2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 23-27, 2022, Kyoto, Japan

Resilient Detection and Recovery of Autonomous Systems Operating under On-board Controller Cyber Attacks

Paul J Bonczek and Nicola Bezzo

Abstract— **Cyber-attacks, failures, and implementation errors inside the controller of an autonomous system can affect its correct behavior leading to unsafe states and degraded performance. In this paper, we focus on such problems specifically on cyber-attacks that manipulate controller parameters like the gains in a feedback controller or that triggers different behaviors or block inputs based on specific values of the state and tracking error. If such attacks are undetected, they can lead to the partial or complete loss of system's control authority, resulting in a hijacking and leading the autonomous system towards unforeseen states. To deal with this problem, we propose a runtime monitoring and recovery scheme in which: 1) we leverage the residual between the expected and the received measurements to detect inconsistencies in the generated inputs and 2) provide a recovery method for counteracting the malicious effects to allow for resilient operations by manipulating the reference signal and state vector provided to the system to avoid the affected regions in the state and error space. We validate our approach with Matlab simulations and experiments on unmanned ground vehicles resiliently performing operations in the presence of malicious attacks to on-board controllers.**

I. Introduction

Present-day autonomous robotic systems possess increased complexities to support an expanded array of computers and sensors to assist in advanced capabilities such as navigation, warehouse logistics, and industrial operations, towards truly unmanned operations. With such complexity, however, comes higher risks of malicious cyber attacks due to their unsupervised, autonomous applications and the numerous entry points to implement an attack.

While the vast majority of the literature in robotics and cyber-physical systems security deal primarily with attacks on the sensing and communication infrastructure of a system [1], in this work we consider attacks that interfere with the control logic to hijack a system. For this class of attacks, controller parameter gains can be altered to trigger an undesirable behavior under certain states or tracking errors. For example, in Fig. 1 shows a motivational case in which a robot needs to turn to the right but ends up turning to the left, away from the desired trajectory and into an obstacle when the tracking error is within a compromised region.

One of the key principles that we leverage in this work is that such robotic systems, in nominal conditions, i.e., when uncompromised, have well-designed dynamical models that enable accurate predictions of output measurements from their control input signals. A cyber attack on the on-board controller can cause inconsistencies from the expectation of this input-output model, leading to observable deviations from its nominal behavior. To this end, we consider a residual-based monitoring approach that leverages the chi-squared detection scheme to reduce the residual vector into a scalar test measure to detect controller integrity inconsistencies. Regions of the state space or the tracking error space that are deemed compromised are monitored for future operations to avoid them. A compensator to alter information provided to the controller (i.e., reference signal and state vector) is built to avoid any compromised regions within the state or tracking error spaces. Moreover, to deal with this problem of maintaining desirable control performance during operations, the altered information is designed to minimize the difference in the *compensated* control input signal in comparison to the originally intended (but compromised) control input.

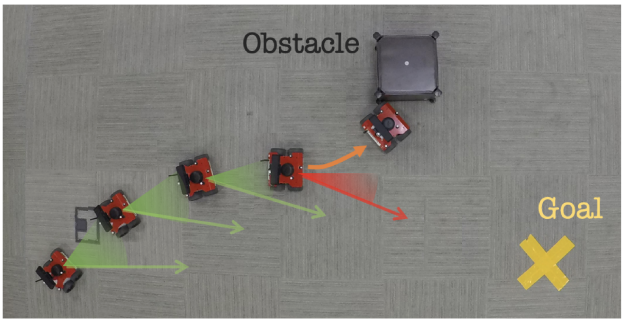


Fig. 1. Pictorial representation of the problem investigated in this paper in which a robot is tasked to navigate toward a goal under a controller attack that gets activated only when the tracking error crosses a certain threshold (red region in the figure).

The contribution of this work is twofold: 1) a detection framework to discover compromised regions of the state or tracking error space within a controller that cause anomalous system behavior and 2) a compensator that alters the reference signal and state information provided to the controller in order to bypass compromised regions to achieve desired control performance to resiliently continue operations.

A. Related Literature

The subject of security in autonomous systems has received significant attention within the robotics and control communities. Various well-studied areas within security have presented frameworks to provide resilience to vulnerable access points on a system that are susceptible to cyber attacks that include: sensors, actuators, communication, and on-board controllers [1]–[4].

A classic example of controller integrity issues are data injection attacks described in [5], where the intended control signal is replaced with an undesired signal by malicious attackers. The authors characterize the detection limitations of such attacks and quantified the performance degradation

Paul J Bonczek and Nicola Bezzo are with the Charles L. Brown Department of Electrical and Computer Engineering, and members of the Link Lab, University of Virginia, Charlottesville, VA, USA. Email: {pjb4xn,nb6be}@virginia.edu

2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) | 978-1-6654-7927-1/22/$31.00 ©2022 IEEE | DOI: 10.1109/IROS47612.2022.9981844

impacts. In [6], a reference governor-based defense mechanism that utilizes pseudorandom vectors was proposed to detect malicious setpoints (i.e., references) received from a command center to affect tracking performance. Authors in [7] proposed a framework where an attacker gains knowledge (i.e., obtain an estimate) of a reference signal within the on-board controller by manipulating sensor measurements. The proposed approach in [8] leveraged a bank of residual-based monitors to detect when a malicious attacker implements data injections attacks or has switched to a different control model within a hybrid controller. Furthermore, encryption techniques have been used to secure on-board controllers. For example, a dynamic management approach for key switching using public and private keys can be used as in [9] to detect falsified inputs and replay attacks. Furthermore, homomorphic encryption techniques have been used to protect on-board linear controllers [10] and state estimators [11].

Differing from the works previously mentioned, we design a framework to detect and recover from anomalous system behavior triggered by specific conditions of information provided to the on-board controller. We assume that anomalous behavior can be caused by: i) malicious attackers that deliberately change control parameters and/or inject false data to the control signal and ii) unintentional software related issues (i.e., fault and code bugs) that compute undesired control input signals for the autonomous system [12].

The remainder of this work is organized as follows: We introduce the system and threat models in Section II and the problem formulation in Section III. In Section IV we characterize the attack detection method within the error and/or state space, then describe the approach for system recovery. In Section V we present simulations and experiments on unmanned ground vehicles (UGVs) to validate our framework. Finally, in Section VI we summarize our results and discuss possible future directions for this work.

II. Preliminaries

This work considers robotic systems modeled as discrete-time linear time-invariant (LTI) systems of the form:

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{u}_k + \boldsymbol{\nu}_k, \tag{1}$$

$$\boldsymbol{y}_k = \boldsymbol{C}\boldsymbol{x}_k + \boldsymbol{\eta}_k, \tag{2}$$

where $\boldsymbol{x}_k \in \mathbb{R}^n$ denotes the state vector, $\boldsymbol{u}_k \in \mathbb{R}^m$ is the control input, and $\boldsymbol{y}_k \in \mathbb{R}^s$ represents the output vector at every time instance $k \in \mathbb{N}$. The state, input, and output matrices $\boldsymbol{A}$, $\boldsymbol{B}$, and $\boldsymbol{C}$ are of appropriate dimensions, while $\boldsymbol{\nu}_k \sim \mathcal{N}(0, \boldsymbol{Q}) \in \mathbb{R}^n$ and $\boldsymbol{\eta}_k \sim \mathcal{N}(0, \boldsymbol{R}) \in \mathbb{R}^s$ are i.i.d. Gaussian process and measurement uncertainties.

During operations, a Kalman Filter (KF) is implemented to provide a state estimate $\hat{\boldsymbol{x}}_k \in \mathbb{R}^n$ in the form:

$$\hat{\boldsymbol{x}}_{k+1} = \boldsymbol{A}\hat{\boldsymbol{x}}_k + \boldsymbol{B}\boldsymbol{u}_k + \boldsymbol{L}(\boldsymbol{y}_k - \boldsymbol{C}\hat{\boldsymbol{x}}_k) \tag{3}$$

where $\boldsymbol{L} = \boldsymbol{P}\boldsymbol{C}^\mathsf{T}(\boldsymbol{C}\boldsymbol{P}\boldsymbol{C}^\mathsf{T} + \boldsymbol{R})^{-1}$ is defined as the Kalman gain matrix which is solved by the algebraic Riccati equation.

A. Threat Model

We assume a general feedback controller with a nominal control input signal that is described as

$$\boldsymbol{u}_k = \boldsymbol{K}\left(\boldsymbol{x}_k^{\text{ref}} - \hat{\boldsymbol{x}}_k\right) = \boldsymbol{K}\boldsymbol{x}_k^e \tag{4}$$

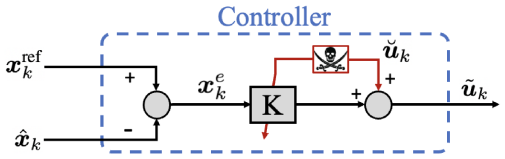


Fig. 2. A diagram describing altered control parameters and additive inputs to result in undesired control behavior.

where $\boldsymbol{x}_k^e = \boldsymbol{x}_k^{\text{ref}} - \hat{\boldsymbol{x}}_k$ is the tracking error between a reference signal (i.e., desired state) and the state estimate, while $\boldsymbol{K}$ is a feedback gain to provide desired control performance of the system. Additionally, we assume the true control inputs to the system

$$-\boldsymbol{u}_{max} \leq \boldsymbol{u}_k \leq \boldsymbol{u}_{max} \tag{5}$$

are constrained to due to actuation limitations.

We consider control inputs (4) that can be altered due to undesired (and unknown) changes in controller parameters and/or additive inputs, as depicted in Fig. 2. These changes occur as signals fed to the controller satisfy specific compromised ranges of tracking error $\widetilde{\mathcal{E}}$ and state $\widetilde{\mathcal{X}}$ within a finite tracking error space $\widetilde{\mathcal{E}} \subset \mathcal{E}$ and/or finite state space $\widetilde{\mathcal{X}} \subset \mathcal{X}$. Within these compromised regions, we consider scenarios such as: i) cyber attacks that are able to maliciously modify control parameters and/or introduce control signal biases at runtime or ii) faulty code that is defined before operations begin; resulting in undesirable control inputs provided to the system.

The altered control inputs $\tilde{\boldsymbol{u}}_k \neq \boldsymbol{u}_k$ are presented as

$$\tilde{\boldsymbol{u}}_k = \widetilde{\boldsymbol{K}}\boldsymbol{x}_k^e + \breve{\boldsymbol{u}}_k \in \mathbb{R}^m \tag{6}$$

with the feedback gain $\widetilde{\boldsymbol{K}} \neq \boldsymbol{K}$ and additive input signal $\breve{\boldsymbol{u}}_k \in \mathbb{R}^m$ when the following condition is satisfied:

$$\boldsymbol{x}_k^e \in \widetilde{\mathcal{E}}, \quad \hat{\boldsymbol{x}}_k \in \widetilde{\mathcal{X}}. \tag{7}$$

III. Problem Formulation

An attack or fault to an on-board controller will consequently result in anomalous behavior due to unreliable control signals $\tilde{\boldsymbol{u}}_k$ being applied to the system. In this work, we focus on discovering specific operating conditions (i.e., regions and ranges) from the information (i.e., reference signal $\boldsymbol{x}_k^{\text{ref}}$ and state estimate $\hat{\boldsymbol{x}}_k$) provided to the controller that cause undesired behaviors.

Problem 1 (Anomalous Behavior Detection): Given the nominal and altered control inputs represented in (4) and (6), we want to detect compromised regions within the state $\mathcal{X}$ and error $\mathcal{E}$ spaces. Formally, the objective is to find conditions of the reference signal and state information

$$\boldsymbol{u}_k = \begin{cases} \widetilde{\boldsymbol{K}}\boldsymbol{x}_k^e + \breve{\boldsymbol{u}}_k, & \text{if } \{\boldsymbol{x}_k^{\text{ref}}, \hat{\boldsymbol{x}}_k\} \in \widetilde{\mathcal{X}}, \widetilde{\mathcal{E}} \\ \boldsymbol{K}\boldsymbol{x}_k^e, & \text{otherwise} \end{cases} \tag{8}$$

that trigger undesired control inputs which are sent to the robot, hence resulting in undesired system behavior.

Upon detection of compromised regions of state $\widetilde{\mathcal{X}} \subset \mathcal{X}$ and tracking error $\widetilde{\mathcal{E}} \subset \mathcal{E}$ spaces, the robot aims to avoid triggering these undesired behaviors such that resilient operation can continue. Formally:

Problem 2 (System Recovery): Design a policy such that the robot computes a *compensated* reference signal $\bar{\boldsymbol{x}}_k^{\text{ref}}$ and state $\bar{\boldsymbol{x}}_k$ information for the controller, where $\bar{\boldsymbol{x}}_k^{\text{ref}} \neq \boldsymbol{x}_k^{\text{ref}}$ and $\bar{\boldsymbol{x}}_k \neq \hat{\boldsymbol{x}}_k$, in order to avoid malicious regions within the state and error spaces to maintain desirable control performance. Furthermore, the *compensated input* $\bar{\boldsymbol{u}}_k$ which is computed using the compensated reference signal and state information, seeks to minimize the following:

$$\bar{\boldsymbol{u}}_k = \left\{ \arg\min_{\bar{\boldsymbol{u}}_k} (\bar{\boldsymbol{u}}_k - \boldsymbol{u}_k^*) : \{\bar{\boldsymbol{x}}_k^{\text{ref}}, \bar{\boldsymbol{x}}_k\} \notin \widetilde{\mathcal{X}}, \widetilde{\mathcal{E}} \right\} \tag{9}$$

where $\boldsymbol{u}_k^*$ is the *desired* control input before compensation and $\arg\min(\bar{\boldsymbol{u}}_k - \boldsymbol{u}_k^*)$ represents the objective to find a compensated input with minimum difference from the desired.

IV. Framework

In this section we describe the monitoring and recovery framework to detect anomalous controller behavior during specific ranges of information, then implement a recovery mechanism for an autonomous robot to provide uncompromised control inputs for motion. The overall control architecture is highlighted in Fig. 3 where a detector monitors the residual vector to determine if anomalous system behavior is occurring. This allows for compensation of information into the controller (i.e., reference signal and state estimate vector) to ensure uncompromised control inputs are sent to the robot. Our focus is on attacks or faults taking place on the on-board controller as specific ranges of input information fed to the controller are provided. Within these ranges, unknown controller parameter changes and/or additive control signals are included to the control input, causing undesired behavior of the robot.

A. Space Partitioning

In this work, we want to discover specific regions within the error and state spaces that may be compromised due to cyber attacks (or possibly faulty code) that alters the control inputs computed by the on-board controller. To monitor for compromised regions within tracking error space $\mathcal{E} = \{\mathcal{E}_1, \ldots, \mathcal{E}_n\}$ or the state space $\mathcal{X} = \{\mathcal{X}_1, \ldots, \mathcal{X}_n\}$ for an ith state, $i = 1, \ldots, n$, we first partition the spaces into a finite number of bins.

For generalization, lets define any given space by the set $\mathcal{S} = \{\mathcal{S}_1, \ldots, \mathcal{S}_n\}$. An ith state in space $\mathcal{S}_i$ to be monitored is partitioned into N_b bins to check for inconsistent behavior within each bin (i.e., partitioned region). The set of $j = 1, \ldots, N_b$ partitioned bins in an ith state are represented as $\mathcal{B}_i = \{b_{i,1}, \ldots, b_{i,j}, \ldots, b_{i,N_b}\}$ that span the entire space. Each partitioned bin of arbitrary size is a subset of the set of bins (i.e., $b_{i,j} \subset \mathcal{B}_i$) within a space $\mathcal{S}_i$ and described by:

$$b_{i,j} = \begin{cases} \mathcal{S}_{i,\min} \leq b_{i,j} \leq \mathcal{S}_{i,(b_{i,j,\max})} & \text{if } j = 1 \\ \mathcal{S}_{i,(b_{i,j-1,\max})} < b_{i,j} \leq \mathcal{S}_{i,(b_{i,j,\max})} & \text{if } j = 2, \ldots, N_b - 1 \\ \mathcal{S}_{i,(b_{i,j-1,\max})} < b_{i,j} \leq \mathcal{S}_{i,\max} & \text{if } j = N_b \end{cases} \tag{10}$$

such that

$$\bigcup_{j=1}^{N_b} b_{i,j} = \mathcal{S}_i \quad \text{and} \quad \bigcap_{j=1}^{N_b} b_{i,j} = \emptyset \tag{11}$$

are satisfied.

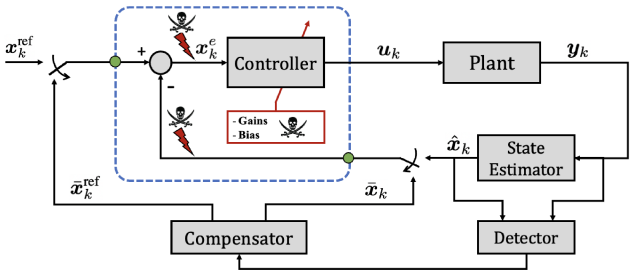


Fig. 3. Overall detection and compensation architecture of our framework. The compensator manipulates the reference and state estimate vectors to compute a compensated input $\bar{\boldsymbol{u}}_k$ when $\hat{\boldsymbol{x}}_k \in \widetilde{\mathcal{X}}$ or $\hat{\boldsymbol{x}}_k^e \in \widetilde{\mathcal{E}}$ are satisfied.

The objective is to discover if specific regions within a space lead to faulty/anomalous behavior due to a compromised on-board controller providing unknown, malicious control signals to the system. For ease in the remainder of this paper, we describe the bin subspaces of a state i in general terms that may be utilized for either the tracking error or state space. At every time iteration, the system determines which bin j the information that is monitored belongs to (i.e., tracking error $x_{k,i}^e \in b_{i,j}$ or state $\hat{x}_{k,i} \in b_{i,j}$). In the next subsection, we describe how to monitor for anomalous behavior within each bin $b_{i,j}$ in an ith state or error space.

B. Residual-based State and Error Consistency Monitoring

During an operation, the robot checks for anomalous system behavior due to attacks or faults within the on-board controller. In particular, each error/state subspace (i.e., denoted by the partitioned bins described in Section IV-A) is monitored as the conditions of the reference and state estimate information sent to the controller are met. If specific subspaces during operations present anomalous behavior, they are flagged as compromised regions. We leverage a residual-based fault detection scheme to check for anomalous system behavior at runtime. The idea behind this scheme is to utilize the state predictions $\hat{\boldsymbol{x}}_{k+1}$ in (3) to determine if the system is responding to the computed control inputs $\boldsymbol{u}_k$ accordingly. To monitor for inconsistent system behavior, we compute the measurement residual vector

$$\boldsymbol{r}_k = \boldsymbol{y}_k - \boldsymbol{C}\hat{\boldsymbol{x}}_k \in \mathbb{R}^s \tag{12}$$

which is defined as the difference between sensor measurements and the state prediction that was computed at the previous time $k-1$. The measurement residual is modeled by a zero-mean Gaussian distribution $\boldsymbol{r}_k = \mathcal{N}(0, \boldsymbol{\Sigma})$ with an expected covariance matrix $\boldsymbol{\Sigma} = \mathbb{E}[\boldsymbol{r}_k \boldsymbol{r}_k^{\mathsf{T}}] = \boldsymbol{C}\boldsymbol{P}\boldsymbol{C}^{\mathsf{T}} + \boldsymbol{R} \in \mathbb{R}^{s \times s}$, where $\boldsymbol{P}$ is the estimation error covariance matrix. A system that is behaving in a consistent manner will display residuals that follow this expected distribution, while misbehaving systems violate this expectation. We utilize the well-known chi-square detection scheme by reducing the residual vector into a scalar test measure [13]:

$$z_k = \boldsymbol{r}_k^{\mathsf{T}} \boldsymbol{\Sigma}^{-1} \boldsymbol{r}_k \tag{13}$$

which is chi-squared distributed that follows $z_k \sim \chi(s)$.

To determine if undesired control inputs $\widetilde{\boldsymbol{u}}_k$ are being computed by the compromised controller in a specific bin $b_{i,j} \subset \mathcal{B}_i$, we monitor for expected behavior of the test

measure. Similar to our previous work [14], we monitor for unexpected sign switching rates to detect inconsistent behavior. The sign of each incoming test measure (13) with respect to a user-defined reference value $z^{\text{ref}} \in \mathbb{R}_{>0}$ is computed at every time k. Moreover, the signed test measure value $\text{sgn}(z_k - z^{\text{ref}})$ is compared with the sign of the previous signed test measure value when the system belonged to the same jth bin in state i, $b_{i,j} \subset \mathcal{B}_i$, at time a $k - T_{b_{i,j}}$. If the test measure comparison is of opposite signs, then an alarm is triggered, otherwise an alarm is not triggered. Formally, the procedure to trigger an alarm follows:

$$\zeta_k = \begin{cases} 1, & \text{if } \text{sgn}(z_k - z^{\text{ref}}) = -\text{sgn}(z_{k-T_{b_{i,j}}} - z^{\text{ref}}) \\ 0, & \text{otherwise} \end{cases} \tag{14}$$

where $T_{b_{i,j}} \in \mathbb{N}$ denotes the number of time steps since the jth bin in the space $\mathcal{B}_i$ was entered. The alarm $\zeta_k \in \{0, 1\}$ signifies that the test measure at time k is of the opposite sign (i.e., a sign switch) with respect to the previous test measure within the same bin at time $k - T_{b_{i,j}}$ to trigger an alarm $\zeta_k = 1$, otherwise $\zeta_k = 0$. The alarm ζ_k is placed into a runtime alarm rate estimator:

$$\hat{A}_{k|b_{i,j'}} = \begin{cases} \hat{A}_{k-1|b_{i,j'}} + \frac{\zeta_k - \hat{A}_{k-1|b_{i,j'}}}{\ell} & \text{if } j' = j \\ \hat{A}_{k-1|b_{i,j'}} & \text{if } j' \neq j \end{cases} \tag{15}$$

to compute an updated estimate for the jth bin within the set $\mathcal{B}_i$, where ℓ is a "pseudo-window" length. All other alarm rate estimates corresponding to a bin $j' \neq j$ are carried over from the previous time step, since they are unaffected as the system did not belong to the space pertaining to the j'th bin. All alarm rate estimates $\forall j \in b_{i,j}$ are initialized to $\hat{A}_{0|b_{i,j}} = \mathbb{E}[A]$ at time $k = 0$, and alarm rate estimates should follow

$$\hat{A}_k = \mathbb{E}[A] = \mathbb{P}[\zeta_k = 1] \tag{16}$$

where $\mathbb{E}[A]$ is the expected rate which alarms are triggered. Bounds on the estimated alarm rates, denoted by $[\Omega_-, \Omega_+] \in (0, 1)$ that satisfies $\Omega_- < \mathbb{E}[A] < \Omega_+$, can be computed to signify anomalous system behavior, which is discussed in greater detail in [14]. To summarize, when the alarm (i.e., sign switching) rate for the detection of anomalous controller behavior satisfies,

$$\hat{A}_{k|b_{i,j}} \notin [\Omega_-, \Omega_+] \longrightarrow \textit{Anomalous Behavior} \tag{17}$$

the robot detects anomalous behavior within bin j on an ith state. The robot then places the bin j presenting the anomalous behavior into a compromised bin set $b_{i,j} \rightarrow \widetilde{\mathcal{B}}_i$, where $\widetilde{\mathcal{B}}_i \subset \mathcal{B}_i$, to allow for compensation to avoid this region that results in undesired control performance.

C. State and Reference Compensation

The goal for the compensator is to provide compensated values of the reference signal $\bar{\boldsymbol{x}}_k^{\text{ref}}$ and state $\bar{\boldsymbol{x}}_k$ to the controller in order to avoid any compromised regions within the error $\widetilde{\mathcal{E}}$ and/or state $\widetilde{\mathcal{X}}$ space. We define the control input

$$\bar{\boldsymbol{u}}_k = \boldsymbol{K}\left(\bar{\boldsymbol{x}}_k^{\text{ref}} - \bar{\boldsymbol{x}}_k\right) = \boldsymbol{K}\bar{\boldsymbol{x}}_k^e \tag{18}$$

as the computed input that utilizes the compensated reference and state vectors. Additionally, when providing compensated

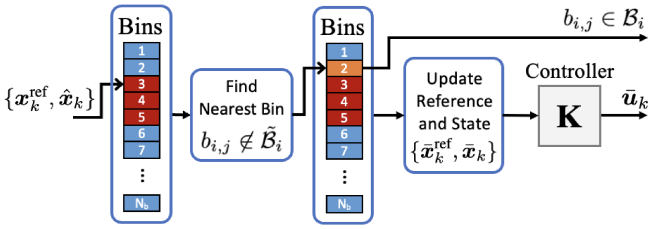


Fig. 4. The compensation process to provide altered references and states to the controller to avoid compromised regions (red) within an ith statee.

information $\{\bar{x}_{k,i}^{\text{ref}}, \bar{x}_{k,i}\}$ on an ith state to the controller, our objective is to update the information in a manner to minimize the difference in compensated control input signal from the nominal (i.e., desired) input

$$\bar{\boldsymbol{u}}_k = \arg\min\left(\bar{\boldsymbol{u}}_k - \boldsymbol{u}_k^*\right) = \arg\min\left(\boldsymbol{K}(\bar{\boldsymbol{x}}_k^e - \boldsymbol{x}_k^e)\right) \tag{19}$$

where $\bar{\boldsymbol{x}}_k^e = \bar{\boldsymbol{x}}_k^{\text{ref}} - \bar{\boldsymbol{x}}_k$ and $\boldsymbol{x}_k^e = \boldsymbol{x}_k^{\text{ref}} - \hat{\boldsymbol{x}}_k$.

In Fig. 4 we show a high-level view of the compensation approach to satisfy (19). The compensator determines whether the incoming information (i.e., the reference $x_{k,i}^{\text{ref}}$ and state estimate $\hat{x}_{k,i}$) belong to any compromised bins $\widetilde{\mathcal{B}}_i \subset \mathcal{B}_i$ (highlighted by the red bins in Fig. 4) within an ith state/error space. If the information belongs to a compromised bin, then the compensator chooses the nearest uncompromised bin (colored in orange) from the current state/error, which is leveraged to compute a compensated reference signal $\bar{x}_{k,i}^{\text{ref}}$ and state estimate $\bar{x}_{k,i}$, respectively. Next, we provide the compensation procedure for the cases when an attack to the controller affects regions within the state $\widetilde{\mathcal{X}}_i \subset \mathcal{X}_i$ and tracking error $\widetilde{\mathcal{E}}_i \subset \mathcal{E}_i$ spaces.

1) Compromised State Space: In this subsection, we describe compensation that occurs as one or more bins within the state space $\mathcal{X}_i$ of any ith state are deemed compromised. The compensation occurs only when the ith state estimate element is within a compromised jth bin $\hat{x}_{k,i} \in b_{i,j}$ where $b_{i,j} \subset \widetilde{\mathcal{B}}_i$. The objective is to find a compensated state

$$\bar{x}_{k,i} = \{\bar{x}_{k,i} \in b_{i,j} \,:\, \min(\hat{x}_{k,i} - \bar{x}_{k,i}), b_{i,j} \in \mathcal{B}_i \setminus \widetilde{\mathcal{B}}_i\} \tag{20}$$

that is provided to the controller which belongs to an uncompromised bin $b_{i,j} \subset \mathcal{B}_i \setminus \widetilde{\mathcal{B}}_i$. To maintain desired reference tracking performance, we also compensate the reference signal $\bar{x}_{k,i}^{\text{ref}}$ by the same amount as the state compensation

$$\bar{x}_{k,i}^{\text{ref}} = x_{k,i}^{\text{ref}} + \Delta x_{k,i} \tag{21}$$

such that the tracking error remains unchanged, where $\Delta x_{k,i} = (\bar{x}_{k,i} - \hat{x}_{k,i})$ is the change in state.

Lemma 1: Given the compensated state estimation and reference signal that is provided to the controller to avoid compromised regions in an ith state space $\bar{x}_{k,i} \notin \widetilde{\mathcal{X}}_i$, the computed control input $\bar{\boldsymbol{u}}_k$ is equal to the desired input $\boldsymbol{u}_k^*$.

Proof: We observe that the difference in tracking error

$$(\bar{x}_{k,i}^{\text{ref}} - \bar{x}_{k,i}) - (x_{k,i}^{\text{ref}} - \hat{x}_{k,i}) = \bar{x}_{k,i}^e - x_{k,i}^e = 0 \tag{22}$$

between the compensated and uncompensated information of an ith state that is provided to the controller is zero. The resulting tracking error vector $\bar{\boldsymbol{x}}_k^e = \boldsymbol{x}_k^e$ remains unchanged, thus the control input signal to the system is the same (i.e., $\boldsymbol{K}\bar{\boldsymbol{x}}_k^e = \boldsymbol{K}\boldsymbol{x}_k^e \rightarrow \bar{\boldsymbol{u}}_k = \boldsymbol{u}_k^*$). ■

2) Compromised Error Space: Next, we describe the scenario where compensation occurs to avoid regions that are within the compromised tracking error space which result in undesired control performance of the system. Similar to the previous subsection for anomalies in the state space, we characterize the compensation effort that occurs as anomalous behaviors are detected within any jth bin of the error space $x^e_{k,i} = x^{\text{ref}}_{k,i} - \hat{x}_{k,i} \in b_{i,j}$, where $b_{i,j} \subseteq \widetilde{\mathcal{B}}_i$. However, attackers may reduce the usable tracking errors to a much smaller subset of the original error space $\mathcal{E}_i$.

Depending on feasible control inputs (5) for a given system (1), we can describe the tracking error space for these suitable states as a *loop*, as depicted in Fig. 5. In particular, reference signals for these states can be switched to the opposite sign (i.e., reverse direction) to enable the robot to reach specific positions within the environment. Furthermore, a user-defined *buffer* region between bins 1 and N_b may be included to any suitable ith error element whose state can leverage the *loop* method to control when a switching (i.e., reversal) of reference signals is applied. In Fig. 6, we provide two examples of this method for both the velocity and heading angle of a robot. The examples depict: 1) a robot navigating with a negative velocity (i.e., velocity reference signal is of the opposite sign) and 2) a robot turning in the opposite direction by creating a loop (i.e., heading angle reference is shifted by $\pm 2\pi$). These scenarios can be exploited such that a robot can reach desired positions within the environment when certain control input conditions cannot be attained due to compromised regions within the error space. If the current tracking error satisfies $x^e_{k,i} \in b_{i,j} \subset \widetilde{\mathcal{B}}_i$ and the nearest tracking error $\breve{x}^e_{k,i} \neq x^e_{k,i}$ within an uncompromised bin

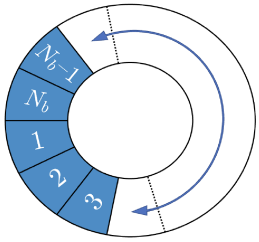


Fig. 5. Viewing bins within an ith error space as a loop.

$$\breve{x}^e_{k,i} = \arg\min(x^e_{k,i} - \breve{x}^e_{k,i}), \quad \breve{x}^e_{k,i} \in b_{i,j} \subset \mathcal{B}_i \setminus \widetilde{\mathcal{B}}_i \tag{23}$$

crosses over the buffer region (i.e., $1 \to N_b$ or $N_b \to 1$), we update the ith reference signal to the opposite direction with the function $f : \mathbb{R} \mapsto \mathbb{R}$ defined by

$$\bar{x}^{\text{ref}}_{k,i} = f(x^{\text{ref}}_{k,i}) \Longrightarrow x^e_{k,i} = \bar{x}^{\text{ref}}_{k,i} - \hat{x}_{k,i} \tag{24}$$

and then the tracking error $x^e_{k,i}$ is updated accordingly.

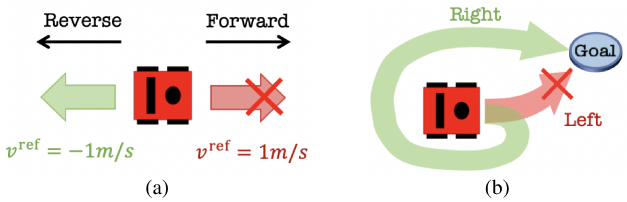


Fig. 6. Examples of system states such as (a) velocity and (b) heading angle that are capable of providing an "opposite" reference signal.

From the given tracking error $x^e_{k,i}$, the objective is to find the nearest tracking error to provide to the controller

$$\bar{x}^e_{k,i} = \{\bar{x}^e_{k,i} \in b_{i,j} : \arg\min_{\bar{x}^e_{k,i}}(x^e_{k,i} - \bar{x}^e_{k,i}), b_{i,j} \subset \mathcal{B}_i \setminus \widetilde{\mathcal{B}}_i\} \tag{25}$$

which belongs to an uncompromised bin $b_{i,j} \subset \mathcal{B}_i \setminus \widetilde{\mathcal{B}}_i$ such that $\bar{x}^e_{k,i} \notin \widetilde{\mathcal{E}}_i$. We compensate the reference signal $\bar{x}^{\text{ref}}_{k,i}$ by

$$\bar{x}^{\text{ref}}_{k,i} = \hat{x}_{k,i} + \bar{x}^e_{k,i} \tag{26}$$

to achieve the compensated tracking error in (25).

After compensation for an ith reference signal has occurred, the compensated control input is no longer equal to the desired control input $\bar{\boldsymbol{u}}_k \neq \boldsymbol{u}^*_k$ as $\boldsymbol{K}\bar{\boldsymbol{x}}^e_k \neq \boldsymbol{K}\boldsymbol{x}^e_k$. The goal is to find a feasible solution to update reference signals for any state $i' \neq i$, $i' = \{1, \ldots, n\} \setminus i$ to minimize the difference between the compensated control input $\bar{\boldsymbol{u}}_k$ with compensated reference signals and the desired control input $\boldsymbol{u}^*_k$. To minimize the difference in control input to the system, the following objective function is solved

$$J(\bar{\boldsymbol{x}}^e_k) = \arg\min_{\check{\boldsymbol{x}}^e_k}\left(\|\check{\boldsymbol{K}}\check{\boldsymbol{x}}^e_k - \boldsymbol{u}^*_k + \boldsymbol{K}_i\bar{x}^e_{k,i}\|\right) \tag{27}$$

where $\check{\boldsymbol{K}}$ is the feedback control matrix with the ith column removed, $\check{\boldsymbol{x}}^e_k$ is the tracking error vector with the ith element removed, and $\boldsymbol{K}_i$ is the ith column in the feedback matrix $\boldsymbol{K}$. To satisfy the compensated tracking error vector in (27) that minimizes the change in the control input, we compensate the reference signals elements i' for any altered tracking error

$$\bar{x}^{\text{ref}}_{k,i'} = x^{\text{ref}}_{k,i'} + (\bar{x}^e_{k,i'} - x^e_{k,i'}), \quad \forall i' \neq i. \tag{28}$$

The following Lemma provides details to show that the true system state converges to the desired reference signal, even as the reference signal is being compensated.

Lemma 2 (System Stability): Given the compensated reference signals in (26) and (28) to minimize (27) in order to avoid any compromised regions within the tracking error space $\bar{\boldsymbol{x}}^e_k \notin \widetilde{\mathcal{E}}$ with control input $\bar{\boldsymbol{u}}_k = \boldsymbol{K}\bar{\boldsymbol{x}}^e_k$, the reference tracking closed-loop system is asymptotically stable.

Proof: We omit the full proof due to page limitations. However, Lemma 2 can be proved via Lyapunov stability to show that reference tracking is globally asymptotically stable such that the system state converges toward any desired bounded reference signal $\boldsymbol{x}^{\text{ref}}_k$ during attacks/faults to the tracking error space (i.e., $\widetilde{\mathcal{E}} \neq \emptyset$). In other words, the expectation of the true tracking error $\boldsymbol{x}^t_k = \boldsymbol{x}^{\text{ref}}_k - \boldsymbol{x}_k$ is always converging (i.e., $\mathbb{E}[\boldsymbol{x}^t_{k+1} - \boldsymbol{x}^t_k] \to 0$ as $k \to \infty$) for any compensated reference signal $\bar{\boldsymbol{x}}^{\text{ref}}_k$ and compensated state $\bar{\boldsymbol{x}}_k$ computed in (24)–(28). ■

We note that when the compensator is providing compensated signals to the controller $\{\bar{\boldsymbol{x}}^{\text{ref}}_k, \bar{\boldsymbol{x}}_k\} \to \bar{\boldsymbol{u}}_k$, the resulting computed compensated control input $\bar{\boldsymbol{u}}_k$ is utilized in the state estimation process in (3).

V. Results

Our framework is validated with Matlab simulations and lab experiments using a Husarion Rosbot 2.0 robot that performs go-to-goal operations. To highlight the generality and applicability of our proposed scheme, we consider attacks that maliciously affect the velocity and heading angle

of the robots. In both case studies, the robots are subject to on-board controller attacks that maliciously alter control parameters and/or adds control input biases with the intention to degrade system performance. Videos for the simulation and experiments are provided in the supplemental material.

A. Simulation

We present here one of the simulation case studies that considers a differential drive UGV dynamical model [15]:

$$\begin{aligned} \dot{v} &= \frac{1}{m}(F_l + F_r - B_r v), \\ \dot{\omega} &= \frac{1}{I_z}\Big(\frac{w}{2}(F_l - F_r) - B_l \omega\Big), \ \dot{\theta} = \omega, \\ \dot{x} &= v\cos(\theta), \quad \dot{y} = v\sin(\theta), \end{aligned} \tag{29}$$

where v, θ, and ω denote velocity, vehicle heading angle, and angular velocity, along with the position coordinates x and y to form the state vector $\boldsymbol{x} = [x, y, v, \theta, \omega]^\mathsf{T}$. F_l and F_r describe the left and right input forces from the wheels, w is the vehicle width, while B_r and B_l are resistances due to the wheels rolling and turning. We perform linearization and assume the sensor sampling rate is $t_s = 0.05$s to satisfy the system model in (1) and (2).

In the simulation, the robot is tasked to visit a series of waypoints (i.e., goals) within an obstacle-filled environment while maintaining a velocity $v^{\text{ref}} = 0.15$m/s. We present a scenario where attacks occur within the error space when velocity information provided to the controller is within a compromised region $\hat{v}_k^e \in \widetilde{\mathcal{E}}$. Three simulations are provided that highlight the *Nominal* (i.e., no attack), *Uncorrected*, and *Corrected* scenarios where attacks start at $k = 2700$ and the robot begins from the same initial state $\boldsymbol{x}_0 = [0, 1.5, 0, 0, 0]^\mathsf{T}$.

In Fig. 7, we highlight the case study where attacks occur as the velocity tracking error is any positive value, causing feedback gains for velocity to be reduced by 90% and an input bias of $\breve{\boldsymbol{u}}_k = -0.08$. Shown in Fig. 7(a), when not monitoring for malicious behavior in the *Uncorrected* scenario, the robot continues to attempt a forward (positive) velocity and is driven away from its next intended goal point. In the *Corrected* scenario, the robot discovers anomalous behavior in bins $b_{i,j} \subset \widetilde{\mathcal{B}}_i$ corresponding to positive velocity tracking errors (Fig. 7(c)). This allows the robot to recover by updating its reference velocity to the opposite direction to navigate the environment in reverse to resiliently maintain the operation.

B. Experiments

Experimental validations are implemented on Husarion Rosbot 2.0 robots performing a go-to-goal operation within a lab environment. We show two case studies where attacks are triggered based on information from the heading angle this time; the first with an attack scenario within the state space and the second within the error space. For each experiment case study, we provide results where the robot is: a) unprotected from attacks/faults, b) protected from attacks/faults while leveraging our compensation framework for recovery, and c) a Matlab representation of the robot positions during both the unprotected and protected experiments.

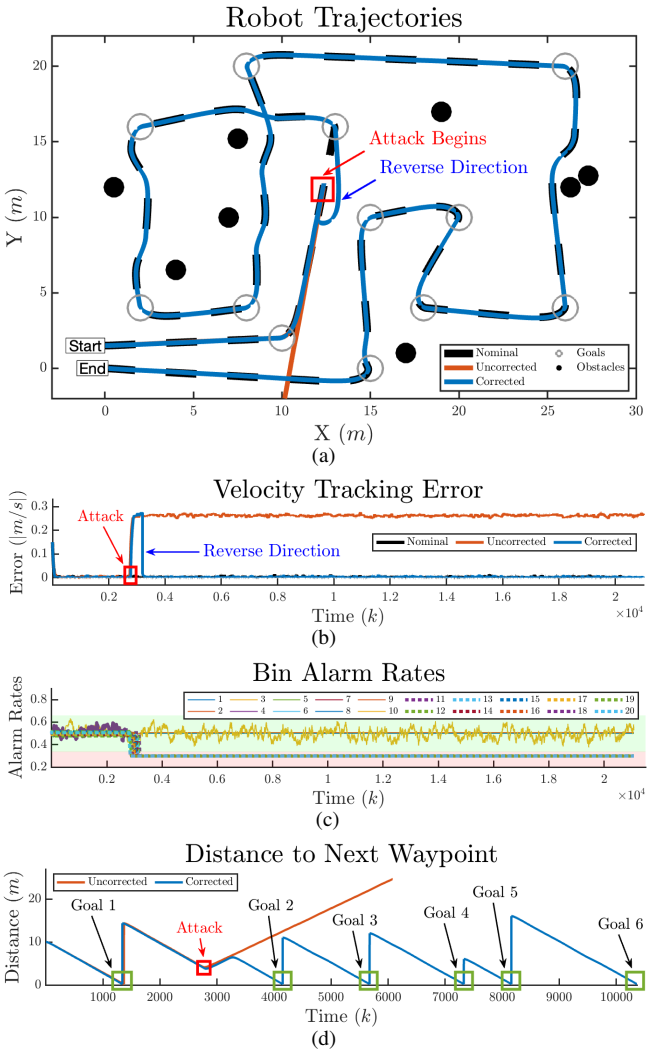
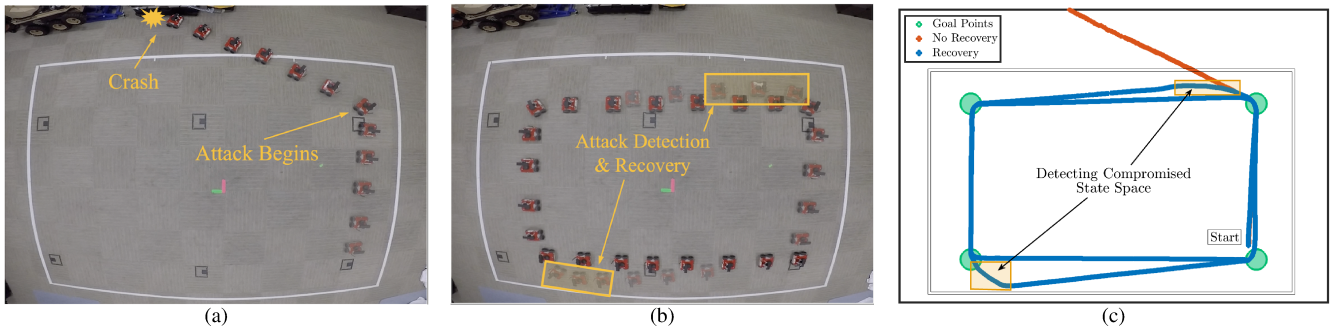


Fig. 7. The compromised tracking error space case study displaying: (a) the resulting robot trajectories, (b) velocity tracking error, (c) alarm rates for the $N_b = 20$ bins, and (d) distance to the next goal/waypoint.

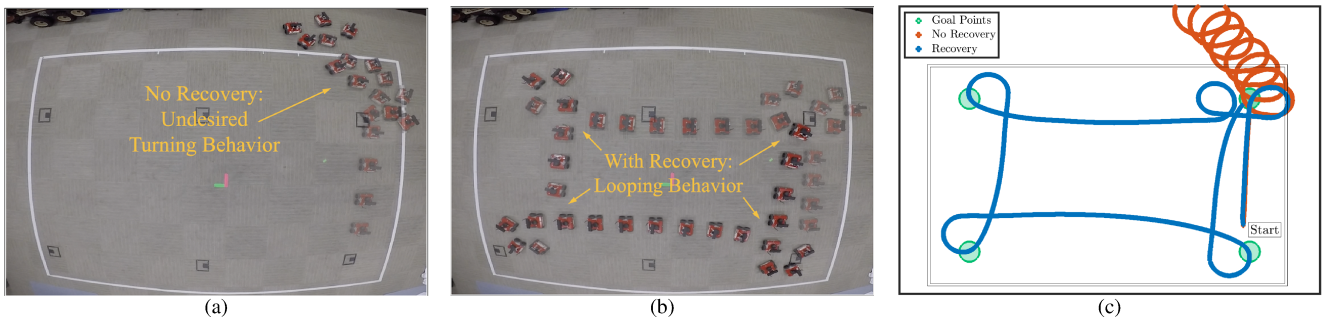
Snapshots of the first case study experiment are presented in Fig. 8, which captures the robot navigating to a series of goals. The malicious threat to the controller occurs when the robot heading angle (in degrees) satisfies $\theta \in [140, 245]$ or $\theta \in [-75, 30]$. Without our detection and recovery being performed, shown in Fig. 8(a), the control signal to the system is compromised and eventually the robot crashes into a wall. We see in Fig. 8(b) where our framework is leveraged, the robot compensates the information to the controller upon discovering anomalous regions within the state space to allow for continued navigation to each of the goal points.

Our second case study demonstrates the robot that is subject to attacks in the heading angle error space, as shown in Fig. 9. The attack occurs at any instance the robot desires to turn left (i.e., negative tracking error), but instead the attack causes a turning action to the right. When our framework is not implemented (Fig. 9(a)), the robot fails to complete the operation due to continued circling motion. As shown in Fig. 9(b), since the robot is not able to turn left, the compensator alters the reference signal such that the robot performs a "looping" action by always turning right (i.e.,



(a) (b) (c)

Fig. 8. An experiment showing a robot detecting anomalous behavior due to an attack within the state space of the heading angle estimate $\hat{\theta}$, then compensating information provided to the controller to avoid any compromised states $\hat{\theta} \in \mathcal{X}_i \setminus \widetilde{\mathcal{X}}_i$.



(a) (b) (c)

Fig. 9. A robot resiliently navigates to a series of waypoints while control parameters corresponding to angular velocity are altered during an attack within the error space. The attacks occur as the robot's tracking error for heading angle is negative (i.e., desired turn to the left).

positive tracking error) as this is the only possible action that the robot can do to avoid the compromised tracking error space to continue the operation.

VI. Conclusion & Future Work

In this paper we have proposed a detection and recovery framework for autonomous mobile robots to maintain uncompromised control actions to resiliently perform a desired operation. The robot is able to identify attacks/faults to its on-board controller within specific regions of the state and tracking error spaces by leveraging a residual-based attack detection scheme within the partitioned spaces. Furthermore, the robot uses a compensator to alter reference signal and state estimation information which is fed into the controller to maintain desired performance while avoiding any compromised regions in the state/tracking-error spaces. Our future plans include extending the current framework to trigger replanning operations (e.g., changes in mission goals and trajectories) in the event that control recovery is not possible. Further implementation on different vehicles such as aerial robots and multi-robot systems are also in our agenda.

Acknowledgments

This work is based on research sponsored by DARPA under Contract No. FA8750-18-C-0090 and NSF under grant number #1816591

References

[1] F. Sommer, J. Dürrwang, and R. Kriesten, "Survey and classification of automotive security attacks," *Information*, vol. 10, no. 4, 2019.

[2] N. Bezzo, J. Weimer, M. Pajic, O. Sokolsky, G. J. Pappas, and I. Lee, "Attack resilient state estimation for autonomous robotic systems," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2014, pp. 3692–3698.

[3] K. Saulnier, D. Saldana, A. Prorok, G. J. Pappas, and V. Kumar, "Resilient flocking for mobile robot teams," *IEEE Robotics and Automation letters*, vol. 2, no. 2, pp. 1039–1046, 2017.

[4] E. Mousavinejad, F. Yang, Q.-L. Han, and L. Vlacic, "A novel cyber attack detection method in networked control systems," *IEEE Transactions on Cybernetics*, vol. 48, no. 11, pp. 3254–3264, 2018.

[5] C.-Z. Bai, F. Pasqualetti, and V. Gupta, "Data-injection attacks in stochastic control systems: Detectability and performance tradeoffs," *Automatica*, vol. 82, pp. 251–260, 2017.

[6] W. Lucia, K. Gheitasi, and M. Ghaderi, "Setpoint attack detection in cyber-physical systems," *IEEE Transactions on Automatic Control*, vol. 66, no. 5, pp. 2332–2338, 2021.

[7] D. Umsonst and H. Sandberg, "On the confidentiality of the reference signal under sensor attacks," in *2021 60th IEEE Conference on Decision and Control (CDC)*, 2021, pp. 3468–3473.

[8] D. Sun and I. Hwang, "Resilient control design for hybrid systems against switching and data injection attacks," in *2019 IEEE 58th Conference on Decision and Control (CDC)*, 2019, pp. 3854–3859.

[9] K. Kogiso, "Attack detection and prevention for encrypted control systems by application of switching-key management," in *2018 IEEE Conference on Decision and Control (CDC)*, 2018, pp. 5032–5037.

[10] ——, "Encrypted control using multiplicative homomorphic encryption," in *Privacy in Dynamical Systems*. Springer, 2020, pp. 267–286.

[11] Z. Zhang, P. Cheng, J. Wu, and J. Chen, "Secure state estimation using hybrid homomorphic encryption scheme," *IEEE Transactions on Control Systems Technology*, vol. 29, no. 4, pp. 1704–1720, 2021.

[12] P. Koopman and M. Wagner, "Challenges in autonomous vehicle testing and validation," *SAE International Journal of Transportation Safety*, vol. 4, no. 1, pp. 15–24, 2016.

[13] P. J. Bonczek and N. Bezzo, "Detection of hidden attacks on cyber-physical systems from serial magnitude and sign randomness inconsistencies," in *2021 American Control Conference (ACC)*, 2021, pp. 3281–3287.

[14] ——, "Detection and inference of randomness-based behavior for resilient multi-vehicle coordinated operations," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 5844–5850.

[15] J. J. Nutaro, *Building software for simulation: theory and algorithms, with applications in C++*. John Wiley & Sons, 2011.