Pinpointing Actuator Attacks: A Novel Diagnostic Framework for Cyber-Physical Systems

Zifan Wang
Syracuse University
zwang345@syr.edu

Qinru Qiu
Syracuse University
qiqiu@syr.edu

Fanxin Kong
University of Notre Dame
fkong@nd.edu

2024 International Conference on Assured Autonomy (ICAA) | 979-8-3315-2101-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICAA64256.2024.00019

***Abstract*—Attack defense is a critical research problem in Cyber-Physical System security. While significant research has focused on attack detection, the crucial aspect of attack diagnosis, particularly temporal diagnosis, remains underexplored. This paper addresses this critical gap by proposing a novel approach to actuator attack diagnosis. We present a real-time, data-driven solution for actuator attack diagnosis in CPS that operates without prior knowledge of the system and is not limited to specific attack types. Our approach leverages attention mechanisms and provides both temporal and spatial diagnosis. Through extensive testing using high-fidelity simulators and a case study on Raspberry Pi, we demonstrate the robustness, accuracy, and efficiency of our method. This research contributes to advancing CPS security and facilitating effective attack recovery strategies.**

***Index Terms*—cyber-physical systems, actuator attacks, real-time, detection, diagnosis**

I. INTRODUCTION

Cyber-Physical Systems (CPS) integrate computing and communication components with sensing and actuation to engage with the physical environment. The progression of CPS has shifted from simple control systems to intricate, diverse networks, facilitating advanced capabilities. This advancement fosters the emergence of novel applications like autonomous vehicles, unmanned aerial vehicles, and intelligent manufacturing, presenting significant advantages. At the same time, the open architecture of modern CPS gives rise to potential security vulnerabilities [1]–[3].

Contrary to conventional IT systems, the challenges in CPS security stand out not only in terms of the consequences of security breaches but also in the breadth of attack surfaces [4]–[6]. Breaches in CPS can result in physical property damage and pose potential risks to human lives, for example, plant explosion [7], power cutoff [8] and car accidents [9].

Actuators, which take commands from the cyber side and enact physical changes in the environment, serve as the bridge between the two domains. Consequently, their command integrity is critical to CPS security. Control commands can be spoofed both in cyberspace, such as through software and network attacks [10] and in the physical space, such as transduction attacks [11], [12].

The increasing prevalence of security threats has prompted significant research efforts aimed at safeguarding the integrity of actuators [13]. A key area of focus is the detection of compromised actuator command data, known as actuator attack detection. Studies in this domain can be categorized into two main groups. The first category utilizes domain knowledge of Cyber-Physical Systems (CPS), often leveraging mathematical models of the physical system, including both linear and nonlinear models [14]. The second category employs deep-driven methods to handle the high dimensionality inherent in spatial (i.e., involving a large number of sensors) and temporal (i.e., dealing with long time series) aspects [1], [15]–[17].

Despite considerable efforts in attack detection, another critical area, attack diagnosis, remains insufficiently addressed. Attack diagnosis focuses on two core aspects: spatial diagnosis, which involves identifying the specific dimensions of the attacked system, and temporal diagnosis, which aims to pinpoint when the attack began.

Spatial diagnosis, also known as attack localization or identification, has garnered research attention. Similar to attack detection, existing spatial diagnosis methods can be grouped into two categories. The first group utilizes prior mathematical models [18], [19]. Recognizing the challenges in obtaining accurate mathematical models [20], the second group uses deep-driven models as the solution [21], [22].

In contrast, few works have addressed the temporal aspect of attack diagnosis in CPS. This lack of attention stems from two primary reasons. First, temporal diagnosis is a relatively new research area, and its significance in CPS has not been fully realized. Historically, researchers viewed attack detection as the final step in the defense process, assuming immediate system reboot upon detection. However, certain systems, like drones in flight, cannot be instantly rebooted. Consequently, research on attack recovery, aimed at restoring CPS to safe states, has gained traction [23]–[29].

Meanwhile, temporal attack diagnosis is gaining attention due to its importance in identifying trustworthy historical data for estimating the current system state during attack recovery. Second, many researchers underestimated the complexities of attack diagnosis, assuming that fast attack detection methods suffice for diagnosis. However, this assumption overlooks the variability in attack magnitudes and resulting varying detection delays. While robust detection methods offer short delays, these delays cannot reliably pinpoint the attack starts, especially for stealthy attacks with prolonged delays. Therefore, advanced techniques in attack diagnosis are essential to complement existing detection strategies. Recently, [30] proposed a data-driven sensor attack temporal diagnosis system. However,

979-8-3315-2101-1/24/$31.00 ©2024 IEEE
DOI 10.1109/ICAA64256.2024.00019

the actuator attack diagnosis area remains unexplored.

Motivated by the above observations, this work focuses on the development of a real-time data-driven attack diagnosis solution for cyber-physical systems under actuator attack. The attack diagnosis is triggered subsequent to attack detection, making it compatible with most existing detection methods. The system comprises six modules trained offline and executed online to offer temporal actuator attack diagnosis. Our solution explores novel uses of the attention mechanisms [31] and its feature of being sensitive to input changes. To be specific, the contribution of this work is summarized as follows:

- We address a critical research gap in actuator attack diagnosis for CPS. Our solution diagnoses actuator attacks with zero knowledge of the corresponding CPS and is not confined to certain types of attacks, facilitating effective attack recovery and advancing related research domains.
- We propose a lightweight real-time end-to-end attack diagnosis system addressing both spatial and temporal diagnosis for CPS actuator attack. It provides diagnosis in millisecond-level computing time in edge devices such as Raspberry Pi.
- We validate the effectiveness of our proposed method through extensive testing using high-fidelity simulators on the Raspberry Pi platform. Our experimental results confirm that the solution can robustly and accurately diagnose attacks with minimal computational overhead. These findings establish the practicality and effectiveness of our approach for real-world applications in Cyber-Physical Systems (CPS).

The rest of this paper is organized as follows. Section II presents preliminaries. Section III describes the system overview. Section IV details the design for each system component. Section V evaluates the proposed solution. Section VI concludes the paper.

II. PRELIMINARIES

This section outlines the scope of the paper, the system model, and the threat model.

A. System Model

We consider a Cyber-Physical System (CPS) in which a controller manages a physical system to adhere to control references. The controller operates on a periodic schedule. During each control step, the controller first retrieves sensor measurements (such as velocity, pressure, etc.) and then applies a control policy to compute control commands or signals (e.g., throttle). These commands are subsequently executed on the actuators to influence the physical system. In this paper, the control references, control commands, and sensor measurements are treated as N_r-dimensional, N_c-dimensional, and N_s-dimensional multivariate time series, respectively, denoted by $\boldsymbol{R}$, $\boldsymbol{C}$, and $\boldsymbol{S}$. $\boldsymbol{R} = \{\boldsymbol{r}_1, .., \boldsymbol{r}_{N_r}\} \in \mathbb{R}^{t \times N_r}$, $\boldsymbol{C} = \{\boldsymbol{c}_1, .., \boldsymbol{c}_{N_c}\} \in \mathbb{R}^{t \times N_c}$, and $\boldsymbol{S} = \{\boldsymbol{s}_1, .., \boldsymbol{s}_{N_s}\} \in \mathbb{R}^{t \times N_s}$, where t denotes the number of time steps until the current time.

Every dimension of $\boldsymbol{R}$, $\boldsymbol{C}$, or $\boldsymbol{S}$ represents a reference, control, or sensor channel. We use subscript i and superscript t to represent the data of i^{th} dimension and t^{th} time step:

$$\boldsymbol{r}_i = \{r_i^1, .., r_i^t\}^T \in \mathbb{R}^{t \times 1}, \boldsymbol{r}^t = \{r_1^t, ..., r_{N_r}^t\} \in \mathbb{R}^{1 \times N_r}.$$

$\boldsymbol{c}_i$ and $\boldsymbol{s}_i$ form as the same. Our actuator attack diagnosis system also uses the entire CPS system state for prediction, and we use $\boldsymbol{X}$ for notation:

$$\boldsymbol{X} = \{r_1, ..., r_{N_r}, c_1, ..., c_{N_c}, s_1, ..., s_{N_s}\} \in \mathbb{R}^{t \times N},$$

where $N = N_r + N_c + N_s$. Moreover, we use double superscripts t, w to indicate the sub-multivariate time series or univariate time series with a window size of w:

$$\boldsymbol{R}^{t,w} = \{\boldsymbol{r}^{t-w}, ..., \boldsymbol{r}^{t-1}\}^T = \{\boldsymbol{r}_1^{t,w}, ..., \boldsymbol{r}_{N_r}^{t,w}\} \in \mathbb{R}^{w \times N_r},$$

$\boldsymbol{C}^{t,w}$, $\boldsymbol{S}^{t,w}$ and $\boldsymbol{X}^{t,w}$ form as the same.

B. Threat Model

We consider a malicious attacker who can launch actuator attacks manipulating the control commands transmitted to actuators generated by the controller. Consequently, the actuators may engage with the physical environment in an adverse manner, potentially leading the system into an unsafe state. The attacker can compromise the integrity of control commands: $\hat{\boldsymbol{C}}^{\tau,\hat{w}} = \boldsymbol{C}^{\tau,\hat{w}} \pm \boldsymbol{V}^{\tau,\hat{w}}$, where τ and $\hat{w}$ denote the attack end and duration, and $\hat{\boldsymbol{C}}^{\tau,\hat{w}}$ and $\boldsymbol{V}^{\tau,\hat{w}}$ are the compromised value and attack magnitude from time step $\tau - \hat{w}$ to $\tau - 1$. This threat model is widely used in CPS and security papers [32], [33]. Typically, defense strategies concentrate on mitigating a specific attack vector (such as actuator attacks in this paper) since a singular defense against all types of attacks is often unfeasible. Therefore, this paper centers its focus on actuator attacks, operating under the assumption that the other system components remain unaffected.

Please note that our proposed method is not limited to specific types of attacks. The following merely provides examples used for evaluation purposes. One common attack type is the bias attack, wherein the attacker alters measurements by introducing fixed values, leading to a stable drift: $v_i^t = v_i^{\tau-\hat{w}}$, where $t \in (\tau - \hat{w}, \tau)$. The second type is a stealthy attack, which gradually modifies control commands starting from a negligible magnitude: $v_i^t = v_i^{t-1} + v_i^{\tau-\hat{w}-1}$, where $t \in [\tau - \hat{w}, \tau)$ and $v_i^{\tau-\hat{w}-1} = 0$.

C. Problem Statement

While numerous studies have focused on attack detection, there remains a critical gap in post-detection diagnosis. In this work, the proposed actuator attack diagnosis system is designed to determine attack onset and identify the most affected actuator. We use z and $\hat{i}$ to denote the attack start time and most affected actuator at detection time. Here, $z = \tau - \hat{w}$, and $\hat{i} = \text{argmax}_i |c_i^{\tilde{\tau}} - \hat{c}_i^{\tilde{\tau}}|$, where $\tilde{\tau}$ is the time step when the system attack detection module detects the attack.

The proposed attack diagnosis module is designed to provide a diagnosis suggesting the attack onset ϕ and the most affected actuator ψ that minimize the diagnosis error:

$$\underset{\phi}{\text{minimize}}\, max(0, \phi - z), \ \underset{\psi}{\text{minimize}}\, |\psi - \hat{i}|.$$

The proposed diagnosis module does not serve as a substitute for attack detection; rather, it complements most existing data-driven attack detection models. This research contributes to the field by offering a comprehensive approach to CPS security.

III. SYSTEM OVERVIEW

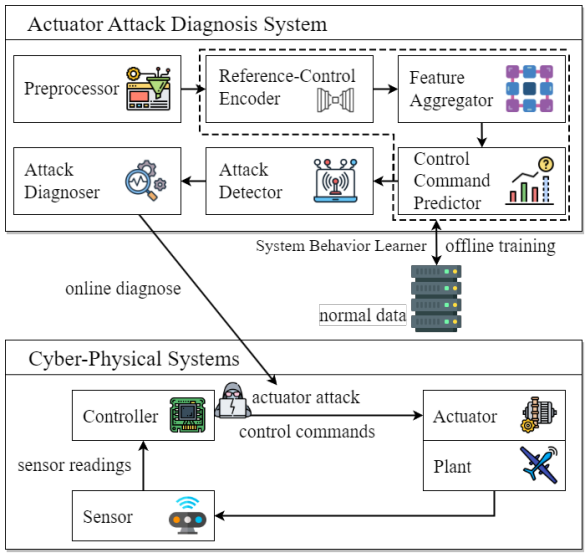

Fig. 1: Actuator Attack Temporal And Spatial Diagnosis System Overview.

This section presents the overview of our real-time actuator attack diagnosis system. Fig. 1 depicts the system design. Our actuator attack diagnosis system comprises six interconnected components operating sequentially:

1) Preprocessor: Responsible for scaling and normalizing system inputs, ensuring consistent data representation across different input ranges and types.
2) Reference-Control (Ref-Ctrl) Encoder: Encodes control references and control commands into a latent space for later use in attack detection and diagnosis.
3) Feature Aggregator: Composes rich feature inputs by projecting encoded latent features and other system information.
4) Control Command Predictor: Uses rich feature inputs to predict control command in the following time step.
5) Attack Detector: Identify potential actuator attacks and raise the alarm after detection.
6) Attack Diagnoser: Analyzes the detected attacks and determines the attack onset and most affected actuator.

These components can be categorized into two groups: algorithm-based and data-driven. The algorithm-based components include the preprocessor, attack detector, and attack diagnoser. Data-driven components, collectively referred to as the system behavior learner, consist of the reference-control encoder, feature aggregator, and control command predictor.

The incorporation of data-driven components necessitates an offline training phase for the system behavior learner to effectively learn and predict control commands. In this context, "system behaviors" encompass the linear or non-linear mappings from control references and system states to control commands. These behaviors are independent of specific operator preferences or algorithm implementations, ensuring a generalized approach to system behavior modeling. The system behavior learner employs a zero-positive learning approach, training exclusively on normal, unattacked data. This methodology is predicated on the learner's ability to accurately forecast system behavior in subsequent control loops with minimal error when provided with unattacked inputs. The rationale behind this approach is three-fold:

- Predictive accuracy: By training on normal data, the learner develops a robust understanding of expected system behaviors, enabling highly accurate predictions under normal operating conditions.
- Attack sensitivity: When confronted with attack-induced input patterns unseen during training, the predictor will generate significantly erroneous predictions due to misinterpretations. Prediction errors serve as crucial indicators for the detection and subsequent diagnosis of attacks.
- Performance consistency: Regardless of the scope and extent of data collection from attacked systems. This characteristic ensures that the proposed system is not restricted to specific types of attacks, enhancing its versatility and applicability across various attack scenarios.

The offline training process continues until the system behavior learner's performance converges. Upon convergence, the learner's parameters are fixed, enabling it to operate online alongside other components within the CPS control loop to provide real-time diagnosis.

It is important to note that while the Attack detector could potentially be implemented as a data-driven component, we do not delve into its optimization in this work as it falls outside the primary focus of our study. Future research could explore the potential benefits of a data-driven approach to attack detection, potentially improving the system's overall performance and adaptability.

IV. ATTACK DIAGNOSIS SYSTEM

The proposed actuator attack diagnosis system comprises six components, with three core components—reference-control encoder, feature aggregator, and attack diagnoser—being the focus of this work. We claim novelty only for the core modules in actuator attack diagnosis, while acknowledging that standard approaches may be applied to the supporting modules. The design of these supporting modules is presented flexibly, allowing for the adoption of alternative models as deemed appropriate. Figure 2 illustrates the details of the modules' design and corresponding notations.

A. Preprocessor

The preprocessor ensures consistent data representation across various input ranges and types, which is essential for the accurate functioning of the attack diagnosis system.

During the offline training phase, the preprocessor determines the maxima and minima for each dimension of

the system state $\boldsymbol{X}$. These values establish the lower and upper bounds within which the attack diagnosis system can accurately process data. For both online training and online running phases, the preprocessor projects inputs into these predetermined bounds using a min-max normalization technique. This scaling operation is defined as: $\hat{x}_i^t = (x_i^t - min(\tilde{\boldsymbol{x}}_i))/(max(\tilde{\boldsymbol{x}}_i) - min(\tilde{\boldsymbol{x}}_i))$, where $\hat{x}_i^t$ and x_i^t represent the scaled and original values, respectively, for the i-th dimension at time step t. This normalization ensures that all input features are scaled $[0, 1]$, which helps prevent certain features from dominating others due to differences in their original scales. In the rest of this paper, we use $\boldsymbol{X}$ to denote the scaled inputs when it does not cause confusion for cleaner expressions.

The effectiveness of the preprocessor is contingent upon the input values falling within the established bounds. Values outside these bounds can lead to extrapolation errors and consequently, incorrect diagnoses. While extending these bounds to accommodate a wider range of inputs is a potential area for future research, it falls outside the scope of this paper. Instead, we focus on the system's performance within the defined operational range, which encompasses the majority of expected scenarios based on our training data.

B. Reference-Control Encoder

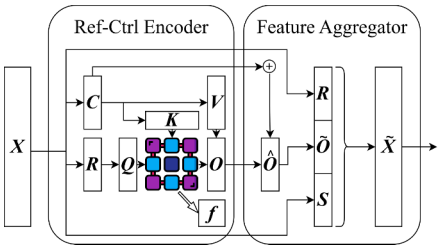

Fig. 2: Details of Ref-Ctrl encoder and feature aggregator.

The reference-control encoder, shown in Figure 2, derived from the design of [30], introduces a novel attention mechanism tailored for cyber-physical systems (CPS). This mechanism, called Reference-Control Command attention (Ref-Ctrl attention), replaces the self-attention in the vanilla multi-head attention mechanism (MHA) [31].

The reference-control encoder serves two purposes: 1) Embedding control references and commands into latent features for improved prediction and detection, and 2) Facilitating temporal attack diagnosis through a novel Attention Fluctuation Index (AFI). Initially, we discuss the vanilla Multi-Head Attention (MHA) mechanism, followed by our proposed Reference-Control (Ref-Ctrl) attention, and then introduce the AFI for diagnosing temporal actuator attacks.

1) Vanilla MHA: In the context of CPS, vanilla MHA processes a window of the entire system states $\boldsymbol{X}^{t,w}$ at every time step t, projecting it into a latent space embedding that represents underlying system behavior. First, $\boldsymbol{X}^{t,w}$ is triplicated and projected into query $\boldsymbol{Q}^{t,w}$, key $\boldsymbol{K}^{t,w}$, and value $\boldsymbol{V}^{t,w}$ using distinct fully-connected layers with weights $\boldsymbol{W}_q$, $\boldsymbol{W}_k$, and $\boldsymbol{W}_v$, respectively. The resulting query, key, and value maintain the same shape as the input $\boldsymbol{X}^{t,w}$. Second, it dot-products $\boldsymbol{Q}^{t,w}$ and $\boldsymbol{K}^{t,w}$ and generates a matrix of attention weights $\mathcal{A}^t$ at each time step. At last, the attention weights $\mathcal{A}^t$ are used to reweight value $\boldsymbol{V}^{t,w}$, producing a latent representation, which is then projected by $\boldsymbol{W}_o$ to obtain a latent feature $\boldsymbol{O}^{t,w}$. Formally:

$$\boldsymbol{Q}^{t,w} = \boldsymbol{X}^{t,w}\boldsymbol{W}_q,\ \boldsymbol{K}^{t,w} = \boldsymbol{X}^{t,w}\boldsymbol{W}_k,\ \boldsymbol{V}^{t,w} = \boldsymbol{X}^{t,w}\boldsymbol{W}_v,$$
$$\mathcal{A}^t = softmax(\boldsymbol{Q}^{t,w}\{\boldsymbol{K}^{t,w}\}^T),\ \boldsymbol{O}^{t,w} = \mathcal{A}^t\boldsymbol{V}^{t,w}\boldsymbol{W}_o,$$

where $\boldsymbol{Q}^{t,w}$, $\boldsymbol{K}^{t,w}$, $\boldsymbol{V}^{t,w}$, and $\boldsymbol{O}^{t,w} \in \mathbb{R}^{w\times N}$, $\mathcal{A}^t \in \mathbb{R}^{w\times w}$, and $\boldsymbol{W}_q$, $\boldsymbol{W}_k$, $\boldsymbol{W}_v$, and $\boldsymbol{W}_o \in \mathbb{R}^{N\times N}$.

The MHA architecture allows for various attention strategies across different sets of dimensions by dividing the inputs into distinct groups. When multiple heads are employed, $\boldsymbol{Q}^{t,w}$, $\boldsymbol{K}^{t,w}$, and $\boldsymbol{V}^{t,w}$ are split into H subsets. Each head computes $\mathcal{A}$ and $\boldsymbol{O}$ independently. The final $\mathcal{A}$ is the average of all heads, while $\boldsymbol{O}$ is their concatenation.

Rationale for using MHA: The adoption of MHA in our attack diagnosis system is motivated by several key factors:

- Enhanced Generalization: MHA significantly extends neural networks' ability to generalize, enabling more precise and accurate predictions [34], [35].
- Interpretability: MHA provides insights into the neural network's focus [36], [37], which is crucial for determining attack onsets.
- Sensitive to Input Fluctuations: The matrix multiplication in the attention mechanism amplifies changes, making them more detectable. Also, a well-trained attention network learns to focus more on significant fluctuations, effectively filtering out noise.

2) Ref-Ctrl MHA: Different from the vanilla attention mechanism that employs self-attention by projecting the same inputs $\boldsymbol{X}^{t,w}$ to $\boldsymbol{Q}^{t,w}$, $\boldsymbol{K}^{t,w}$, and $\boldsymbol{V}^{t,w}$, our novel Ref-Ctrl attention mechanism utilizes different inputs for these projections. This design choice offers several advantages in the context of CPS actuator attack diagnosis.

In Ref-Ctrl attention, control references $\boldsymbol{R}^{t,w}$ are projected to $\boldsymbol{Q}^{t,w}$ using a fully-connected layer with weights $\boldsymbol{W}_q \in \mathbb{R}^{N_r\times N_r}$. In addition, control commands $\boldsymbol{C}^{t,w}$ are projected to both $\boldsymbol{K}^{t,w}$ and $\boldsymbol{V}^{t,w}$ using fully-connected layers with weights $\boldsymbol{W}_k \in \mathbb{R}^{N_c\times N_r}$ and $\boldsymbol{W}_v \in \mathbb{R}^{N_c\times N_r}$, respectively. Consequently, $\boldsymbol{O}^{t,w} \in \mathbb{R}^{w\times N_r}$ and $\boldsymbol{W}_v \in \mathbb{R}^{N_r\times N_r}$. The proposed Ref-Ctrl attention offers several key advantages over conventional self-attention:

- Computational Efficiency: By reducing input dimensions from N to N_r, Ref-Ctrl attention decreases computational overhead in computing attention weights $\mathcal{A}$.
- Enhanced Correlation Exploitation: Ref-Ctrl attention fully leverages the correlation between control references and control commands by projecting them as $\boldsymbol{Q}$ and $\boldsymbol{K}$/$\boldsymbol{V}$, respectively. This approach enables our solution to capture the complex relationships between these elements more effectively, leading to improved performance.
- Reduced Interference in Attention Weights: By avoiding the placement of the same components ($\boldsymbol{R}$, $\boldsymbol{C}$, or $\boldsymbol{S}$) in both $\boldsymbol{Q}$ and $\boldsymbol{K}$, Ref-Ctrl attention minimizes interference

in attention weights. This is particularly important in CPS contexts where there may be a lag between $\boldsymbol{R}$, $\boldsymbol{C}$, and $\boldsymbol{S}$. If $\boldsymbol{Q}$ or $\boldsymbol{K}$ contained all these components, it would be challenging for the attention weights to effectively address all phases of the system's operation. By mitigating this interference, Ref-Ctrl attention enables more accurate attack temporal diagnosis.

3) Attention Fluctuation Index: The Attention Fluctuation Index (AFI) $\boldsymbol{f}$ is a novel metric designed to represent attention weights and plays a pivotal role in our proposed solution for accurate temporal diagnosis of CPS actuator attacks. This score is motivated by the observation that attention weights $\mathcal{A}$ exhibit discernible fluctuations in response to input changes, particularly when utilizing our specially designed Ref-Ctrl Attention mechanism. These fluctuations align with input variations, providing valuable insights into actuator attacks.

Our extensive exploration has revealed two key characteristics of these fluctuations: 1) Fluctuations typically disrupt previously stable patterns within $\mathcal{A}$, influencing neighboring time steps by either attracting or redistributing weights. This temporal impact is crucial for identifying the onset of potential attacks. 2) The distribution of weights across dimensions within these fluctuations is uneven and lacks predictability. This characteristic necessitates a robust metric capable of capturing diverse fluctuation patterns.

Given these observations, we introduce the "attention fluctuation index (AFI)" as a metric to identify the time steps when fluctuations display in their corresponding $\mathcal{A}$. Developing such a metric presents several challenges: 1) the metric must be sufficiently sensitive to capture subtle fluctuations that may indicate the early stages of an attack. 2) it should facilitate accurate temporal diagnosis, pinpointing the exact time steps when attacks potentially occur. 3) the computational overhead associated with calculating and tracking the metric must be carefully balanced to ensure real-time applicability in CPS.

After extensive experimentation and analysis, we found that tracking the maxima and minima in $\mathcal{A}$ provides an effective approach that balances these factors. The inclusion of minima is essential because the attention mechanism may allocate small attention weights to attacks at the beginning in certain attack settings, which are not easily predictable. Formally:

$$
\begin{aligned}
&a_u^t = max(\mathcal{A}^t) - 1/w,\ a_l^t = 1/w - min(\mathcal{A}^t),\\
&m_j^t = \frac{\sum_{t'=max(\phi', t-w)}^{t-1} a_j^{t'}}{t - max(\phi', t-w)},\ j \in \{u, l\},\\
&\boldsymbol{f}^t = \{f_u^t, f_l^t\} = \{\frac{|a_u^t - m_u^t|}{m_u^t}, \frac{|a_l^t - m_l^t|}{m_l^t}\},
\end{aligned}
$$

where ϕ' is initialized as 0 and represents the latest fluctuation, and $1/w$ represents the average weights under the softmax function, serving as a baseline to reduce variance. The AFI's design inherently supports the temporal attack diagnosis. By continuously updating and comparing against recent historical data, the index can pinpoint attack onset. The system keeps updating ϕ', and we present the algorithm to determine it in the attack diagnoser component.

Rationale for the moving windows: The attention weight maxima a_u^t and minima a_l^t are observed to vary within a small range when the CPS is operating normally. However, they exhibit significant fluctuations after sudden control command changes because the model learned to pay more attention to changes, leading to the allocation of more attention to abnormal time steps and resulting in polarized attention weights. To capture this behavior, two moving windows are maintained, and the AFI is calculated as maxima/minima divided by their respective moving averages. This approach allows amplification of anomalies, enhancing detection sensitivity.

In summary, the Attention Fluctuation Index (AFI) serves as a powerful tool for temporally diagnose potential attacks in CPS by leveraging the unique properties of attention weights in our Ref-Ctrl Attention mechanism. The reference-control encoder only records the AFI and the attack diagnoser analyses it to determine the attack onset. Because AFI cannot be used for detection and is only useful after detecting attacks.

C. Feature Aggregator

The feature aggregator, shown in Figure 2, serves as a crucial intermediary between the ref-ctrl encoder and the control command predictor. This component performs three steps: projection, skip connection, and feature aggregation.

The Aggregator first projects the encoded reference-control matrix $\boldsymbol{O}^{t,w} \in \mathbb{R}^{w \times N_r}$ to a new space $\hat{\boldsymbol{O}}^{t,w} \in \mathbb{R}^{w \times N_c}$. This projection aligns the dimensions of the encoded reference-control matrix with the control command matrix. Subsequently, it applies a skip connection from the control command matrix $\boldsymbol{C}^{t,w}$ to $\hat{\boldsymbol{O}}^{t,w}$, resulting in $\tilde{\boldsymbol{O}}^{t,w}$. Formally:

$$
\begin{aligned}
\hat{\boldsymbol{O}}^{t,w} &= \boldsymbol{O}^{t,w}\boldsymbol{W}_{\hat{o}},\\
\tilde{\boldsymbol{O}}^{t,w} &= SiLU(\boldsymbol{C}^{t,w} + \hat{\boldsymbol{O}}^{t,w}),\\
\tilde{\boldsymbol{X}}^{t,w} &= \{\boldsymbol{R}^{t,w}, \tilde{\boldsymbol{O}}^{t,w}, \boldsymbol{S}^{t,w}\},
\end{aligned}
$$

where $\boldsymbol{W}_{\hat{o}} \in \mathbb{R}^{N_f \times N_c}$ is a learnable weight matrix, $\tilde{\boldsymbol{O}}^{t,w} \in \mathbb{R}^{w \times N_c}$, and SiLU (Sigmoid Linear Unit) [38] is an activation function chosen for its superior performance compared to other commonly used activation functions.

In its final step, the feature aggregator constructs the comprehensive feature matrix $\tilde{\boldsymbol{X}}^{t,w} \in \mathbb{R}^{w \times N}$ by concatenating $\tilde{\boldsymbol{O}}^{t,w}$ with the reference matrix $\boldsymbol{R}^{t,w}$ and the sensor measurement matrix $\boldsymbol{S}^{t,w}$. This aggregated feature matrix serves as the input for the subsequent control command predictor.

Rationale for skip connection: The skip connection from $\boldsymbol{C}^{t,w}$ to $\tilde{\boldsymbol{O}}^{t,w}$ serves two primary purposes: 1) It mitigates the vanishing gradient problem during backpropagation, facilitating more stable and efficient training. 2) It preserves the original control command information, ensuring that critical data is not lost during the encoding and projection processes.

Rationale for feature aggregation: The concatenation of $\tilde{\boldsymbol{O}}^{t,w}$, $\boldsymbol{R}^{t,w}$, and $\boldsymbol{S}^{t,w}$ provides a comprehensive representation of the system state: 1) $\tilde{\boldsymbol{O}}^{t,w}$ captures the cross-encoded information from control references and commands. 2) $\boldsymbol{R}^{t,w}$ and $\boldsymbol{S}^{t,w}$ provide direct access to the original control references

and sensor measurements, allowing the model to consider the actual system state and, together with aligned control commands, enabling better predictions.

This rich, multi-faceted feature representation enhances the control command predictor's ability to accurately forecast future control commands, thereby improving the overall performance of the actuator attack diagnosis system.

D. Control Command Predictor

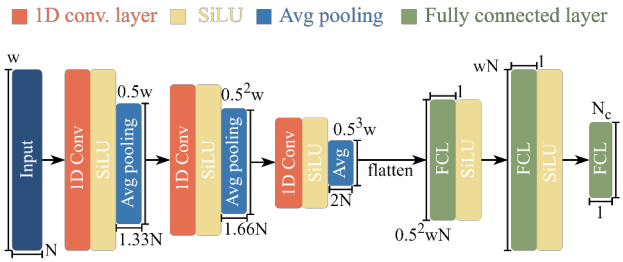

Fig. 3: **Model Architecture of Control Command Predictor.**

The control command predictor, shown in Figure 3, employs a neural network to forecast the control command for the subsequent time step, denoted as $\boldsymbol{y}^{t+1}$, utilizing the comprehensive feature matrix $\tilde{\boldsymbol{X}}^{t,w}$ as input. The prediction is crucial for detecting potential discrepancies between expected and actual control commands, which may indicate the presence of an actuator attack.

While designing adaptable prediction neural networks presents challenges, our work focuses on demonstrating the efficacy of the proposed diagnosis system rather than introducing novel network architectures. Previous research [30] has shown that Convolutional Neural Networks (CNNs) offer a balanced trade-off between performance and computational overhead in similar prediction tasks. Consequently, we adopt a CNN-based approach as a representative example of commonly used prediction models in this domain.

Our CNN implementation consists of two components: convolutional sub-net and fully connected sub-net. The convolutional sub-net comprises three sequential blocks, each containing a 1D-convolution layer, a Sigmoid Linear Unit (SiLU) activation layer, and an average pooling layer. The final block's convolution layer has a spatial dimension of $2N$ and employs a kernel size of 3 with a padding length of 1 on both sides. The parameters of the preceding layers increase linearly to accommodate the growing complexity of extracted features. The average pooling layers serve to downsample the temporal dimension by averaging every two consecutive time steps, effectively reducing the sequence length while preserving important temporal information.

Following the convolutional layers, outs will be flattened and fed to the fully connected sub-net. The first layer doubles the hidden size to allow for more complex feature interactions, while the second layer projects the enlarged hidden representation into an N_c-dimensional space, meaning the prediction of N_c-dimensional control commands.

The control command predictor is the final data-driven component in our system, and the back-propagation for parameter optimization begins from this point during the offline training phase. We employ the mean absolute error between predicted and actual control commands as the loss function, aiming to minimize this error across all time steps. Formally, for every time step t in the training phase: $loss_\theta^t = \sum_{i=1}^{N_c} |y_i^t - c_i^t|$, where y_i^t is the predicted control command for i-th actuator at time step t, and θ represents the learnable parameters in the reference-control encoder, feature aggregator, and control command Predictor. The objective during the training phase is to minimize the expected loss: $\text{minimize}_\theta \, \mathbb{E}[loss_\theta]$, where $\mathbb{E}[\cdot]$ computes the average loss across all training samples.

E. Attack Detector

The attack detector continuously analyzes the discrepancies between predicted and actual control commands at each time step. Its primary function is to calculate a score that quantifies the deviation of the actual control commands from the predicted ones, raising an alarm when this score surpasses a predefined threshold. While various methods can be employed to calculate this score, such as Local Outlier Factor (LOF), One-Class Support Vector Machines (OCSVM), and Isolation Forest, we opt for the Cumulative Sum (CUSUM) method due to its optimal balance of effectiveness, accuracy, and computational overhead [39].

The CUSUM method operates by computing residuals through element-wise squared errors and assigning a CUSUM score for each dimension of the control command vector. The mathematical formulation of this process is as follows: $e_i^t = (y_i^t - c_i^t)^2$, $d_i^t = \max(0, d_i^{t-1} + e_i^t - \omega_i)$, where d_i^t represents the CUSUM score for actuator i at time step t, e_i^t is the squared error, and ω_i is the drift parameter. The CUSUM score is initialized as $d_i^0 = 0$ for all actuators. The drift parameter ω_i plays a crucial role in filtering out noise and reducing false alarms, thereby enhancing the robustness of the detection mechanism.

When the CUSUM score d_i^t exceeds a predetermined threshold η_i, the attack detector resets d_i^t to 0 and raises an alarm, signaling the detection of an attack. In our implementation, we determine the values of $\boldsymbol{\omega}$ and $\boldsymbol{\eta}$ through offline training and keep them fixed during operation. While adaptive methods for adjusting these parameters on-the-fly exist, such as those proposed in [40] and [29], they are beyond the scope of this work but can be readily integrated into our framework.

The drift parameter ω_i for each actuator is set to the minimum value that exceeds 99% of its prediction errors on a separate validation dataset. This approach ensures that the detector is sensitive enough to capture genuine anomalies while maintaining resilience against normal operational variations. The threshold η_i is then calculated as a multiple of the drift: $\boldsymbol{\eta} = R\boldsymbol{\omega}$, where R is a scalar ratio that primarily determines the detection sensitivity of the system.

It is worth noting that decreasing the value of R universally increases the sensitivity of the attack detector across all systems. However, as the primary focus of this work is on the

temporal diagnosis of actuator attacks rather than optimizing detection sensitivity, we employ a fixed value for R.

F. Attack Diagnoser

The attack diagnoser serves as the final component in our actuator attack diagnosis system, providing both temporal and spatial analysis of detected attacks.

Note that temporal diagnosis is distinct from attack detection and cannot be used independently to raise alarms. This distinction arises because normal system behaviors, such as sudden changes in control references, can also cause attention fluctuations and increase the AFI. The attack detector, on the other hand, raises alarms only for unexpected system behaviors that deviate significantly from normal operations.

The attack diagnoser continuously analyzes $\boldsymbol{f}$ and records the most probable attack onset ϕ, but only provides a formal diagnosis after the attack detector confirms an attack. This approach ensures that the system doesn't misclassify normal fluctuations as attacks. In contrast, spatial diagnosis, which identifies the most affected actuator by analyzing the CUSUM score $\boldsymbol{d}$, can be provided instantly upon attack detection.

To perform temporal diagnosis, the attack diagnoser monitors for fluctuation, i.e., sudden increases or decreases in $\boldsymbol{f}$, by comparing it to a predefined threshold λ. The diagnosis process is more than simply identifying the latest fluctuation because an attack typically invokes continuous fluctuations lasting multiple time steps. It keeps track of the beginning of every fluctuation, assuming that all fluctuation start points are potential attack onsets, treating them as temporal diagnosis candidates. After confirming an attack, the attack diagnoser returns the latest candidate as the temporal diagnosis.

Algorithm 1: Actuator Attack Diagnosis

```
Input: λ, d^t ;        // λ: threshold, d^t: CUSUM score
Output: φ, ψ ;         // φ, ψ: temporal & spatial diagnosis
1  φ ← 1, φ' ← 1; ;  // φ': latest fluctuation
2  while t ≥ 2 do
3      if attack detector detected an attack then
4          ψ ← argmax_i d_i^t; // spatial diagnosis
5          return φ, ψ
6      foreach j in {u, l} do          // Iterate f^t
7          if f_j^t ≥ λ then
8              if t > φ' + 1 then          // If AFI surpasses threshold and not due to prior fluctuation.
9                  φ ← t ;    // latest candidate
10             φ' ← t ;                    // Update φ'
```

Algorithm 1 presents the detailed process of actuator attack diagnosis. The algorithm takes as input the threshold λ and the CUSUM score $\boldsymbol{d}^t$, and outputs the temporal diagnosis ϕ and spatial diagnosis ψ. The algorithm initializes ϕ and ϕ' to 1, where ϕ' serves as an indicator of the latest time step of every fluctuation, while ϕ points to the first time step in each fluctuation. The algorithm executes at every time step but only returns a diagnosis after the attack detector confirms an attack (lines 3-5). If any dimension of the AFI $\boldsymbol{f}$ exceeds the threshold λ (lines 6-7), the algorithm updates ϕ' to indicate the last time step of the current fluctuation (line 10). However, it only updates the temporal diagnosis ϕ if the current fluctuation is newly started, pointing to the current time step as the first step of the new fluctuation (lines 8-9). Upon detection of an attack, the attack detector also provides a spatial diagnosis, identifying the most affected actuator (line 4). This is determined by the highest CUSUM score at the time step when alarm is raised.

It's important to note that attention scores fluctuate both when attacks start and when reference states change normally. However, the latter case is not caused by attacks, and the attack detector will not raise an alarm in such instances. Consequently, the diagnosis will not be triggered for these normal fluctuations. The recorded fluctuations only provide a diagnosis after the attack detector confirms an attack, ensuring that the system distinguishes between normal operational changes and actual attack scenarios.

The threshold λ is a critical hyper-parameter that controls the diagnostic sensitivity of the system. Determining its optimal value often requires an iterative process involving data collection, λ adjustment, and performance evaluation. While different attack types might benefit from varied λ values to maximize diagnostic efficacy, it's generally advisable to choose a λ that accommodates noise-induced effects rather than targeting specific attack types.

Our experiments indicate that the optimal value range for λ can be efficiently determined using a simple approach. Starting with a modest λ value, one can employ a straightforward technique such as the bisection method to refine the threshold. This process allows for the identification of a λ value that balances sensitivity to attacks with resilience against false positives from normal system fluctuations.

In practice, the optimization of λ should be performed as part of the system's calibration process, taking into account the specific characteristics of the CPS being protected and the expected range of normal operational fluctuations. Regular re-evaluation of λ may be necessary to adapt to evolving system dynamics and emerging attack patterns.

V. Experimental Evaluation

To validate the efficacy of our proposed actuator attack diagnosis system, we conduct comprehensive experiments using a high-fidelity simulator and present a case study on a real-world testbed. This section details our implementation, dataset generation, evaluation metrics, and baseline comparisons.

A. Overall Settings

We implement the System Behavior Learner using PyTorch v2.2.2, with parameters optimized by an Adam optimizer. To ensure the robustness and generalizability of our approach,

we train each system configuration five times and report the average performance across these runs. Evaluation of the simulator is on an NVIDIA GeForce RTX 3080. Unless otherwise specified, we use the following default parameters: a window length (w) of 50, a detection threshold ratio (R) of 5, and a diagnosis threshold (λ) of 0.1.

B. Dataset

We employ an ArduCopter drone [41] to generate a high-fidelity dataset that closely mimics real-world CPS scenarios. The drone operates at a frequency of $100Hz$. It has twelve control reference channels ($N_r = 12$), four control command channels ($N_c = 4$), and twenty-eight sensors ($N_s = 28$).

To create a diverse and representative dataset, we operate the drone within a predefined three-dimensional space: latitude: $[40.764485, 40.766485]$, longitude: $[-113.812210, -113.810210]$, altitude: $[20m, 40m]$.

Our data collection process yields 2.28 million time steps of benign operational data, 1.89 million time steps of which are collected by operating the drone to follow rapidly changing references, exploring a wide range of system behaviors. The remaining data are collected by following references set near area edges to capture terrain-specific information. We split this dataset into training and validation sets using a 2:1 ratio, ensuring a robust model development process.

For the test set, we simulate two types of actuator attacks, i.e., bias attack and stealthy attack, on all four drone actuators. Each test record comprises 5000 time steps of normal operation, followed by a 200-time step attack period ($\hat{w} = 200$).

C. Evaluation Metrics and Baseline

To comprehensively evaluate our system's performance, we employ the following metrics, adapted from [30]:

- **False Positive Duration (FPD)**: The ratio of accumulated false alarm time steps to total normal operation time steps. FPD is equivalent to the false positive rate (FPR) in traditional classification tasks.
- **True Positive Rate (TPR)**: Defined as $TPR = T^+/M$, where T^+ is the number of true positive records and M is the total number of records. This metric quantifies the system's ability to correctly identify attack instances.
- **Average Detection Delay (ADD)**: The time delay between the attack initiation and the earliest alarm from the detection module. If an attack goes undetected in a test record, we set the corresponding detection delay to the total duration of the attack.
- **Average Temporal Diagnosis Error (ATDE)**: The absolute error between the actual attack starting time step and the diagnosed time ϕ. For cases where the system fails to deliver a temporal attack diagnosis, we assign the ATDE the value of the total attack duration.
- **Spatial Diagnosis Accuracy (SDA)**: The success rate of spatial diagnosis, indicating the system's ability to correctly identify the most affected actuator.

Given the novelty of temporal actuator attack diagnosis in CPS, there are no existing approaches available as baselines. However, since ADD is traditionally used for defending against attacks, we can evaluate the performance improvement of the proposed approach by comparing ATDE with ADD.

D. Main Results

We evaluate the proposed approach on attacks with different strengths. Results are averaged among five trained models and four actuators and they are shown in Table I. Numbers following attack types are the strengths. For example, Bias 6 means $v_i^t = 6$, and Stealthy 1 means $v_i^{\tau-\hat{w}} = 1$. It is clear that the proposed diagnosis system provided accurate temporal and spatial diagnosis on all scenarios. Specifically, the proposed approach yielded up to 250.54% and 61.24% improvement in temporal diagnosis (ATDE) over conventional detection delay (ADD) for stealthy and bias attacks. Note that a stealthy attack with a strength of 1 is too small for detection, let alone diagnosis. The average overhead running the proposed system is 1.11ms.

TABLE I: Main Results

Attack	TPR	FPD	ADD	ATDE	SDA	Improv.
Stealthy 1	55.00	0.35	121.40	119.75	30.00	1.38
Stealthy 2	100.00	0.13	8.31	2.73	100.00	205.05
Stealthy 3	100.00	0.23	5.43	1.55	100.00	250.54
Stealthy 4	100.00	0.19	4.20	1.37	100.00	207.32
Stealthy 5	100.00	0.23	3.61	1.10	100.00	228.41
Bias 6	95.00	0.32	14.39	12.12	100.00	18.73
Bias 8	100.00	0.14	2.87	1.78	100.00	61.24
Bias 10	100.00	0.21	2.14	1.36	100.00	57.35
Bias 12	100.00	0.16	1.86	1.17	100.00	58.97
Bias 14	100.00	0.25	1.34	1.05	100.00	27.62

E. Showcase of Reference-Control Attention.

To demonstrate the innovative nature and effectiveness of reference-control attention in diagnosing temporal actuator attacks, we present a visual example. Figure 4 displays the attention weights derived from the proposed reference-control attention mechanism when applied to a stealthy attack test record. We focus on four key time points: t1 (10 time steps before the attack), t2 (1 time step before the attack), t3 (1 time step after the attack), and t4 (10 time steps after the attack).

The figure clearly illustrates that as the inputs shift, the attention weights correspondingly shift to the left. Notably, the weights begin to polarize following the onset of the attack. A distinct fluctuation in attention weights is observable from t3 onwards, persisting until t4. This visual representation demonstrates how the novel reference-control attention design provides a robust foundation for precise temporal diagnosis.

The ability to capture the fluctuation in attention weights underscores the proposed system's capacity for accurate temporal actuator attack diagnosis

F. Sensitivity Analysis

To assess the robustness of our model and evaluate the impact of key parameters on the experimental results, we con-

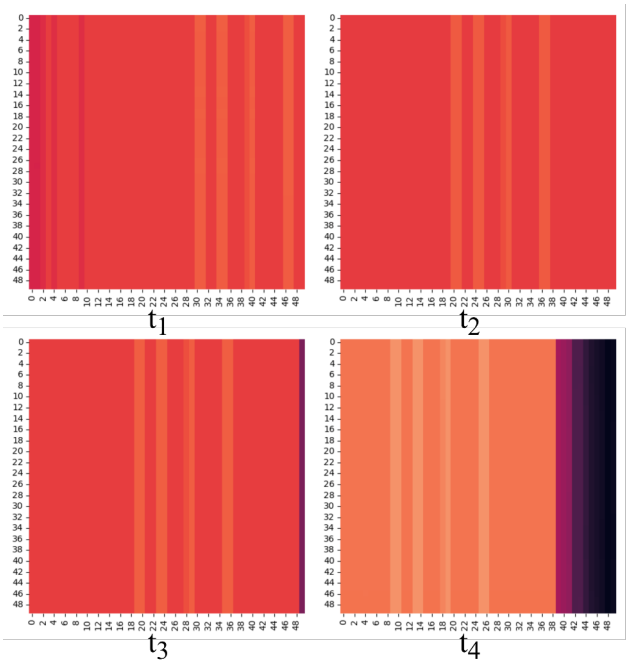

Fig. 4: Showcase of Reference-Control Attention Weights. Brighter areas have higher values.

ducted a comprehensive sensitivity analysis. All experiments are under the Stealthy 3 scenario.

1) Comparison of diagnosis threshold: The diagnosis threshold λ plays a crucial role in determining the sensitivity of the temporal diagnosis. The optimal value for λ varies depending on the specific tasks and scenarios under consideration. Fig. 5 illustrates the impact of different λ values on the ATDE. As evident from the figure, there is a positive correlation between λ and ATDE; as λ increases, so does the ATDE. This relationship can be attributed to the fact that a larger λ value filters out smaller fluctuations, potentially causing the attack diagnoser to provide diagnoses with increased delays.

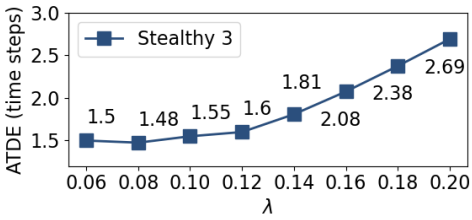

Fig. 5: Comparison of diagnosis threshold.

In real-world applications, a conservative selection of λ is generally appropriate for a broad spectrum of scenarios. This approach is effective because the performance differences among various λ values are typically minimal, provided that the diagnosis algorithm is properly implemented.

2) Comparison of window size: We evaluate the performance across various window sizes, with the results presented in Table II. It is important to note that in this experiment, we solely adjust the window size parameter (w) while keeping other parameters constant, including the number of layers in the CNN. The optimal window size may vary depending on the specific scenario and overall system configuration. A general principle is that a larger w corresponds to a greater amount of information in the inputs, which typically requires a more complex neural network architecture. Consequently, increasing w without correspondingly adjusting the network structure may not necessarily yield improved performance. Moreover, a large w combined with a complex network architecture can impose significant computational overhead on edge devices. Therefore, it is crucial to carefully evaluate and fine-tune the window size parameter through empirical testing, taking into account the specific use case and system requirements. This process ensures an optimal balance between performance and computational efficiency.

TABLE II: Comparison of Window Size.

	TPR	FPD	ADD	ATDE	SDA	Improv.
30	100.00	0.24	5.92	1.82	100.00	225.27
40	100.00	0.24	5.79	1.76	100.00	228.98
50	100.00	0.23	5.43	1.55	100.00	250.54
60	100.00	0.23	5.47	1.52	100.00	259.87
70	100.00	0.24	5.38	1.53	100.00	251.63

G. Testbed Case Study

Figure 6 illustrates our testbed design and presents the experimental results. A stealthy attack is initiated at time step 144, resulting in observable fluctuations. Although the attack detector triggers an alarm at time step 155, the attack diagnoser successfully provides an accurate temporal diagnosis, pinpointing the attack's onset. Running on a Raspberry Pi 4 Model B board computer, the proposed system's average overhead is 2.3ms.

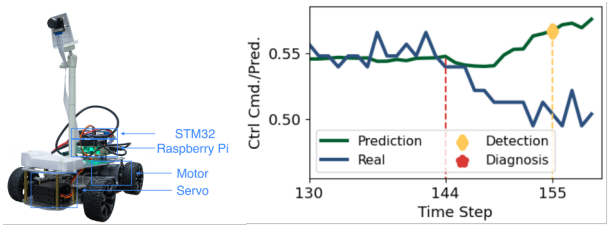

Fig. 6: **Testbed Sensor Attack Temporal Diagnosis.**

VI. CONCLUSION

This paper underscores the critical importance of actuator attack diagnosis in Cyber-Physical Systems (CPS). Precise detection of actuator attack initiation is crucial for maintaining CPS reliability and security. Such detection not only safeguards the system but also facilitates advanced research in areas such as system estimation and attack recovery strategies. Consequently, the development of efficient and accurate actuator attack diagnosis techniques is of paramount importance in the field of CPS security.

ACKNOWLEDGEMENT

This work was supported in part by NSF CNS-2333980. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the National Science Foundation (NSF).

REFERENCES

[1] T. He, L. Zhang, F. Kong, and A. Salekin, "Exploring inherent sensor redundancy for automotive anomaly detection," in *2020 57th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2020, pp. 1–6.

[2] L. Zhang, Z. Wang, and F. Kong, "Work-in-progress: Optimal checkpointing strategy for real-time systems with both logical and timing correctness," in *2022 IEEE Real-Time Systems Symposium (RTSS)*. IEEE, 2022, pp. 515–518.

[3] Z. Yu, Z. Kaplan, Q. Yan, and N. Zhang, "Security and privacy in the emerging cyber-physical world: A survey," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 3, pp. 1879–1919, 2021.

[4] F. Akowuah and F. Kong, "Physical invariant based attack detection for autonomous vehicles: Survey, vision, and challenges," in *4th International Conference on Connected and Autonomous Driving*. IEEE, 2021.

[5] H. Zhu, Z. Yu, W. Cao, N. Zhang, and X. Zhang, "Powertouch: A security objective-guided automation framework for generating wired ghost touch attacks on touchscreens," in *Proceedings of the 41st IEEE/ACM International Conference on Computer-Aided Design*, 2022, pp. 1–9.

[6] Z. Yu, Y. Chang, S. Zhai, N. Deily, T. Ju, X. Wang, U. Jammalamadaka, and N. Zhang, "{XCheck}: Verifying integrity of 3d printed {Patient-Specific} devices via computing tomography," in *32nd USENIX Security Symposium (USENIX Security 23)*, 2023, pp. 2815–2832.

[7] Slammer worm crashed ohio nuke plant network: https://www.securityfocus.com/news/6767. [Online]. Available: https://www.securityfocus.com/news/6767

[8] D. U. Case, "Analysis of the cyber attack on the ukrainian power grid," *Electricity Information Sharing and Analysis Center (E-ISAC)*, vol. 388, pp. 1–29, 2016.

[9] A. H. Rutkin, "Spoofers use fake gps signals to knock a yacht off course," *MIT Technology Review*, 2013.

[10] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, and T. Kohno, "Comprehensive experimental analyses of automotive attack surfaces," in *20th USENIX security symposium (USENIX Security 11)*, 2011.

[11] F. Fakhfakh, M. Tounsi, and M. Mosbah, "Cybersecurity attacks on can bus based vehicles: a review and open challenges," *Library hi tech*, vol. 40, no. 5, pp. 1179–1203, 2022.

[12] X. Ju, D. Zhang, R. Xiao, J. Li, S. Li, M. Zhang, and G. Zhou, "Joint multi-modal aspect-sentiment analysis with auxiliary cross-modal relation detection," in *Proceedings of the 2021 conference on empirical methods in natural language processing*, 2021, pp. 4395–4405.

[13] X. Jin, "Adaptive finite-time fault-tolerant tracking control for a class of mimo nonlinear systems with output constraints," *International Journal of Robust and Nonlinear Control*, vol. 27, no. 5, pp. 722–741, 2017.

[14] Y. Zhang and K. Rasmussen, "Detection of electromagnetic signal injection attacks on actuator systems," in *Proceedings of the 25th International Symposium on Research in Attacks, Intrusions and Defenses*, 2022, pp. 171–184.

[15] M. Kravchik and A. Shabtai, "Efficient cyber attack detection in industrial control systems using lightweight neural networks and pca," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 4, pp. 2179–2197, 2021.

[16] S. V. Thiruloga, V. K. Kukkala, and S. Pasricha, "Tenet: Temporal cnn with attention for anomaly detection in automotive cyber-physical systems," 2021.

[17] Z. Yu, A. Li, R. Wen, Y. Chen, and N. Zhang, "Physense: Defending physically realizable attacks for autonomous systems via consistency reasoning," in *Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security*, 2024.

[18] A. Ameli, A. Hooshyar, A. H. Yazdavar, E. F. El-Saadany, and A. Youssef, "Attack detection for load frequency control systems using stochastic unknown input estimators," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 10, pp. 2575–2590, 2018.

[19] J. Sakhnini, H. Karimipour, A. Dehghantanha, and R. M. Parizi, "Physical layer attack identification and localization in cyber–physical grid: An ensemble deep learning based approach," *Physical Communication*, vol. 47, p. 101394, 2021.

[20] P. Antsaklis, "Goals and challenges in cyber-physical systems research editorial of the editor in chief," *IEEE Transactions on Automatic Control*, vol. 59, no. 12, pp. 3117–3119, 2014.

[21] M. Mohammadpourfard, I. Genc, S. Lakshminarayana, and C. Konstantinou, "Attack detection and localization in smart grid with image-based deep learning," in *2021 IEEE international conference on communications, control, and computing technologies for smart grids (SmartGridComm)*. IEEE, 2021, pp. 121–126.

[22] W. Aoudi, M. Iturbe, and M. Almgren, "Truth will out: Departure-based process-level detection of stealthy attacks on control systems," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, 2018, pp. 817–831.

[23] F. Kong, M. Xu, J. Weimer, O. Sokolsky, and I. Lee, "Cyber-physical system checkpointing and recovery," in *2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems*. IEEE, 2018, pp. 22–31.

[24] L. Zhang, X. Chen, F. Kong, and A. A. Cardenas, "Real-time attack-recovery for cyber-physical systems using linear approximations," in *2020 IEEE Real-Time Systems Symposium*. IEEE, 2020, pp. 205–217.

[25] R. Ma, S. Basumallik, S. Eftekharnejad, and F. Kong, "Recovery-based model predictive control for cascade mitigation under cyber-physical attacks," in *2020 IEEE Texas Power and Energy Conference (TPEC)*. IEEE, 2020, pp. 1–6.

[26] L. Zhang, P. Lu, F. Kong, X. Chen, O. Sokolsky, and I. Lee, "Real-time attack-recovery for cyber-physical systems using linear-quadratic regulator," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 20, no. 5s, pp. 1–24, 2021.

[27] R. Ma, S. Basumallik, S. Eftekharnejad, and F. Kong, "A data-driven model predictive control for alleviating thermal overloads in presence of possible false data," *IEEE Transactions on Industry Applications*, 2021.

[28] L. Zhang, K. Sridhar, M. Liu, P. Lu, X. Chen, F. Kong, O. Sokolsky, and I. Lee, "Real-time data-predictive attack-recovery for complex cyber-physical systems," in *2023 IEEE 29th Real-Time and Embedded Technology and Applications Symposium*. IEEE, 2023, pp. 209–222.

[29] L. Zhang, Z. Wang, M. Liu, and F. Kong, "Adaptive window-based sensor attack detection for cyber-physical systems," in *2022 59th ACM/IEEE Design Automation Conference (DAC)*, 2022, pp. 1–6.

[30] Z. Wang, L. Zhang, Q. Qiu, and F. Kong, "Catch you if pay attention: Temporal sensor attack diagnosis using attention mechanisms for cyber-physical systems," in *2023 IEEE Real-Time Systems Symposium (RTSS)*. IEEE, 2023, pp. 64–77.

[31] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[32] S. Z. Yong, M. Q. Foo, and E. Frazzoli, "Robust and resilient estimation for cyber-physical systems under adversarial attacks," in *2016 American Control Conference (ACC)*. IEEE, 2016, pp. 308–315.

[33] Y. Gao, G. Sun, J. Liu, Y. Shi, and L. Wu, "State estimation and self-triggered control of cpss against joint sensor and actuator attacks," *Automatica*, vol. 113, p. 108687, 2020.

[34] J.-B. Cordonnier, A. Loukas, and M. Jaggi, "On the relationship between self-attention and convolutional layers," *arXiv preprint arXiv:1911.03584*, 2019.

[35] J. Baan, M. ter Hoeve, M. van der Wees, A. Schuth, and M. de Rijke, "Understanding multi-head attention in abstractive summarization," *arXiv preprint arXiv:1911.03898*, 2019.

[36] J. Vig, "Visualizing attention in transformer-based language representation models," *arXiv preprint arXiv:1904.02679*, 2019.

[37] S. Wiegreffe and Y. Pinter, "Attention is not not explanation," *arXiv preprint arXiv:1908.04626*, 2019.

[38] S. Elfwing, E. Uchibe, and K. Doya, "Sigmoid-weighted linear units for neural network function approximation in reinforcement learning," *Neural Networks*, vol. 107, pp. 3–11, 2018.

[39] R. Quinonez, J. Giraldo, L. Salazar, E. Bauman, A. Cardenas, and Z. Lin, "SAVIOR: Securing autonomous vehicles with robust physical invariants," in *29th USENIX Security Symposium*, 2020.

[40] F. Akowuah and F. Kong, "Real-time adaptive sensor attack detection in autonomous cyber-physical systems," in *2021 IEEE 27th Real-Time and Embedded Technology and Applications Symposium (RTAS)*. IEEE, 2021, pp. 237–250.

[41] Ardupilot. [Online]. Available: https://ardupilot.org/