triangleq \bar{u}_{i-1}(t) - \hat{\beta}_i(t)$ yields

$$\tilde{u}_{i-1}(t) = u_{i-1}(t) - \bar{u}_{i-1}(t) + \hat{\beta}_i(t), \tag{11}$$

we consider $\hat{\beta}_i \in \mathbb{R}$ as the estimation of FDI attack and design it such that it remains bounded. Therefore, $\bar{u}_{i-1}(t)$ is bounded and $\|\bar{u}_{i-1}(t)\| \leq \bar{U}_i$ where $\bar{U}_i \in \mathbb{R}_{>0}$.

B. CONTROL DESIGN

The control signal is designed based on the Lyapunov stability analysis as

$$\begin{aligned} u_i(t) \triangleq \frac{\gamma_{1_i}}{\gamma_{2_i}} v_i(t) - \frac{\gamma_{1_i}}{\gamma_{2_i}} v_{i-1}(t) + \bar{u}_{i-1}(t) - \hat{\beta}_i(t) - \frac{1}{\gamma_{2_i}} \ddot{x}_{d_i}(t) \\ - \frac{\alpha_i}{\gamma_{2_i}} r_i(t) - \frac{k_i}{\gamma_{2_i}} r_i(t) - \frac{1}{\gamma_{2_i}} e_i(t) + \frac{\alpha_i^2}{\gamma_{2_i}} e_i(t), \end{aligned} \tag{12}$$

where $k_i \in \mathbb{R}_{>0}$ is a gain specified for the controller that will be further optimized.

Taking the time derivative of (7), and substituting (6) yields the closed loop tracking error of the system as

$$\dot{r}_i(t) = \ddot{x}_i(t) - \ddot{x}_{i-1}(t) + \ddot{x}_{d_i} + \alpha_i \dot{e}_i(t). \tag{13}$$

Replacing $\ddot{x}_i$, $\ddot{x}_{i-1}$ from the model into (13) and substituting $\dot{e}_i(t)$ from (7) yields

$$\begin{aligned} \dot{r}_i(t) = -\gamma_{1_i} v_i(t) + \gamma_{2_i} u_i(t) + \gamma_{1_i} v_{i-1}(t) \\ - \gamma_{2_i} u_{i-1}(t) + \ddot{x}_{d_i}(t) + \alpha_i r_i(t) - \alpha^2 e_i(t). \end{aligned} \tag{14}$$

Adding and subtracting $\gamma_{1_i}\dot{x}_{d_i}$ to and from (14) generates the tracking error term as

$$\begin{aligned} \dot{r}_i(t) = -\gamma_{1_i} \dot{e}_i(t) + \gamma_{2_i} u_i(t) - \gamma_{2_i} u_{i-1}(t) \\ - \beta_i k_i r_i(t) + \ddot{x}_{d_i}(t) + \alpha_i r_i(t) - \alpha^2 e_i(t), \end{aligned} \tag{15}$$

To introduce the effect of FDI attack in the error dynamic, $u_{i-1}(t)$ term is substituted from (11)

$$\begin{aligned} \dot{r}_i(t) = -\gamma_{1_i} \dot{v}_i + \gamma_{2_i} u_i(t) + \gamma_{1_i} v_{i-1}(t) - \gamma_{1_i} \bar{u}_{i-1} \\ + \gamma_{2_i} \beta_i + \ddot{x}_{d_i}(t) + \alpha_i r_i(t) - \alpha_i^2 e_i(t), \end{aligned} \tag{16}$$

Finally, if we place the designed control law for $u_i(t)$ from (12), we can generate the tracking error as

$$\dot{r}_i(t) = -k_i r_i(t) - e_i(t) + \gamma_{2_i}(\beta_i - \hat{\beta}_i). \tag{17}$$

C. OBSERVER AND ESTIMATION DESIGN

Based on the Lyapunov stability analysis, we define the observer as

$$\ddot{\hat{x}}_{i-1}(t) = -\gamma_{1_i} v_{i-1}(t) + \gamma_{2_i} \bar{u}_{i-1}(t) - \gamma_{2_i} \hat{\beta}_i + L_{1_i} \tilde{r}_{i-1}(t) + \alpha_{i-1} \tilde{r}_{i-1}(t) - \alpha_{i-1}^2 \tilde{x}_{i-1}(t) + \dot{\tilde{x}}_{i-1}(t), \quad (18)$$

such that L_{1_i} denoted the observer gain, that will be tuned further. Taking the derivative of (9) with respect to time yields

$$\dot{\tilde{r}}_{i-1}(t) = \ddot{\tilde{x}}_{i-1}(t) + \alpha_{i-1} \dot{\tilde{x}}_{i-1}(t), \quad (19)$$

substituting $\tilde{x}_{i-1}(t)$ from (8), results

$$\dot{\tilde{r}}_{i-1}(t) = -\gamma_{1_i} v_{i-1}(t) + \gamma_{2_i} u_{i-1}(t) + \alpha_{i-1} \tilde{r}_{i-1}(t) - \alpha_{i-1}^2 \tilde{x}_{i-1}(t) - \ddot{\tilde{x}}_{i-1}(t), \quad (20)$$

considering $u_{i-1}(t)$ as in (11) and the observation rule as in (18) and substitute them in (20), we derive tracking error estimation as

$$\dot{\tilde{r}}_{i-1}(t) = -\gamma_{2_i}(\beta_i - \hat{\beta}_i) - L_{1_i} \tilde{r}_{i-1}(t) - \tilde{x}_{i-1}(t). \quad (21)$$

D. FDI ATTACK ESTIMATION

The accuracy of the FDI attack estimation is monitored using an estimation error signal, $\tilde{\beta}_i : [t_0, \infty) \to \mathbb{R}$, defined as

$$\tilde{\beta}_i(t) \triangleq \beta_i(t) - \hat{\beta}_i(t), \quad (22)$$

using Lyapunov stability analysis, we determined the FDI attack estimation as

$$\dot{\hat{\beta}}_i(t) = \gamma_{2_i}(r_i(t) - \tilde{r}_{i-1}(t)). \quad (23)$$

E. STABILITY ANALYSIS

For simplicity in further analysis, the parameter t, which is time, is dropped from the equations. Let's define z_i as

$$z_i \triangleq [e_i^T \; r_i^T \; \tilde{x}_{i-1}^T \; \tilde{r}_{i-1}^T]^T \quad (24)$$

Theorem 1: The controller given in (12), state estimator in (18), and FDI attack estimator in (23) ensure that the z_i converges to zero at $t \to \infty$ in the presence of FDI attack, and the FDI attack estimation error $\tilde{\beta}_i$ is bounded.

Proof: We define Lyapunov candidate function as

$$V_i \triangleq \frac{1}{2} e_i^2 + \frac{1}{2} r_i^2 + \frac{1}{2} \tilde{x}_{i-1}^2 + \frac{1}{2} \tilde{r}_{i-1}^2 + \frac{1}{2} \tilde{\beta}_i^2 \quad (25)$$

where $V_i : \mathbb{R}^n \to \mathbb{R}$ is a continuous positive definite and continuously differentiable function. Taking the time derivative of (25) results

$$\dot{V}_i = e_i \dot{e}_i + r_i \dot{r}_i + \tilde{x}_{i-1} \dot{\tilde{x}}_{i-1} + \tilde{r}_{i-1} \dot{\tilde{r}}_{i-1} + \tilde{\beta}_i \dot{\tilde{\beta}}_i, \quad (26)$$

substituting $\dot{e}_i$, $\dot{r}_i$, $\dot{\tilde{x}}_{i-1}$, and $\dot{\tilde{r}}_{i-1}$ from (7), (17), (9), and (21), respectively, yields

$$\dot{V}_i = e_i(r_i - \alpha_i e_i) + r_i(\gamma_{2_i} \tilde{\beta}_i - k_i r_i - e_i) + \tilde{x}_{i-1}(\tilde{r}_{i-1} - \alpha_{i-1} \tilde{x}_{i-1}) + \tilde{r}_{i-1}(-\gamma_{2_i} \tilde{\beta}_i - L_{1_i} \tilde{r}_{i-1} - \tilde{x}_{i-1}) - \tilde{\beta}_i \dot{\tilde{\beta}}_i, \quad (27)$$

After some simplification, we have

$$\dot{V}_i = -\alpha_i e_i^2 - k_i r_i^2 - \alpha_{i-1} \tilde{x}_{i-1}^2 - L_{1_i} \tilde{r}_{i-1}^2 - \tilde{r}_{i-1} \gamma_{2_i} \tilde{\beta}_i + r_i \gamma_{2_i} \tilde{\beta}_i - \tilde{\beta}_i \dot{\tilde{\beta}}_i, \quad (28)$$

substituting $\dot{\tilde{\beta}}_i$ from (23) the extra terms with $\tilde{\beta}_i$ in (28) cancels. The derivative of Lyapunov candidate function can be written as is

$$\dot{V}_i \leq -\alpha_i \|e_i\|^2 - k_i \|r_i\|^2 - \alpha_{i-1} \|\tilde{x}_{i-1}\|^2 - L_{1_i} \|\tilde{r}_{i-1}\|^2. \quad (29)$$

The $\dot{V}_i$ inequality derived in (29) is a negative semi-definite function since the $\tilde{\beta}_i$ effect cancelled the FDI attack estimation. According to the Lyapunov stability theorem and Barbalat's Lemma [48], we can conclude that there exist α_i, k_i, α_{i-1} and L_{1i} gains such that $z_i \to 0$ as $t \to \infty$ in presence of FDI attacks, and $\tilde{\beta}_i \in \mathcal{L}_\infty$. □

VI. TESTING AND VERIFICATION APPROACH

In [15] and [49], mathematical modeling for the scene and scenario has been introduced. This model is based on the mathematical definition of the scene vector $\mathbf{C}_k \in \mathbb{R}^{n_i}$ that represents the environment surrounding the vehicle or the Unit Under Test (UUT) at a certain time step k. The dimension of scene vector n_i corresponds to the number of parameters used to represent the UUT and the other agents in the environment. The scene vectors form a scene $\mathbf{C}$ with a specific radius. The Scenario is a matrix of consecutive Scene Vectors. To validate the CAV's response using different scenarios, the authors define an assertion function that evaluates the probability of the vehicles passing a set of predetermined weighted assertions. The assertion function is calculated using an assertion matrix $\mathcal{A}$, which is a multi-layer matrix representing the assertion and parameters in the scene vectors. The method allows for flexibility in defining equivalence relations between scenarios, improving the completeness and coverage of testing.

That is for a scene $\mathbf{C}$ the verification cost function $\mathcal{V}$ is defined as

$$\mathcal{V}(C) = \frac{1}{2} \left(\mathcal{M}^{-1} \left(C_{ref} - \mathcal{A}C \right) + \bar{1} \right). \quad (30)$$

The vector denoted by C_{ref} is the reference scene and it is comprised of acceptable parameter values that are automatically generated using a set of rules detailed in [50]. These rules may include the speed limit derived from the road structure input and the minimum safety distances mandated for the assertion. The matrix $\mathcal{M}$ is a diagonal matrix with its entries being the maximum between the reference parameter values from C_{ref} and the corresponding actual values.

To illustrate this, let us assume that a tester intends to asses a CAV's ability to maintain a velocity below the speed limit while maintaining a following distance greater than or equal to the minimum safety distance between itself and another

vehicle. The tester can then compute the following:

$$\mathcal{V}(C) = \frac{1}{2}\left(\begin{bmatrix} v_{max} & 0 & 0 \\ 0 & d_{x,max} & 0 \\ 0 & 0 & d_{y,max} \end{bmatrix}^{-1} \left(\begin{bmatrix} v_{limit} \\ d_{min,x}(t) \\ d_{min,y}(t) \end{bmatrix} - \begin{bmatrix} v \\ d_{1,x}(t) \\ d_{1,y}(t) \end{bmatrix}\right) + \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}\right), \quad (31)$$

where $d_{min,x}(t)$ and $d_{min,y}(t)$ are the minimum latitudinal and longitudinal safe distance between the UUT and the actor. This minimum distance depends on the position and speed of UUT and the actor. The distance between the actor and the UUT defined as $d_{1,x}(t)$ and $d_{1,y}(t)$. Additionally, $v_{max} = \max(v_{limit}, v)$, $d_{x,max} = \max(d_{min,x}, d_{1,x})$ and finally $d_{y,max}$ is defined similarly. The tested unit would *"pass"* the test if the result of (31) is a vector with entries less than 0.5 in all its rows.

This raises an important question about how to improve the operational efficiency of autonomous vehicles. One crucial factor that requires attention relates to enhancing the behavior of CAVs. Note that if a CAV operates at a speed significantly below the prescribed speed limit, such as 10 mph in a 45 mph zone, it may still pass the regulatory tests. However, such a sub-optimal driving approach may not align with the desired outcome.

Therefore, it is crucial to optimize the CAV's responses by minimizing a cost function. It is important to note that all the entries of $\mathcal{V}(C)$ are in the interval of $[0, 1]$. Hence one can now identify "optimized behavior" when the value of entries is closer to 0.5.

VII. PARTICLE SWARM OPTIMIZATION

A. BACKGROUND

PSO is a population-based stochastic optimization technique first proposed in 1995 by Kennedy and Eberhart [37]. This technique takes inspiration from the behaviors of a flock of birds or a school of fish and is an example of evolutionary computing. PSO has been demonstrated to obtain better results in a faster, cheaper way than other methods, partially due to the algorithm's ability to be parallelized [51]. Furthermore, the PSO algorithm possesses very few tuning parameters that are simple and intuitive to understand. These parameters are population size, the number of iterations, particle inertia, inertia damping, and acceleration coefficients.

The first parameter, population size, simply defines the number of particles in the swarm. The next parameter, number of iterations, describes the number of times the PSO algorithm will execute. The swarm size and the number of iterations can be optimized, depending on the number of tunable parameters in the optimization problem [52], [53]. Particle inertia, inertia damping, and acceleration coefficients determine how the particles traverse the problem space. The inertia θ_i at time-step i should be set to a value between 0.9 and 1.2 [54], [55]. Inertia decreases at each iteration proportionate to the inertia damping parameter. The acceleration coefficients, often referred to as C_1 and C_2, allow the user to tune the extent to which each particle will explore the problem space versus exploiting the swarm's knowledge. C_1 tells the swarm how much weight to give to the best position of each individual particle while C_2 defines the weight of the best solution obtained within the swarm. According to another follow-up paper by Eberhart [56], the parameters should be set such that $C_1 + C_2 = \phi$ where $\phi > 4$. However, other more recent studies have suggested the opposite, $\phi < 4$ [57], [58].

In practice, the above parameters may be defined at run-time or even tuned during execution to tune the optimization algorithm. This has given rise to several variations of PSO, which will not be covered in this paper. In addition, for the purposes of this research, the parameters will be set at run-time and change only as an effect of the algorithm's execution.

B. MATHEMATICAL MODEL OF PARTICLE SWARM OPTIMIZATION

In this section, a mathematical model of PSO will be defined and discussed. First, we will formally define the parameters responsible for enabling this algorithm. There are nine (9) parameters in total with several of them being tunable. These tunable parameters are the number of particles in the swarm n, the minimum S_{min} and maximum S_{max} permissible variance, the particle inertia θ_i, and inertia damping ratio ζ, as well as the personal and global learning coefficients.

Swarm size should be defined with respect to the problem space and number of parameters being solved for, so long as n is a positive real integer. The S_{min} and S_{max} parameters are floating-point real numbers that may be negative. These values are utilized for applying bounds to particle velocity. θ_i and ζ determine the future velocities of each particle and, as such, the potential solutions they may find. As the iterations increase, θ_i decreases proportionately to ζ. This is to cause the particles to explore the problem space instead of erratically jumping from point to point.

During each iteration, each particle's velocity is used to determine the next position and is calculated by

$$S_i = \theta_i S_{i-1} + C_1\phi \cdot (\overrightarrow{Pos}_{P_{Best}} - \overrightarrow{Pos}_i) + C_2\phi \cdot (\overrightarrow{Pos}_{G_{Best}} - \overrightarrow{Pos}_i), \quad (32)$$

where $0 < \theta_i < 1$ is the inertia coefficient, S_{i-1} is the previous particle velocity, C_1 and C_2 are the exploration and exploitation coefficients, with $\overrightarrow{Pos}_{P_{Best}}$ and $\overrightarrow{Pos}_i$ being the individual particle's best and current positions while $\overrightarrow{Pos}_{G_{Best}}$ is the swarm's most optimal position.

To prevent a particle's velocity from exceeding permissible bounds,

$$S_i = min(max(S_i, S_{max}), S_{min}), \quad (33)$$

where S_i is the current velocity, S_{max} and S_{min} are the maximum and minimum permissible velocities.

Similarly, Var_{min} and Var_{max} are utilized in a similar way to restrict each particle's position,

$$\overrightarrow{Pos_i} = min(max(\overrightarrow{Pos_i}, \overrightarrow{Pos_{max}}), \overrightarrow{Pos_{min}}), \tag{34}$$

where $\overrightarrow{Pos_i}$ denotes the current position vector, $\overrightarrow{Pos_{min}}$ denotes the minimum position, and $\overrightarrow{Pos_{max}}$ denotes the maximum position.

C. PROBLEM STATEMENT

The objective is to utilize a PSO algorithm, as discussed in Section VII, in the testing and verification approach discussed in Section VI, in order to optimize the verification cost function $\mathcal{V}(C)$ (30). The verification cost will be minimized by tuning the parameter k_i of the controller (12), α_i, and α_{i-1}, and L_i of the observer (9) and (18) such that the preexisting CACC algorithm performs safely while it is under randomly generated FDI attacks, as defined in (5).

In an ideal environment, the controller should be capable of achieving a perfect distance tracking cost of 0.5. However, any cost less than or equal to 0.5 is acceptable as this shows our controller maintains an adequate following distance.

VIII. IMPLEMENTATION

A. EXPERIMENTAL SETUP

To enable our research, MATLAB/Simulink was leveraged for running the CACC algorithm, test scenario generator, and PSO algorithm. The dynamic model for the CACC was obtained through experimental system identification results of a *Ford Fusion S 2016*, illustrated in TABLE 1.

TABLE 1. Constant parameters.

Parameter	Value	Description
γ_{1_i}	0.1413	model gain 1
γ_{2_i}	6.6870	model gain 2

The test scenario utilized in this study was a straight stretch of highway with two actors, a leader and the follower. The follower vehicle is the UUT while the lead vehicle's speed profile is randomly generated using a normal distribution between $0\,m/s$ and $50\,m/s$. Similarly, the FDI attack is randomly generated using a normal distribution ranging from 0% and 99%. The FDI signal is injected during each speed change (every 21.3 seconds). This period between reference changes is derived by 3τ where τ is the time constant of our system, 7.1. Utilizing this delay ensures that, both, the UUT and the lead vehicle can achieve a steady state in response to the speed changes.

The PSO algorithm was implemented as a MATLAB script that iteratively ran the simulation with random initial values for the four (4) parameters, (k_i, α_i, α_{i-1}, L_i) and cumulatively calculated the cost function value simulation as defined in (31).

The swarm, itself, is composed of 50 particles with a permissible variance of (0, 10). The inertia was set to 1 with a damping ratio of 0.99. The acceleration coefficients, C_1 and C_2, were set to 2.5 and 1.5, respectively. The number of iterations executed for optimization was set to 300 as the algorithm was tasked with several parameters with many solvers. While the iterations were set to 300, the algorithm required less than 40 iterations for the swarm to discover an optimal solution. Due to this, a conditional statement was added to terminate the script after the mean cost of the swarm matched the best cost found.

The workflow of our proposed approach is discussed in Algorithm 1.

Algorithm 1 Algorithmic Definition of Optimization Workflow

```
Begin Optimization;
Initialize each particle's position and velocity utilizing a
  uniform random distribution from Var_min to Var_max;
for Number of Iterations do
    for n do
        Calculate velocity as defined in (32);
        Constrain velocity as shown in (33);
        Determine to a new position with respect to the
          current velocity and position;
        Constrain new position using (34);
        Update inertia as θ_{i+1} = θ_i ζ;
        Begin Cost Function
        Utilize the particle's current position as CACC
          parameters in the cost function as defined in (30);
            1) Generate control signal of follower
               from (12);
            2) Calculate estimated acceleration of
               leader from (18);
            3) Estimate the FDI attack from (23);
        Return P_Cost;
    if P_Best < G_Best then
        Best Solution = Current Particle Solution;
Return Best Solution;
```

IX. RESULTS

During experimentation, the workflow was permitted 300 iterations to optimize. However, during several trials, the algorithm settled upon the same optimal solution within 30 iterations. Shown in Table 2 are the global optimal results obtained from the optimization process with the first column $\overline{cost}$ denoting the cost relative to 0, rather than 0.5, and the following columns denoting the values for each controller and observer parameter.

A. OPTIMIZATION RESULTS

First, a comparative analysis of PSO and a genetic algorithm-based (GA) optimization is presented in Figures 2, 3 and 4. The analysis was performed using a 200-second test scenario to expedite the analysis and determine the algorithm that would be selected for the longer scenario. While both algorithms converged upon the same optimal solution, PSO's mean cost converged over 20 iterations sooner than GA's. However, it is worth noting that GA's mean cost decreased more smoothly than PSO which is likely due to the manner in which the algorithms function. Aside from performance, PSO proved to be more intuitive and simpler to implement

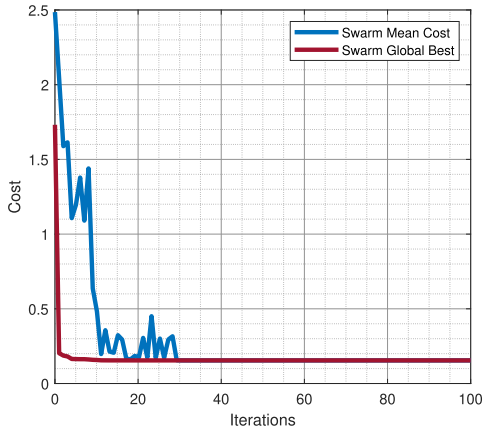


FIGURE 2. PSO algorithm convergence in terms of cost over time.

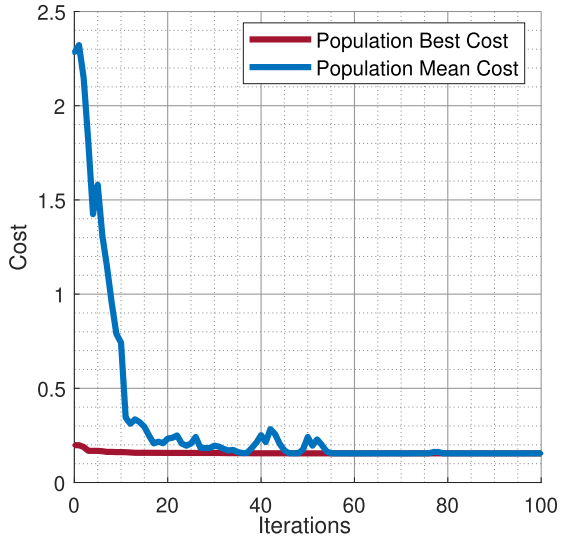


FIGURE 3. Genetic algorithm convergence in terms of cost over time.

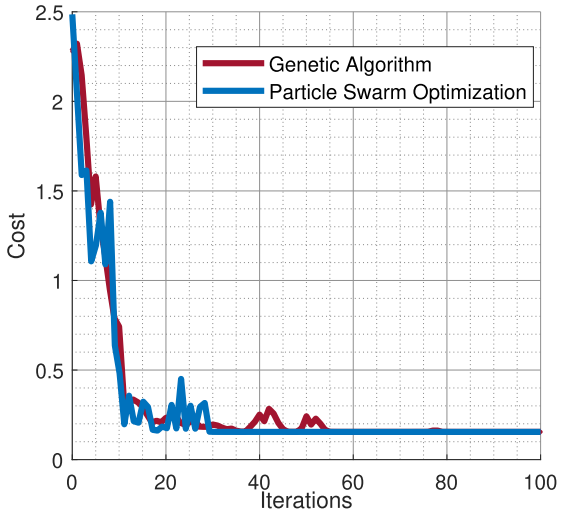


FIGURE 4. Comparative analysis of convergence in terms of cost over time.

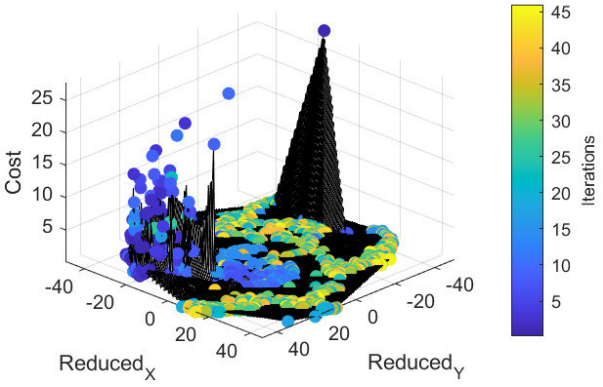


FIGURE 5. Problem space explored by swarm of 50 particles.

from scratch and scale to our use case. Due to these reasons, PSO was selected for integration into our framework.

Next, the results of optimization via PSO are discussed. Shown in Figure 5 is the problem space explored by the 50 particles across 30 iterations, illustrated as a mesh. Specifically, the mesh was created by reducing the dimensions of the particle position data to two dimensions, using the t-SNE MATLAB function. This reduced data serves to define the $xy-plane$ with cost defining displacement along $z-axis$ and each point's shading corresponding with iteration.

As can be seen in Figure 5, the majority of particles start at higher elevations, denoting that their parameter solutions are sub-optimal. The swarm movement progression shows that some particles stumble upon lower-cost solutions within the first few iterations. Due to exploitation, these regions are further explored by other particles. As the iterations increase, the swarm begins to converge upon a relatively contained region of the mesh.

Following the optimization process, the controller was further verified utilizing a longer test scenario. Shown in Table 2 are the baseline and optimal controller configurations with their respective parameter values and performance in terms of cost and the RMSE of FDI estimation, $\tilde{\beta}_i$.

B. DATA ANALYSIS

The optimal configuration presented in Table 2 was then verified using our testing framework. In Figure 6 and 7, the results of the optimal and baseline configurations are plotted over each other. Using the risk formulation from Equation 1, the baseline controller's impact I parameter was found to be over 0.4 while the optimized controller exhibited no impact due to the lack of collisions. Therefore, the risk associated with the baseline controller is 1.4.

Figure 6, demonstrates the FDI attack estimation performance. The true and estimated fault signals from the baseline and optimized controllers has been presented in blue, purple, and orange, respectively.

The following distances of both controllers has been shown in Figure 7. The effects of adversarial conditions and inadequate controller tuning were made apparent by the following distance plot. The baseline configuration possesses

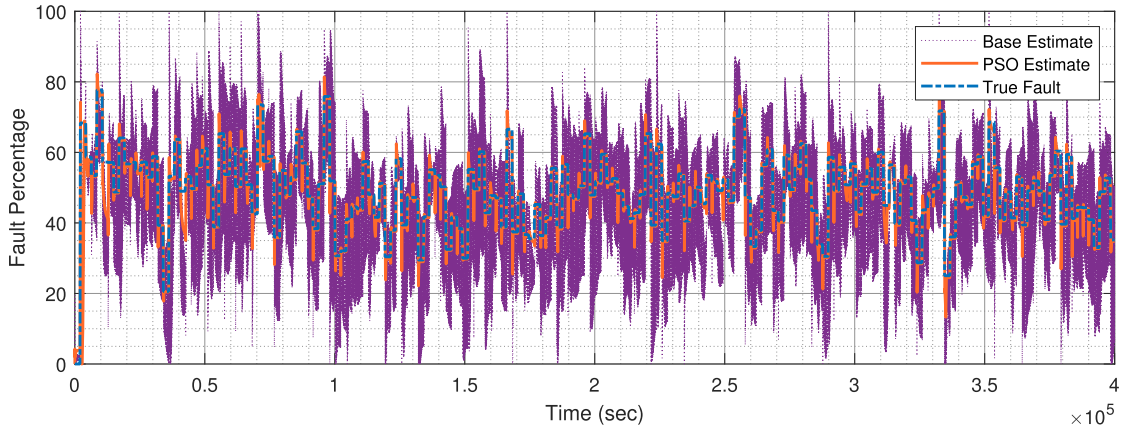


FIGURE 6. FDI attack estimation results using baseline controller and optimized controller.

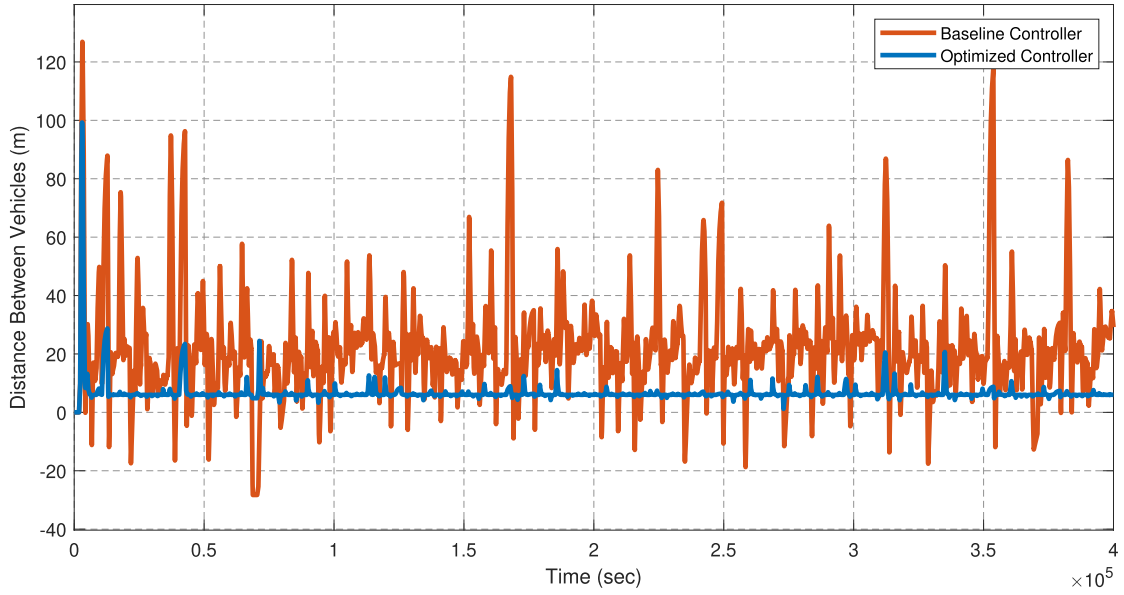


FIGURE 7. Following distance using baseline controller and optimized controller.

TABLE 2. Optimal solutions for resilient CACC parameters.

Config.	$\overline{Cost}$	k_i	α_i	α_{i-1}	L_{1i}	$\tilde{\beta}_i$
Baseline	1.1038	1	1	1	1	15.8953
Optimal	0.065	10	0.3543	0.6372	10	2.3397

a naive definition for the control and FDI estimation parameters, demonstrated by the noisy estimation signal as well as erratic and dangerous following distances. The optimized controller's estimation algorithm adequately tracks and predicts the magnitude of the signal, allowing the controller to compensate. The tuned controller, illustrated in blue, possesses drastically improved performance and adequately prevents collision, unlike the baseline controller that is shown in orange.

X. CONCLUSION

In this paper, we presented a testing and verification approach that has been enhanced with a PSO algorithm to tune the controller and observer parameters of a secure CACC while under FDI attack. The resulting parameter solutions and costs were presented. These parameters were then utilized for defining an optimal controller to be compared against a baseline configuration. After subjecting the controller configurations to testing, it was found that the tuned controller was a drastic improvement over the baseline configuration.

XI. FUTURE WORK

Using our proposed testing and verification framework, further controller tuning will be performed in a ViL environment. In a ViL environment, the testing and tuning process should be completed as quickly and efficiently as possible. Furthermore, due to the nature of testing on a physical system, additional bounds for the parameters are required to prevent damaging the hardware. As such, a novel optimization algorithm will be developed to address these challenges. To accommodate this, our workflow will require

reworking to improve the rate in which optimization iterations are completed as tuning will occur in real-time. Once an algorithm is selected and the workflow revised, we will develop and subject a neural network-based FDI estimator to improve upon the fault detection implemented in this paper.

ACKNOWLEDGMENT

Any opinions, findings, and conclusions, or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the sponsoring agency.

REFERENCES

[1] W. J. Schakel, B. van Arem, and B. D. Netten, "Effects of cooperative adaptive cruise control on traffic flow stability," in *Proc. 13th Int. IEEE Conf. Intell. Transp. Syst.*, 2010, pp. 759–764.

[2] L. Güvenç, I. M. C. Uygan, K. Kahraman, R. Karaahmetoglu, I. Altay, M. Sentürk, M. T. Emirler, A. E. Hartavi Karci, B. Aksun Guvenc, E. Altug, M. C. Turan, Ö. S. Tas, E. Bozkurt, Ü. Ozguner, K. Redmill, A. Kurt, and B. Efendioglu, "Cooperative adaptive cruise control implementation of team mekar at the grand cooperative driving challenge," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 3, pp. 1062–1074, Sep. 2012.

[3] V. Milanés, S. E. Shladover, J. Spring, C. Nowakowski, H. Kawazoe, and M. Nakamura, "Cooperative adaptive cruise control in real traffic situations," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 1, pp. 296–305, Feb. 2014.

[4] A. Sargolzaei, B. C. Allen, C. D. Crane, and W. E. Dixon, "Lyapunov-based control of a nonlinear multiagent system with a time-varying input delay under false-data-injection attacks," *IEEE Trans. Ind. Informat.*, vol. 18, no. 4, pp. 2693–2703, Apr. 2022.

[5] A. Sargolzaei, C. D. Crane, A. Abbaspour, and S. Noei, "A machine learning approach for fault detection in vehicular cyber-physical systems," in *Proc. 15th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2016, pp. 636–640.

[6] X.-M. Zhang, Q.-L. Han, X. Ge, and L. Ding, "Resilient control design based on a sampled-data model for a class of networked control systems under denial-of-service attacks," *IEEE Trans. Cybern.*, vol. 50, no. 8, pp. 3616–3626, Aug. 2020.

[7] A. Abbaspour, A. Sargolzaei, P. Forouzannezhad, K. K. Yen, and A. I. Sarwat, "Resilient control design for load frequency control system under false data injection attacks," *IEEE Trans. Ind. Electron.*, vol. 67, no. 9, pp. 7951–7962, Sep. 2020.

[8] Y. Zhu, D. Zhao, and Z. Zhong, "Adaptive optimal control of heterogeneous CACC system with uncertain dynamics," *IEEE Trans. Control Syst. Technol.*, vol. 27, no. 4, pp. 1772–1779, Jul. 2019.

[9] *Pegasus Project Homepage*. Accessed: Aug. 24, 2023. [Online]. Available: https://www.pegasusprojekt.de/en/about-PEGASUS

[10] *Adaptive Project Homepage*. Accessed: Aug. 24, 2023. [Online]. Available: https://www.adaptive-ip.eu/

[11] P. Koopman, "Edge cases and autonomous vehicle safety," in *Proc. 27th Safety-Critical Syst. Symp. (SSS)*, 2019, pp. 1–25.

[12] D. Zhao, H. Lam, H. Peng, S. Bao, D. J. LeBlanc, K. Nobukawa, and C. S. Pan, "Accelerated evaluation of automated vehicles safety in lane-change scenarios based on importance sampling techniques," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 3, pp. 595–607, Mar. 2017.

[13] L. Castignani, "Simulation in autonomous vehicle development," Repli5, Jul. 2022. [Online]. Available: https://repli5.com/simulation-in-autonomous-vehicles/

[14] N. Kalra and S. M. Paddock, "Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability?" *Transp. Res. A, Policy Practice*, vol. 94, pp. 182–193, 2016.

[15] A. Ala'j, A. Sargolzaei, and M. I. Akbaş, "Autonomous vehicles scenario testing framework and model of computation: On generation and coverage," *IEEE Access*, vol. 9, pp. 60617–60628, 2021.

[16] A. R. Naina Mohamed and A. Singh, "Fuzzy logic based adaptive cruise control for electric vehicles," in *Proc. Int. Conf. Control, Autom. Syst.*, Dec. 2020, pp. 15–20.

[17] Z. Wei, Y. Jiang, X. Liao, X. Qi, Z. Wang, G. Wu, P. Hao, and M. Barth, "End-to-end vision-based adaptive cruise control (ACC) using deep reinforcement learning," 2020, *arXiv:2001.09181*.

[18] Z. Yang, Z. Wang, and M. Yan, "An optimization design of adaptive cruise control system based on MPC and ADRC," *Actuators*, vol. 10, p. 110, 2021.

[19] D. L. Luu, C. Lupu, H. Alshareefi, and H. T. Pham, "Coordinated throttle and brake control for adaptive cruise control strategy design," in *Proc. 23rd Int. Conf. Control Syst. Comput. Sci. (CSCS)*, 2021, pp. 9–14.

[20] S. Gong, A. Zhou, and S. Peeta, "Cooperative adaptive cruise control for a platoon of connected and autonomous vehicles considering dynamic information flow topology," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 2673, no. 10, pp. 185–198, 2019.

[21] S. Tamilarasan and L. Guvenc, "Impact of different desired velocity profiles and controller gains on convoy driveability of cooperative adaptive cruise control operated platoons," 2023, *arXiv:2306.01971*.

[22] B. Wang, D. Zhao, C. Li, and Y. Dai, "Design and implementation of an adaptive cruise control system based on supervised actor-critic learning," in *Proc. 5th Int. Conf. Inf. Sci. Technol. (ICIST)*, Apr. 2015, pp. 243–248.

[23] K. Osman, Mohd. F. Rahmat, and M. A. Ahmad, "Modelling and controller design for a cruise control system," in *Proc. 5th Int. Colloq. Signal Process. Its Appl.*, Mar. 2009, pp. 254–258.

[24] P. Ioannou and Z. Xu, "Throttle and brake control systems for automatic vehicle following," *I V H S J.*, vol. 1, no. 4, pp. 345–377, 1994.

[25] S. Yu, X. Pan, A. Georgiou, B. Chen, I. M. Jaimoukha, and S. A. Evangelou, "A real-time robust ecological-adaptive cruise control strategy for battery electric vehicles," 2023, *arXiv:2308.01201*.

[26] J. Bélanger et al., "The what, where and why of real-time simulation," *Planet Rt*, vol. 1, no. 1, pp. 25–29, 2010.

[27] O. Gietelink, J. Ploeg, B. De Schutter, and M. Verhaegen, "Development of advanced driver assistance systems with vehicle hardware-in-the-loop simulations," *Vehicle Syst. Dyn.*, vol. 44, no. 7, pp. 569–590, Jul. 2006.

[28] P. Wang, Y. Zhou, Q. Luo, C. Han, Y. Niu, and M. Lei, "Complex-valued encoding metaheuristic optimization algorithm: A comprehensive survey," *Neurocomputing*, vol. 407, pp. 313–342, Sep. 2020.

[29] A. Gogna and A. Tayal, "Metaheuristics: Review and application," *J. Experim. Theor. Artif. Intell.*, vol. 25, no. 4, pp. 503–526, Dec. 2013.

[30] S. E. De León-Aldaco, H. Calleja, and J. A. Alquicira, "Metaheuristic optimization methods applied to power converters: A review," *IEEE Trans. Power Electron.*, vol. 30, no. 12, pp. 6791–6803, Dec. 2015.

[31] H. Yiğit, S. Ürgün, and S. Mirjalili, "Comparison of recent metaheuristic optimization algorithms to solve the SHE optimization problem in MLI," *Neural Comput. Appl.*, vol. 35, no. 10, pp. 7369–7388, Apr. 2023.

[32] A. Lambora, K. Gupta, and K. Chopra, "Genetic algorithm—A literature review," in *Proc. Int. Conf. Mach. Learn., Big Data, Cloud Parallel Comput. (COMITCon)*, Feb. 2019, pp. 380–384.

[33] H. Zhou and J. Qiao, "Multiobjective optimal control for wastewater treatment process using adaptive MOEA/D," *Int. J. Speech Technol.*, vol. 49, no. 3, pp. 1098–1126, Mar. 2019.

[34] Y. Zhu, Y. Qin, D. Yang, H. Xu, and H. Zhou, "An enhanced decomposition-based multi-objective evolutionary algorithm with a self-organizing collaborative scheme," *Exp. Syst. Appl.*, vol. 213, Mar. 2023, Art. no. 118915.

[35] H. Zhou, Y. Li, Q. Zhang, H. Xu, and Y. Su, "Soft-sensing of effluent total phosphorus using adaptive recurrent fuzzy neural network with Gustafson-Kessel clustering," *Exp. Syst. Appl.*, vol. 203, Oct. 2022, Art. no. 117589.

[36] S. Akyol and B. Alatas, "Plant intelligence based metaheuristic optimization algorithms," *Artif. Intell. Rev.*, vol. 47, no. 4, pp. 417–462, Apr. 2017.

[37] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. ICNN-Int. Conf. Neural Netw.*, vol. 4, 1995, pp. 1942–1948.

[38] D. W. Boeringer and D. H. Werner, "Particle swarm optimization versus genetic algorithms for phased array synthesis," *IEEE Trans. Antennas Propag.*, vol. 52, no. 3, pp. 771–779, Mar. 2004.

[39] M. Mahak and Y. Singh, "Threat modelling and risk assessment in Internet of Things: A review," in *Proc. 2nd Int. Conf. Comput., Commun., Cyber-Secur.* Singapore: Springer, P. K. Singh, S. T. Wierzchoń, S. Tanwar, M. Ganzha, and J. J. P. C. Rodrigues, Eds. 2021, pp. 293–305.

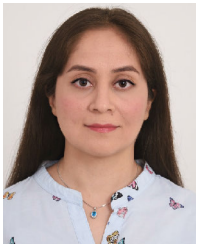
[40] N. P. de Souza, C. D. A. C. César, J. D. M. Bezerra, and C. M. Hirata, "Extending STPA with STRIDE to identify cybersecurity loss scenarios," *J. Inf. Secur. Appl.*, vol. 55, Dec. 2020, Art. no. 102620.

[41] *ISO/SAE 21434:2021—Road Vehicles–Cybersecurity Engineering*, International Organization for Standardization (ISO) and Society of Automotive Engineers (SAE), ISO and SAE, Standard ISO/SAE 21434:2021, 2021.

[42] Z. Abuabed, A. Alsadeh, and A. Taweel, "STRIDE threat model-based framework for assessing the vulnerabilities of modern vehicles," *Comput. Secur.*, vol. 133, Oct. 2023, Art. no. 103391.

[43] S. Ghosh, A. Zaboli, J. Hong, and J. Kwon, "An integrated approach of threat analysis for autonomous vehicles perception system," *IEEE Access*, vol. 11, pp. 14752–14777, 2023.
[44] F. Simone, A. J. N. Akel, G. D. Gravio, and R. Patriarca, "Thinking in systems, sifting through simulations: A way ahead for cyber resilience assessment," *IEEE Access*, vol. 11, pp. 11430–11450, 2023.
[45] S. Hussain, A. Kamal, S. Ahmad, G. Rasool, and S. Iqbal, "Threat modelling methodologies: A survey," *Sci. Int. (Lahore)*, vol. 26, no. 4, pp. 1607–1609, 2014.
[46] Y. Wang, Y. Wang, H. Qin, H. Ji, Y. Zhang, and J. Wang, "A systematic risk assessment framework of automotive cybersecurity," *Automot. Innov.*, vol. 4, no. 3, pp. 253–261, Aug. 2021.
[47] P. M. Patre, W. MacKunis, K. Kaiser, and W. E. Dixon, "Asymptotic tracking for uncertain dynamic systems via a multilayer neural network feedforward and RISE feedback control structure," *IEEE Trans. Autom. Control*, vol. 53, no. 9, pp. 2180–2185, Oct. 2008.
[48] K. S. Narendra and A. M. Annaswamy, *Stable Adaptive Systems*. North Chelmsford, MA, USA: Courier Corporation, 2012.
[49] A. J. Alnaser, M. I. Akbas, A. Sargolzaei, and R. Razdan, "Autonomous vehicles scenario testing framework and model of computation," *SAE Int. J. Connected Automated Vehicles*, vol. 2, no. 4, pp. 205–218, Dec. 2019.
[50] S. Shalev-Shwartz, S. Shammah, and A. Shashua, "On a formal model of safe and scalable self-driving cars," 2017, *arXiv:1708.06374*.
[51] A. G. Gad, "Particle swarm optimization algorithm and its applications: A systematic review," *Arch. Comput. Methods Eng.*, vol. 29, no. 5, pp. 2531–2561, Aug. 2022.
[52] A. P. Piotrowski, J. J. Napiorkowski, and A. E. Piotrowska, "Population size in particle swarm optimization," *Swarm Evol. Comput.*, vol. 58, Nov. 2020, Art. no. 100718.
[53] L. Xueyan and X. Zheng, "Swarm size and inertia weight selection of particle swarm optimizer in system identification," in *Proc. 4th Int. Conf. Comput. Sci. Netw. Technol. (ICCSNT)*, vol. 1, Dec. 2015, pp. 1554–1556.
[54] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Proc. IEEE Int. Conf. Evol. Comput.*, 1998, pp. 69–73.
[55] S. Wang, F. Zhou, and F. Wang, "Effect of inertia weight ω on PSO-SA algorithm," *Int. J. Online Eng.*, vol. 9, pp. 87–91, Jun. 2013.
[56] R. C. Eberhart and Y. Shi, "Comparing inertia weights and constriction factors in particle swarm optimization," in *Proc. Congr. Evol. Computation. CEC00*, vol. 1, 2000, pp. 84–88.
[57] M. S. Innocente and J. Sienz, "Coefficients' settings in particle swarm optimization: Insight and guidelines," 2021, *arXiv:2101.11944*.
[58] P. Sun, H. Sun, W. Feng, Q. Zhao, and H. Zhao, "A study of acceleration coefficients in particle swarm optimization algorithm based on CPSO," in *Proc. 2nd Int. Conf. Inf. Eng. Comput. Sci.*, 2010, pp. 1–4.

JAMES C. HOLLAND (Graduate Student Member, IEEE) received the Bachelor of Science degree in computer science and the master's degree in mechanical engineering from Tennessee Technological University. He is currently pursuing the Ph.D. degree in mechanical engineering with the University of South Florida. He joined the RANCS Research Group as an Undergraduate Student in freshman year. His interest in both hardware and software is of no surprise to those who know him. For as long as he can remember, he has been an avid tinkerer, exploring the intricacies of technology, whether it be an old motherboard or an antique car. His research interests include machine learning, cybersecurity, and testing and verification of connected and autonomous vehicles.



FARAHNAZ JAVIDI-NIROUMAND received the bachelor's and master's degrees in electrical engineering. She is currently pursuing the Ph.D. degree in mechanical engineering with the University of South Florida (USF). She is passionate about enhancing control systems' security and reliability using machine learning tools. She is also experienced in designing and implementing control for unmanned aerial vehicles. As a future goal, she is seeking to implement a secure and reliable cooperative formation between connected autonomous vehicles and drones. Her research interests include the security of multi-agent systems, adaptive control, nonlinear control, and industrial automation.

ALA' J. ALNASER received the bachelor's degree in applied mathematics and statistics from the Jordan University of Science and Technology, and the master's and Ph.D. degrees in mathematics from Kansas State University, in 2006 and 2009, respectively. He completed the Ph.D. dissertation in algebraic number theory and developed new methods for finding upper bounds for waring's number over the rational integers. He has over 17 years of experience in academia to his position as an Assistant Professor of applied mathematics with Florida Polytechnic University. Previously, he was an Assistant Professor of mathematics with Trine University, IN, USA, from January 2010 to May 2015. His current research focus is on mathematical dynamical and stochastic modeling, control theory, and analytical methods.

ARMAN SARGOLZAEI (Senior Member, IEEE) received the first M.S. and first Ph.D. degrees in electrical engineering from Florida International University, Miami, FL, USA, in 2012 and 2015, respectively, and the second master's degree in aerospace engineering and the second Ph.D. degree in mechanical engineering from the University of Florida, Gainesville, FL, in 2019 and 2020, respectively. He currently holds the position of an Assistant Professor with the Department of Mechanical Engineering, University of South Florida (USF). Prior to joining USF, he was an Assistant Professor of mechanical engineering with Tennessee Technological University. He was also an Assistant Professor of electrical engineering and the Director of the Advanced Mobility Institute, Florida Polytechnic University. His mission is to enhance the quality of life for people, with assuring safety, security, and privacy through extensive collaboration among multi-disciplinary fields. He was a recipient of the NSF CAREER Award, in 2022. He was recognized with the honor of "2017 Faculty Research Excellence" and the "2018 Faculty Research Excellence" Award.