Trusted Operations of a Military Ground Robot in the Face of Man-in-the-Middle Cyber-Attacks Using Deep Learning Convolutional Neural Networks: Real-Time Experimental Outcomes

Fendy Santoso, *Senior Member, IEEE* and Anthony Finn

***Abstract*—Safe and secure operations of robotic systems are of paramount importance. Aiming for achieving the trusted operation of a military robotic vehicle under contested environments, we introduce a new cyber-physical system based on the concepts of deep learning convolutional neural networks (CNNs). The proposed algorithm is specifically designed to reduce the cyber vulnerability of the Robot Operating System (ROS), a well-known middleware platform widely used in both civilian and military robots. To demonstrate the efficacy of the proposed algorithm, we conduct penetration testing (real-time man-in-the-middle cyber attack) on the GVR-BOT ground vehicle, a military ground robot, developed by the United States Army Combat Capabilities Development Command (CCDC), Ground Vehicle Systems Center. The cyber attack also exploits the vulnerability of the Robot Operating System (ROS) employed in its onboard computer. We collect experimental data and train our CNN based on two different operating conditions, namely, legitimate and malicious conditions. We normalize and convert the network traffic data in the form of RGB or grayscale images. We introduce two different types of windowing techniques, namely, the independent and overlapping sliding epochs to efficiently feed the network traffic data to our CNN system. Our research indicates the efficacy of the proposed algorithm as our proposed cyber intrusion detection system can achieve reasonably high accuracy of ≥ 99 % and substantially small false-positive rates $\leq$ 2 % supported with minimum detection time. In addition, we also compare and demonstrate the relative merits of our proposed algorithm with respect to the performance of some well-known techniques, namely, 'bag-of-features' and Support Vector Machine (SVM) algorithms.**

***Index Terms*—Cyber-Security, Robot Operating Systems (ROS), Convolutional Neural-Networks (CNNs), Unmanned Ground Vehicles (UGVs), and Man-in-the-Middle Cyber-Attacks.**

I. Introduction

TRUST indicates a social relationship (bond) between two agents when the truster (e.g. a human) delegates a certain task to the trustee (e.g. a ground robot) in the face of the risk and uncertainties [1]. There is no guarantee that the trustee is dependable despite its role to perform a certain critical task and in spite of its best effort and promise to become dependable.

Thus, trusted operations of robotic systems indicate the dependence (reliance) on the safe operation or trustworthy interaction between human and robotic systems [1]. Considering the autonomous nature of robotic systems, namely, the freedom to make a decision (subject to multiple constraints), trusted autonomy refers to the interaction of multiple independent robotic agents and humans, where one party is vulnerable to other parties in their interaction while autonomously performing a certain demanding task [1].

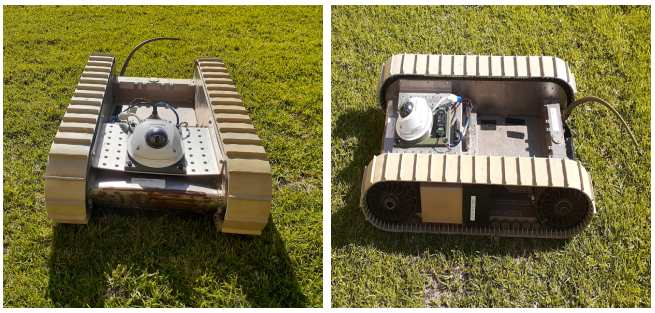


Fig. 1: The GVR-BOT is a replicate of a military unmanned ground vehicle (UGV) employed in our experimental cyber attack. This unmanned ground vehicle is developed by the U.S. Army Tank-Automotive Research, Development, and Engineering Center (TARDEC) and is implemented as a research platform.

Robotic cyber-attacks can breach the trust delegated to the trustee. For instance, the cyber vulnerability of computational resources, communications links, operating systems, software libraries, applications, or sensor information is a serious issue as they can be used to manipulate and compromise the performance of the system so that the operation of a robot may deviate from its original purpose.

Cyber-attacks are indeed serious threats to robotics as people are continuously faced with numerous types of attacks [2], [3] (e.g. malware, denial-of-service, GPS-spoofing, man-in-the-middle, SQL (structured query language) injection, etc); not only in terms of the occurrence but also most importantly, in terms of the quality or the sophistication of attacks. Moreover, the advent of industry 4.0 (the fourth revolution of industry), marked by the evolution in robotics, automation, and the Internet of Things (IoT) [4], [5] has demanded robots to collaboratively worked in highly networked and distributed

Fendy Santoso is with the Artificial Intelligence and Cyber Futures Institute, Charles Sturt University, Barton, Canberra ACT, 2600, Australia. He was with the Defence and Systems Institute, UniSA STEM (Science, Technology, Engineering, and Mathematics), The University of South Australia, Adelaide, Australia.

Anthony Finn is with the Defence and Systems Institute, UniSA STEM (Science, Technology, Engineering, and Mathematics), The University of South Australia, Adelaide, Australia. Corresponding Author: Fendy Santoso, email: Fendy.Santoso@gmail.com.

This research is supported by the US Army Futures Command (AFC) – DEVCOM - Ground Vehicle Systems Center (GVSC) and the International Technology Center Indo-Pacific (ITC-PAC) under contract FA5209-18-P-0146 ITC PAC Fund support for "Trusted Operations for Robotic Vehicles in Contested Environments."

working environments, where multiple agents (e.g. sensors, actuators, and controllers) need to communicate and exchange information with one another via cloud services. As such, the systems are vulnerable to numerous security issues, such as data breaches, and electronic hijacking.

Likewise, military robots, ranging from remotely piloted vehicles (e.g. ground robots (see Fig 1), combat vehicles, and robotic soldiers, are also highly vulnerable to cyber-attacks, as today, cyber-threats have also rapidly flourished in the military domain in parallel with the advancement of other robotics technology (e.g. control, guidance, and navigation systems). For this reason, the research challenges in the area of cyber-physical systems are multidisciplinary as they cover a wide spectrum of scientific disciplines, such as bridging control, communications, networking, as well as cybernetics, and mechatronics.

Given the importance of state-of-the-art autonomous systems in modern society [6], protecting cyber-physical systems from any potential malicious attacks is of critical importance to research tasks to guarantee their trusted and continuous operations. From the attackers' standpoint, multiple security gaps can be explored, such as the vulnerability of the operating (O/S) systems (e.g. Windows and Ubuntu (Linux)), the robotic middle-ware platform [7] (i.e. Robot Operating Systems (ROS), which is an open-source software library, gluing the O/S and its applications) as well as the networking and data distribution systems. In fact, ROS has been widely implemented beyond its original purpose of development for academic research, such as in both civilian and military domains.

The system plays an integral role as the core of software for managing all the computation processes, in addition to memory, data transfer as well as communications and networking of the systems. The system contains a communications platform and repository of libraries for use in robotics [8]. Originally, it was not intentionally designed with a secure platform in mind [8]. For instance, the system employs open topics with limited integrity checking, in addition to its unencrypted network traffic data and anonymous graph structure, making it very susceptible to potential malicious attacks.

The good news is, however, given the current advancement in the speed of computing, which according to Moore's Law, doubles every couple of years; it is now possible to develop and implement some sophisticated artificial intelligence (AI) algorithms, such as deep learning convolutional neural network algorithms, which were deemed to be computationally intensive, but simple, to guard the systems against any digital attacks. As such, we are keen to address this research challenge, and accordingly, we have collaborated with the US Army TARDEC (the US Army Tank-Automotive Command) which provides us with an experimental ground vehicle, the GVR-BOT ground robot as highlighted in Fig. 1, which employs an open-source middleware platform ROS run under Ubuntu Linux as its operating system. Before discussing our potential technical contributions, however, we will discuss current state-of-the-art research papers in the area of cyber-physical systems in the literature as follows.

A. Related Work

There are multiple research papers in the literature addressing the vulnerability of robotic systems. For instance, Lima et al in [9], introduced a defense strategy to prevent man-in-the-middle attacks by means of a security supervisor (NA-Secure System) to address the drawbacks of using firewalls due to its introduced delay.

Santoso in [10] introduced a distributed motion coordination to maximize intrusion detection coverage in wireless sensor networks. The same author in [11], [12], and [13] developed a cyber-physical system to prevent cyber-attacks in vehicular and robotic systems. Renganathan et al. in [14] proposed a class of resilient consensus strategies, namely, weighted mean-subsequence-reduced (W-MSR) consensus by incorporating a physical layer authentication strategy. Despite their promising results, their algorithms are not suitable to monitor large-scale ROS network traffic data.

Joo et al. in [15] developed an attack resilient control for a cyber-physical system (CPS) to address the security against stealthy system integrity attacks. Meanwhile, Ma et al. in [16] studied the problem of the dissipativity-based resilient sliding-mode control design for denial-of-service (DoS) attacks. Also, Wu in [17] developed an active defense-based resilient sliding mode control, specifically designed to counter malicious sensor denial-of-service (DoS) attacks, causing loss of information regarding system state. Cheon et al. in [18] introduced an encryption technique, known as, the linearly homomorphic authenticated encryption (LinHAE) scheme, for the ground control center of a multi-rotor drone to facilitate safe autonomous flight.

Despite current progress, nonetheless, to the best of our knowledge, there has been no research paper that advocates the trusted operations of robotic systems while simultaneously addressing the cyber vulnerability of ROS network traffic data, especially for preventing man-in-the-middle cyber attacks.

Compared to model-based systems discussed earlier, deep learning systems can offer numerous advantages due to their learning capability, flexibility, and scalability, making them suitable candidates to deal with uncertainties in systems, measurements, and unstructured data distributions. In many cases, it is not even possible or impractical to obtain accurate mathematical models describing the dynamics of the systems.

B. Contribution of our Research

Addressing current research gaps in the literature, our technical contributions are to introduce a new application of deep learning convolutional neural networks to address the cyber vulnerability of ROS within the context of a military ground robot.

To be more specific, the framework of our research can be described as follows:

- Specifically designed to prevent man-in-the-middle cyber-attacks, we propose a cyber-intrusion detection framework leveraging the benefits of deep learning convolutional neural networks.
- To transform the data, we develop an interface between the ROS network traffic data (which is of a time series nature and the image input data of the CNN) in the form

of two windowing schemes, namely, the independent and overlapping windowing techniques.
- Exploiting the cyber vulnerability of ROS, we perform a real-time man-in-the-middle cyber-attack (penetration testing) on the replicate of the US military ground robot, the GVR-BOT system, as seen in Fig. 1.
- We collect the real-time ROS network traffic data of the GVR-BOT ground robot under both legitimate operation and malicious attack conditions and transform them into either grayscale or RGB images before using those images to train our intrusion detection system to learn the signature of attack, that is, to be able to detect potential cyber-attack in the minimum time.
- We perform rigorous statistical studies showing the efficacy of the proposed cyber-intrusion detection algorithm in the form of confusion matrices.
- We perform a comparative study to highlight the relative benefits of the independent windowing technique with respect to its overlapping counterparts.
- Last but not least, we also compare the relative merits of our proposed system with respect to the performance of other recognition techniques, namely, bag-of-features (BoFs) and support vector machine (SVM).

We proceed with our discussion as follows. We discuss the cyber vulnerability of ROS in Section II. In Section III, we present the dynamics of our ground robot (the GVR-BOT) system while in Section III we discuss the framework of our cyber intrusion detection algorithm. In Section V, we present the ROS network traffic data for both legitimate and malicious operations in the form of time-domain data and RGB images. In Section VI, we highlight the efficacy of our deep learning cyber-intrusion detection algorithm before finally we conclude our discussion in Section VII.

II. Cyber Vulnerability of Robot Operating Systems (ROS) 1.0

Originally developed by Willow Garage in 2007, Robot Operating Systems (ROS) has become a set of widely implemented software libraries that can assist engineers to build multiple robot applications. The system facilitates the development of cutting-edge algorithms, that are easy to implement in real-time.

The initial development plan of ROS was intended for a smart humanoid 7-DoF personal robot (PR2) in mind, whose some salient characteristics include: being a single robot with no real-time requirements and for academic-research-related applications [19]. Ever since the use of ROS has gained significant overshot in popularity, and as such the system has been widely adopted in both military and civilian domains (e.g. autonomous farming [20], shopping robot [21], GVR-BOT [22], [23]) due to their numerous benefits as an open, modular, and flexible platform.

While these achievements are indeed good news, they are often two sides of the same coin as the ROS system was not designed to embrace potential cyber-security issues due to the following reason [19]:

1) There was no standard methodology for networking procedure (e.g. for multi-robot systems), despite its ability to deal with multi-robot networks. The current approach is merely based on a single-master ROS structure.
2) The system was not specifically designed to accommodate inter-process and inter-machine communication.
3) The system assumed an ideal WiFi connection and is not capable of dealing with various quality WiFi signals, such as poor quality WiFi causing loss or delay.
4) Most importantly, ROS 1.0 largely ignores security issues in its basic coding scheme.

Consequently, there are multiple examples of the vulnerability of ROS 1.0 as follows:

1) The network traffic data of a legitimate user is not encrypted, making it easy for an attacker to eavesdrop, record, or manipulate the data, e.g. performing man-in-the-middle attacks or false data injections.
2) the system also employs an anonymous graph-type structure, making it easier for an attacker to hack the system.
3) Each node within the graph has limited integrity-checking capability. As such, it is not possible to detect activities from potentially malicious users, which can also lead to the possibility of an attacker reverse-engineering the system without the host's knowledge.
4) The system also employs an open subscription system, meaning each node can act as a publisher or subscriber (see Fig. 2). As such, the system is vulnerable to packet sniffers and man-in-the-middle attacks, risking the integrity of the system.

Considering the openness and anonymous nature of ROS, we will discuss the typical ROS handshake process to establish a connection between a publisher and subscriber as given in Fig. 2. It describes the concept of information sharing among

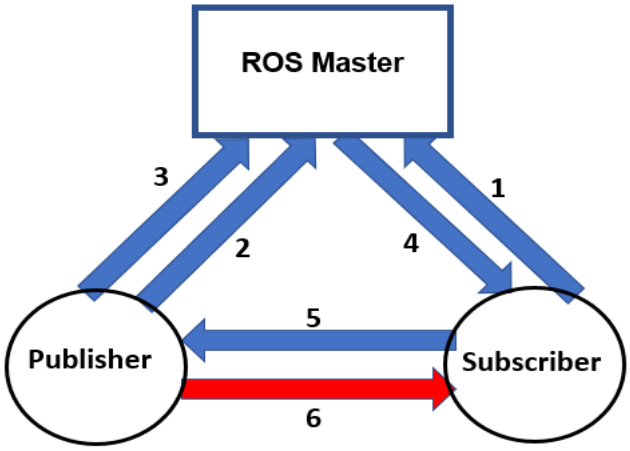


Fig. 2: The concept of ROS handshake (call set-up) in ROS 1.0 API environment for registering publisher and subscriber between two nodes [8]: (1) Registration process of the subscriber, (2) Parameter to be set by the publisher, (3) Registration of the publisher, (4) Update process, (5) Data transport to be requested, (6) Data to be transferred. The XMLRPC protocol is indicated by the blue line and the ROSTCP is highlighted by the red line.

various ROS 1.0 client libraries, implemented via XML-RPC. XMLRPC protocol is a remote call procedure (RPC) protocol with XML to encode its call and HTTP as the transport mechanism. ROSCTP is the transport layer for ROS messages

and services, employing standard TCP/IP sockets for sending the message data.

To begin with, the ROS Master with a known IP address needs to be executed in the system to keep track of the information. Next, the publishers inform the master of the topics they are publishing (e.g. $\backslash$cmd_vel) on certain localhost. The subscribers need to inform the ROS master that they want to subscribe to $\backslash$cmd_vel, and since the master has knowledge about active publishers on that topic, they can directly pass the information to the subscriber to indicate that: 'I know node X is publishing this topic, and it is located at the localhost: 1111'. The subscriber (e.g. node Y) can now establish a peer-to-peer ROS connection to the publisher.

Considering ROS-based commercial products in medical, industrial, and military robots, it is unfortunate that these vulnerabilities can pose serious cyber threats. Data from an open topic can be easily manipulated while sensitive information from any nodes can be reverse-engineered without the knowledge of the hosts. Meanwhile, nodes are also vulnerable to Denial-of-Service (DoS) attacks, where an attacker can also introduce processing delays, leading to the failure of the system to act in real-time. It is also not possible to verify the authenticity of the payload information in transit between nodes in ROS 1.0 [8].

Although ROS 2.0 has been developed to overcome the drawbacks of ROS 1.0, the message from the developer is clear [19] that ROS 2.0 will independently coexist with ROS 1.0 and is not meant to replace it. Therefore, it is important to develop a robust cyber intrusion detection algorithm to safeguard its operation.

III. The Dynamics of the GVR-BOT Ground Vehicle

Before conducting a real-time cyber-attack (penetration testing) on the 'GVR-BOT', a typical differential drive ground vehicle developed by the US Army as shown in Fig. 1, we will first study the normal operation of the system. This includes the dynamics of the robot as well as hardware and control loop configurations.

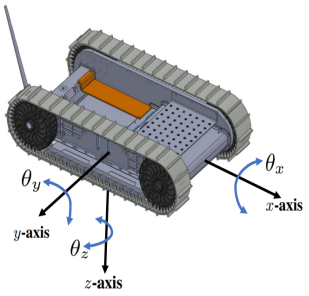
The system is programmable and it employs ROS 1.0, an open-source robotic middleware platform in its main processor. To control the system in the real-time domain, we also employ an operator control unit (OCU) software, that interfaced with the handheld transmitter. The system employs a user interface in the form of XML configuration files. It processes information such as the location, types of maps, video streaming, etc. Meanwhile, the input devices include the mapping of the joystick, keyboard, and mouse in the system. The system can serve two different operations, namely, monitoring and control. In what follows, we will discuss the dynamics of the robot.

A. System Dynamics

The dynamics of a ground vehicle can be represented by the following state space equation;

$$\begin{cases} \dot{x}(t) = f(x(t), u(t)), \ \forall t \geq 0, \\ x(0) = x_0, \end{cases} \tag{1}$$

where $x_0 \in \mathbb{R}^n$ denotes the initial condition of the states of $x(.) \in \mathbb{R}^n$, $u(.) \in \mathbb{R}^p$ indicates the control signals sent to the motor. Meanwhile, the evolution of the state is given by the curve: $x : [0 \ \infty) \rightarrow \mathbb{R}^n$.



h!

Fig. 3: The coordinate axis system of our GVR-BOT ground robot: (x, y, z) and the attitudes of the system are given by $(\theta_x, \theta_y, \theta_z)$.

The GVR-BOT vehicle belongs to differential drive systems. The thrust produced by the left and right electric motors accelerates the system along the x and y directions. For instance, to produce thrust that is aligned with the forward axis, both left and right motors need to have the same angular velocities. To create a turning moment, however, one motor needs to rotate faster than the other, projecting some thrust components along the y-axis. For instance, if the left motor moves faster than the right one (in the same direction), the robot will turn to the right and the other way round.

The dynamics of the system in terms of the linear and angular velocities can be represented as follows: [24], [25]:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\phi}_R \\ \dot{\phi}_L \end{bmatrix} = \begin{bmatrix} \frac{R}{2} cos\theta_z & \frac{R}{2} cos\theta_z \\ \frac{R}{2} sin\theta_z & \frac{R}{2} sin\theta_z \\ \frac{R}{2l} & -\frac{R}{2l} \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{\phi}_R \\ \dot{\phi}_L \end{bmatrix},$$

where R is the radius of the wheels, l is the distance between the center of gravity of the right and left wheels, $\omega_R = \dot{\phi}_R$ and $\omega_L = \dot{\phi}_L$ are the right and left angular velocities of the left and right wheels.

Considering Lagrange dynamics [24], the dynamics of the robot in (1)) can be represented using the following differential equation:

$$M(q)\ddot{q} + V(q, \dot{q})\dot{q} + F(\dot{q}) + G(q) + \tau_d + \epsilon = B(q)\tau - \alpha^T(q)\lambda, \tag{2}$$

where $M(q) \in \mathbb{R}^{(n \times n)}$ indicates a symmetric positive definite inertia matrix, $V(.)$ presents the centripetal and Coriolis matrix, $F(q)$ is the friction matrix, $G(q)$ is the gravitational forces, τ_d is the disturbance torque, ϵ is the unstructured dynamics B is the input matrix, and τ is the input actuation torque while q describes the states of the vehicles in terms of the generalized coordinates, i.e. $q := [x_r \ y_r \ \theta_z \ \omega_R \ \omega_L]^T$, in which (x_r, y_r) are the coordinates of the robot while ω_R and ω_L are the velocities of the wheels, $\alpha(.)$ denotes the constraint matrix and λ indicates the vector of the Lagrange multipliers associated with the constraints.

After some simplification, we finally arrive at [25]:

$$\begin{bmatrix} I_1 & I_2 \\ I_2 & I_1 \end{bmatrix} \begin{bmatrix} \ddot{\phi}_R \\ \ddot{\phi}_L \end{bmatrix} = \begin{bmatrix} \tau_R \\ \tau_L \end{bmatrix} - c \begin{bmatrix} \dot{\phi}_R \\ \dot{\phi}_L \end{bmatrix}, \tag{3}$$

where I_1 and I_2 present the inertial matrices of the robot, $\tau_{R,\ L}$ are the right and left actuation torques while c is a constant.

B. Hardware Configuration

The GVR BOT 1.2 employs the Intel Atom E680, 16 GHz processor, supported by 1GB Double Data Rate Random-Access Memory (DDR) 2 RAM, and non-volatile 2.6 GB NAND flash memory. As the main source of power, the system employs up to four BB-2590 Li-Ion rechargeable batteries. The robot employs a WiFi-enabled communication system at 2.4 GHz with a maximum range of 250 m. The system is supported by an Internal Attitude Heading Reference System (AHRS), giving acceleration, heading, and orientation information. It is also possible to include an internal GPS transceiver with an external antenna port. The net weight (without battery and flippers) of the robot is about 11.8 kg with a maximum speed of 2.0 m/s.

The robot is also equipped with four payload ports to add sensors, processors, radios, or other devices. However, the availability of power and communication is the main consideration for adding an extra payload. The system runs under ROS architecture, meaning the system needs to execute the **roscore** command on its internal processor to interact with it.

IV. The Framework of Our Cyber-Intrusion Detection Systems

Robotic systems can be compromised at multiple different levels, namely, at the system, sub-system, component, or sub-component levels. For instance, an attack at the signal processing level is highly appropriate when it comes to a sensory system such as LIDAR (Light Detection and Ranging) or a vision system to compromise the distance measurements [26]. Preventing these attacks is by no means trivial, especially for sophisticated, complex, and modern robots, which can work even under a fault-tolerant mode, blurring the line between normal operations and fault conditions.

We will investigate the network traffic data of the robot to study its operations, and whether or not the system has been compromised or not. There are some possibilities for such an approach such as using a node-based method, application-based technique, or flow-statistic-based system.

Node-based methods employ information obtained from other nodes while application-based techniques investigate the patterns inside the information packets. On the other hand, flow-statistic methods rely on the metadata of the packet header (e.g. bit transmitted, packet inter-arrival times, window size, etc). Overall, application-based systems turn out to be more reliable despite being more complex due to their reliance on an up-to-date database of the patterns that must be constantly maintained, especially for encrypted data as in the S-ROS traffic flow.

The purpose of developing a fault detection filter is to detect any potential malicious activities attempting to compromise the integrity of the system and its data. If there is something unexpected with the robot (a potential fault or cyber-attack), one must first identify the problem, before safely leaving the robot. As such, we would like our system to have the capability of detecting each possible source of faults and dealing with them accordingly.

A. Mathematical Problem Statement

Our ground robot employs two feedback control loops to regulate its forward velocity and heading. Fig. 4 describes a general high-level block diagram of the interaction between humans and guidance and control systems in an autonomous system. A potential intruder may want to compromise the sensory data or guidance signal in an attempt to electronically hijack the autopilot system e.g. to deviate the robot from its intended trajectory.

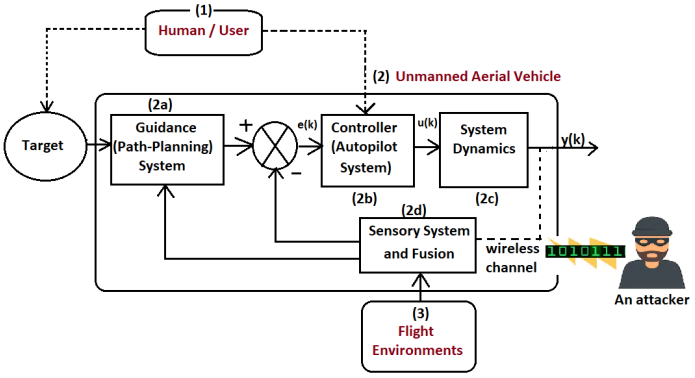


Fig. 4: Block diagram of potential robotic cyber-attack at the system level. Malicious users may compromise the integrity of the system or signals at any nodes.

The nonlinear dynamics of any cyber-physical system can be represented using the following non-linear state space equation:

$$\begin{cases} \dot{x}_{i,j}(t) = f_{i,j}(x_{i,j}(t)) + g_i(x_{i,j}(t))u_i(t) + \delta_{i,j} + d_{i,j}(t),\ \forall t \geq 0, \\ y_i(t) = x_{i,j}(t) + \delta_o(t),\ i = 1, 2, ...N,\ j = 1, 2, ..., M-1, \end{cases} \tag{4}$$

where $x_{i,j} \in \mathbb{R}^j$ is the bounded state vectors of the system within its maximum and minimum range, namely, $\underline{x}_{i,j} \leq x_{i,j} \leq \overline{x}_{i,j}$. $\underline{x}_{i,j}$ may be compromised by $\delta_{i,j}$, a vector of malicious attacks on a particular system state, $(u_i,\ y_i) \in \mathbb{R}$ are the bounded control inputs and the system outputs within an interval of $\underline{u}_i(k) \leq u_i(k) \leq \overline{u}_i(k)$ while $\underline{y}(k) \leq y(k) \leq \overline{y}(k)$ denotes the vector of bounded system output, which may also be corrupted by $\delta_o(k)$, a vector of malicious attack attempting to spoof the output of the system. Meanwhile, $f_{i,j}(.)$ and $g_{i,j}(.)$ are the unknown functions describing the non-linear dynamics of the systems and $d_{i,j}$ is the external disturbance. Please note that i and j denote the input and output indices, where N and M are the maximum values of i and j.

Considering the universal approximation theorem [27], there exists an ideal deep learning neural network estimator, represented by f^*, such that the estimation error of system dynamics in (4), as indicated by $\epsilon(t) = ||w^T G(.) - w^{*^T} G(.)||_2$ is minimized, that is, $lim_{t\to\infty}\ \epsilon(t) = 0$, which can only be achieved if and only if $lim_{t\to\infty}\ ||w(t)^T - w(t)^{*^T}||_2 = 0$. It should be noted that $\hat{f} = w^T G(.)$, where $w = [w_1, w_2, ..., w_n]^T$ is a set of the weighting vectors and $G(.)$ is the activation function of the neural networks as a subset of our CNN filters.

B. Deep Learning CNN and Voting Filter

Deep learning convolutional neural networks (CNNs) employed in our research contain several layers that can be described in more detail as follows:

- **Interface and input layers**: We introduce an interface layer to bridge between the input of the CNN filter in the form of images and the output of the ROS system, which produces large-scale time series data, describing the dynamics of the ground robot. In doing so, firstly, the ROS network traffic data containing 33 features will be normalized at each sampling time with respect to $\pm$ 1 using the min-max normalization method, so that the overall values fall within $[-1, +1]$ using equation (5):

$$x_n = a + \left(\frac{x - min(x)(b-a)}{max(x) - min(x)} \right), \tag{5}$$

  where a, b are the min-max values ($a = -1$, $b = 1$). Accordingly, we will have a large normalized time-series data set obtained from the ROS system.
  To read the data epochs, we propose two ways of sliding windowing techniques in the form of independent and overlapping techniques. Please refer to Fig. 5 for visual representations. While in the independent technique, all information is new at every window, in overlapping windows, there are some overlapping data. Having normalized and partitioned the data, we convert them into the form of RGB or grayscale images as input for the CNN system. The CNN needs information about the size of the image, namely, heights, widths, and the number of channels (one for RGB and three for RGB images).

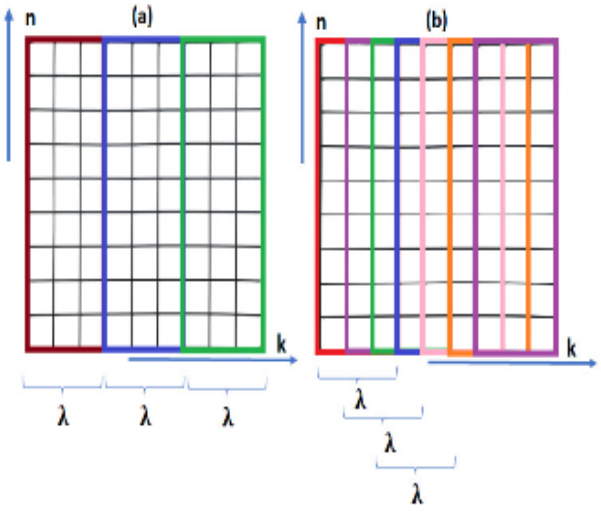


Fig. 5: Fixed-length sliding windowing techniques, containing two major schemes, namely, (a) Left Figure: independent (no data overlapping) windowing technique and (b) Right Figure: overlapping windowing technique with the distance of one epoch epochs with the neighboring segments and overlapping data of $(\lambda - 1)$. In this example, we use $\lambda = 3$. While the vertical axis indicates the system states, the horizontal axis presents the concept of sampling time.

- **2D-Convolutional Layers** This layer serves as the main processing layer in CNN as the system applies sliding convolutional filters to 2-D input. The filter is moved along the input both vertically and horizontally to compute the dot product and the weights and the input while adding a bias term.
- **Batch Normalization Layer** is to allow every piece of the networks to perform independent learning. This way, the learning process can be stabilized while the speed can be improved by adding extra layers to the system.
- **Rectified Linear Unit (ReLU) Layer**: is to set the threshold of operation to each input element by giving any negative values to become zero, that is,

$$f(x) = \begin{cases} x, & x \geq 0 \\ 0, & x < 0. \end{cases} \tag{6}$$

- **Cross Channel (Local Response) Normalization**: is to perform a channel-wise local response stabilization following the ReLu activation function.
- **Max and Average Pooling Layer** is to downsample the data by defining a rectangular area and computing the maximum value of the region. Max pooling leads to the most prominent feature in the map while average pooling gives the mean of features present.
- **Fully Connected Layer** is a typical neural network layer connecting each input neuron to every output neuron. This layer employs weights W that multiply the input signals while scaling them by a bias b. As such, the main pattern of the information obtained from the previous layer across the image can be identified.
- **Output Layer** is suitable for classification problems where the CNN system needs to have recognition or detection capabilities.

C. Voting Filter

To minimize the glitches (instantaneous errors) in the detection outcome of our CNN filter, we employ a voting filter. The idea is to vote using the voice of the majority before making a final decision on whether or not the traffic data are legitimate or malicious. Mathematically, the decision is valid if it is supported by at least κ consecutive outcomes of the CNN filter, where $\kappa > 1$. The lengthy size of the voting filter is undesirable as it will affect the overall detection time. In our experiment, we achieve a delicate balance between time and accuracy by setting $\kappa = 3$.

V. Real-time Man-in-the-Middle Cyber Attacks

Man-in-the-middle-attack is a class of cyber-attack, where the traffic or data communication between two parties (e.g. the legitimate host and the victim) is intercepted for a certain purpose, such as to inject false data (modify the traffic data), to simply passively listen (eavesdrop) the traffic, to completely terminate the communication, or to set up a malicious connection to a different destination. In this research, however, we focus on overwriting the information of certain ROS traffic data using the unintended traffic data from different nodes.

In more detail, we aim to overwrite the guidance information received from the handheld transmitter by the /jaus_mobility node as broadcast in the form /cmd_vel ROS topic to the /gvrbot_mobility node, which is responsible for controlling the motion of the robot. This can be done by

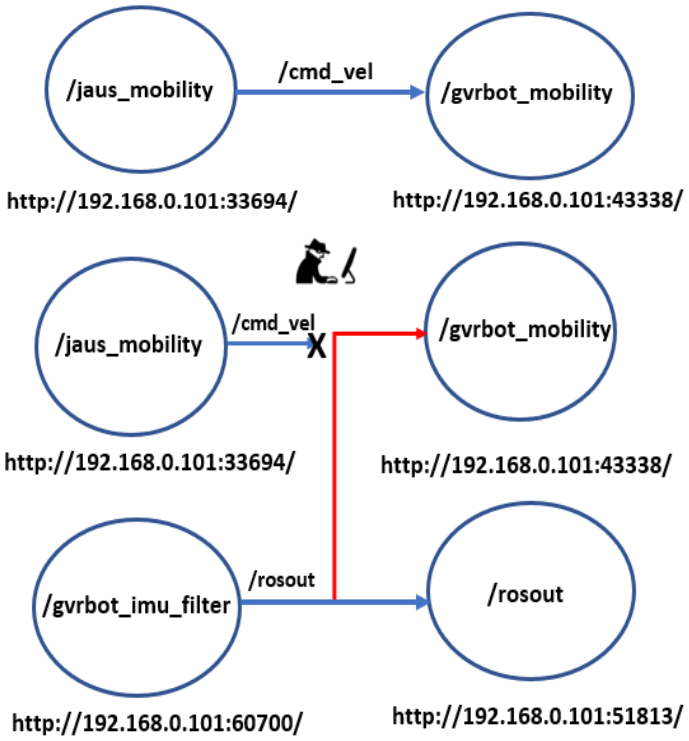


Fig. 6: ROS communications: (a) Top figure: Legitimate condition, where the information from /jaus_mobility node into the form of /cmd_vel ROS topic goes directly to the /gvrbot_mobility node, (b) Bottom figure: man-in-the-middle cyber attack scenario on the GVR-BOT: the /cmd_vel topic from /jaus_mobility node sent to /gvrbot_mobility is overwritten by the unintended traffic data from the /rosout topic sent by the /gvrbot_imu_filter node.

taking the information from the /gvrbot_imu_filter to overwrite the information by the /jaus_mobility node so that the /gvrbot_mobility receives false data during the course of a cyber attack (See Fig. 6).

We attacked an important node ((\cmd_vel) ROS topic) using RosPenTo [8]. The node is responsible for command and control in the GVR-BOT vehicle. As such, the overall mobility of the system was disabled. Please note that there are two control loops in the GVR-BOT vehicle, namely the forward velocity and the heading control loops. \cmd_vel topic contains information for both loops. Considering some relevant work on ROS penetration testing, interested readers are suggested to refer to [8], [28], and [29].

VI. ROS Network Traffic Data and The Performance of Our Intrusion Detection System

Our ground robots were wirelessly connected to two separate computers via a local WiFi network. While the first computer acted as a ground control station for the GVR-Bot, executing the Multi-Operator Control Unit (MOCU) software to send the control signal from the hand-held transmitter, the second computer, run under the Ubuntu Linux O/S, recorded the network traffic data by subscribing to all relevant ROS topics broadcast by the GVR-BOT's onboard computer. We recorded the experimental network data in the form of a multivariate time domain.

A. Time Domain Representation

We recorded our experimental ROS network traffic data within 10 minutes (600 s) with a sampling time T of 0.1s, making up a total duration of 6000 epochs. The man-in-the-middle cyber-attack approximately occurs at t=300s, splitting the data into two main parts, namely, the legitimate and malicious operations. The typical time domain representation of the system state of our GVR-BOT system is given by Figs. 7, 8, 9.

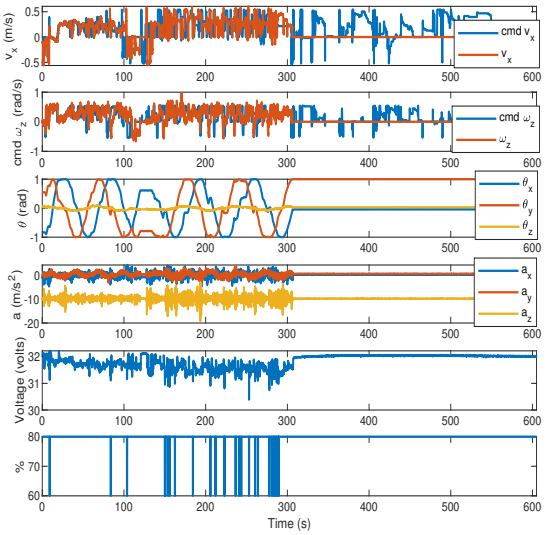


Fig. 7: ROS Network traffic data for outdoor operation of the GVR-BOT under man-in-the-middle attacks. The cyber attack occurs at $t = 300s$. System states of the GVR-BOT describe: (a) the forward velocity and its reference signal (first row), (b) angular velocity and its reference signal (second row), and (c) the direction of the vehicle along the (x, y, z) axes (third row), (d) acceleration (fourth row), (e) battery voltage (fifth row), (f) the percentage of power (sixth row).

As seen from Fig. 7, as soon as a cyber-attack occurs at t=300 s, the ground robot became irresponsive to the command control. Despite still being able to perceive the guidance signal transmitted from the handheld transmitter, the robot was unable to follow it. This is because the guidance signal that should be transmitted to /gvrbot_mobility ROS node was overwritten by the unintended traffic data from /ros_node.

As such, from the systems and control point of view, the robot was made blind with respect to the legitimate reference signal. This way, the attacker can also inject false data regarding the command signal to compromise the intended trajectory of the system.

We used the first half of each data to train our CNN while the rest is to validate the performance of our system. We should also highlight that to mimic the real-time scenario, we operate our ground robot in both indoor and outdoor environments, representing at least 9 different motions, namely, moving forward and backward, turning left (forward left), turning right (forward left), spin left, spin right as well as backward left, backward right, and no motions (remains stationary).

This article has been accepted for publication in IEEE Transactions on Dependable and Secure Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TDSC.2023.3302807

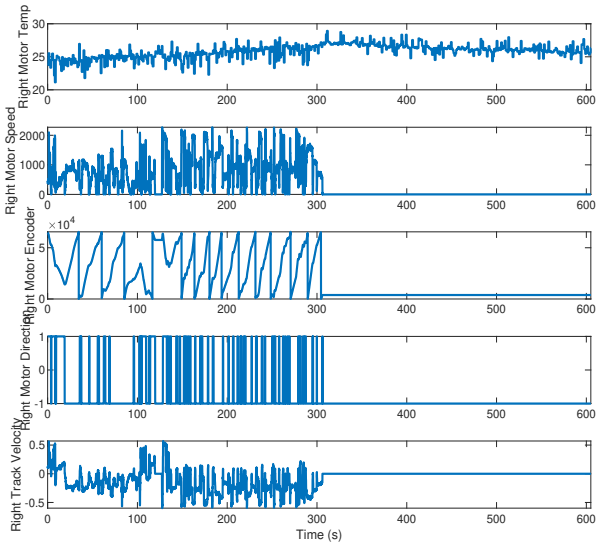


Fig. 8: ROS Network traffic data for outdoor operation of the GVR-BOT under man-in-the-middle attacks. The cyber attack occurs at $t = 300s$. System states of the GVR-BOT describe the state of the right motors, namely, (a) temperature, (b) speed, (c) encoder, (d) direction, and (e) track velocity.

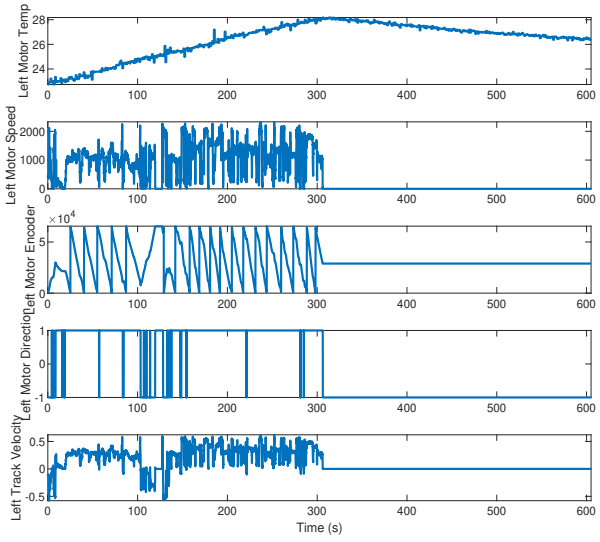


Fig. 9: ROS Network traffic data for outdoor operation of the GVR-BOT under man-in-the-middle attacks. The cyber-attack occurs at $t = 300s$. System states of the GVR-BOT describe the state of the left motors, namely, (a) temperature, (b) speed, (c) encoder, (d) direction, and (e) track velocity.

B. RGB Images

After normalizing the time domain signals (see Figs. 7, 8, 9), we can convert them in the form of RGB images or grayscale suitable as the input of our cyber intrusion detection filter as given by Fig. 10.

Moreover, the network traffic data can be split into smaller chunks following the input sliding window as depicted in Fig. 10. It is apparent that there are certain color configurations (patterns) that differentiate legitimate operations from their malicious counterparts.

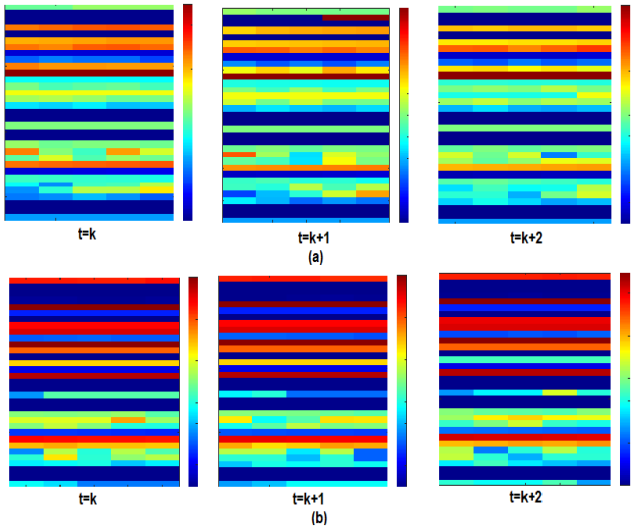


Fig. 10: Windowed network traffic data ($\lambda = 5$), representing a chunk of block data of the RGB images for training and detection of deep learning convolutional neural network (CNN): (a) Top figure describes the normalized windowed network traffic data of ROS under legitimate (normal) operation, (b) Bottom figure presents the normalized windowed network traffic data of ROS under malicious (cyber-attack) operation.

Each vertical row in Fig. 10 depicts the mechanical and electrical states of the robots as described in Table. I. Overall there are 33 states observed by the systems that can be split into three major parts, namely, the overall states of the vehicle, the states of the right tracks, and the states of the left tracks.

C. Statistical Performance

The statistical performance of our cyber intrusion detection system is highlighted in Tables II, III, IV, and V, which are also represented in Figs. 11-17.

As can be seen, the performance of the system with the independent windowing (IW) technique outperforms the performance of the overlapping windowing technique (OW) (see Figs. 14-17). Intuitively, the independent windowing technique can provide more information in each sampling time since it contains no repetitive information with the neighboring windows (segments). This way, prediction accuracy can be increased compared to the overlapping segments.

Also, it turns out that the presence of a voting filter greatly improves the performance of each technique. As the amount of information fed to the system increases (given by the size of epochs), it is also natural to expect an increase in accuracy, precision, TNR, and F_1 score in general.

TABLE I: Nomenclature of each row vectors (features) as presented in the RGB image in Fig. 10

Rows #	Variables
1.	Battery voltage (volts)
2.	Input current (Amps)
3.	Percentage battery power (%)
4.	Left motor current (Amps)
5.	Left motor direction (± 1)
6.	Left motor encoder count
7.	Left motor speed (RPM)
8.	Left motor temperature ($^\circ$C)
9.	Left track velocity (m/s)
10.	Right motor current (Amps)
11.	Right motor direction (± 1)
12.	Right motor encoder count
13.	Right motor speed (RPM)
14.	Right motor temperature ($^\circ$C)
15.	Right track velocity (m/s)
16.	Velocity across the x-axis v_x (m/s)
17.	Velocity across the y-axis v_y (m/s)
18.	Velocity across the z-axis v_z (m/s)
19.	Reference of the linear velocity across the x-axis $\tilde{v}_x$ (m/s)
20.	Reference of the linear velocity across the y-axis $\tilde{v}_y$ (m/s)
21.	Reference of the linear velocity across the z-axis $\tilde{v}_z$ (m/s)
22.	Acceleration across the x-axis (m/s^2)
23.	Acceleration across the y-axis (m/s^2)
24.	Acceleration across the z-axis (m/s^2)
25.	Vehicle orientation across the x-axis θ_x (rad)
26.	Vehicle orientation across the y-axis θ_y (rad)
27.	Vehicle orientation across the z-axis θ_z (rad)
28.	Angular velocity across the x-axis ω_x (rad/s)
29.	Angular velocity across the y-axis ω_y (rad/s)
30.	Angular velocity across the z-axis ω_z (rad/s)
31.	Reference of the angular velocity across the x-axis $\tilde{\omega}_x$ (rad/s)
32.	Reference of the angular velocity across the y-axis $\tilde{\omega}_y$ (rad/s)
33.	Reference of the angular velocity across the z-axis $\tilde{\omega}_z$ (rad/s)

On the other hand, FPR constantly decreases as the amount of information (epochs) increases. It is interesting to note that regardless of the number of data samples (epochs), our system can constantly achieve perfect TPR and FNR as both values constantly hover at 1.000. This way, we can claim that our system is highly secure since our intrusion detection system can detect all malicious attacks.

TABLE II: The performance of our proposed cyber intrusion detection system using **independent epochs** (windows) without a voting filter

Epochs	Acc	TPR	FNR	FPR	TNR	Prec	F1
1	0.982	1.000	0.000	0.036	0.964	0.966	0.983
2	0.988	1.000	0.000	0.022	0.978	0.978	0.989
3	0.990	1.000	0.000	0.020	0.980	0.980	0.990
5	0.978	1.000	0.000	0.004	0.996	0.996	0.998
10	0.997	1.000	0.000	0.007	0.993	0.993	0.997

TABLE III: The performance of our proposed cyber intrusion detection system using **independent epochs** (windows) with the length of the voting filter $\kappa = 3$ consecutive decisions.

Epochs	Acc	TPR	FNR	FPR	TNR	Prec	F1
1	0.992	1.000	0.000	0.016	0.984	0.985	0.992
2	0.996	1.000	0.000	0.009	0.991	0.991	0.996
3	0.996	1.000	0.000	0.009	0.991	0.991	0.996
5	1.000	1.000	0.000	0.000	1.000	1.000	1.000
10	1.000	1.000	0.000	0.000	1.000	1.000	1.000

To help the readers visualize the trend, we will convert numerical data from Tables 7, 8, 9 into statistical performance graphs describing accuracy (Fig 11), precision (Fig. 12), true positive rate (Fig. 13), true negative rate (Fig. 14), F_1 score (Fig. 15), false negative rate (Fig. 16), and false-positive rate (Fig. 17). This way enables direct comparison regarding the performance of the system with independent windows relative to the performance of the system with overlapping windows as well as the relative benefits of adding a voting filter to the system.

TABLE IV: The performance of our proposed cyber intrusion detection system using **overlapping epochs** (windows) without a voting filter.

Epochs	Acc	TPR	FNR	FPR	TNR	Prec	F1
2	0.972	1.000	0.000	0.056	0.944	0.947	0.973
3	0.986	1.000	0.000	0.029	0.971	0.972	0.986
5	0.982	1.000	0.000	0.036	0.964	0.966	0.983
10	0.987	1.000	0.000	0.024	0.976	0.976	0.988

TABLE V: The performance of our proposed cyber intrusion detection system using **overlapping epochs** (windows) with the length of the voting filter $\kappa = 5$ consecutive decisions.

Epochs	Acc	TPR	FNR	FPR	TNR	Prec	F1
2	0.987	1.000	0.000	0.027	0.973	0.974	0.987
3	0.988	1.000	0.000	0.024	0.976	0.976	0.988
5	0.990	1.000	0.000	0.020	0.980	0.980	0.990
10	0.990	1.000	0.000	0.020	0.980	0.980	0.990

TABLE VI: The performance of the 'bag-of-features' detection algorithm as a comparison.

Epochs	Acc	TPR	FNR	FPR	TNR	Prec	F1
1	0.945	0.940	0.060	0.050	0.950	0.950	0.940
2	0.950	0.930	0.070	0.020	0.980	0.980	0.950
3	0.960	0.940	0.060	0.020	0.980	0.980	0.960
5	0.970	0.990	0.010	0.050	0.950	0.950	0.970
10	0.970	0.990	0.010	0.060	0.904	0.940	0.970

Accuracy is defined as the ratio of the correctly identified users to the total number of samples, ie., $Acc = \frac{TP+TN}{TP+TN+FP+FN}$. As seen from Fig. 11, all configurations achieve reasonably high accuracy ≥ 97 % with a period of two epochs. However, by having independent sliding windows, the system can achieve an accuracy of ≥ 98 % with as little as one epoch of information.

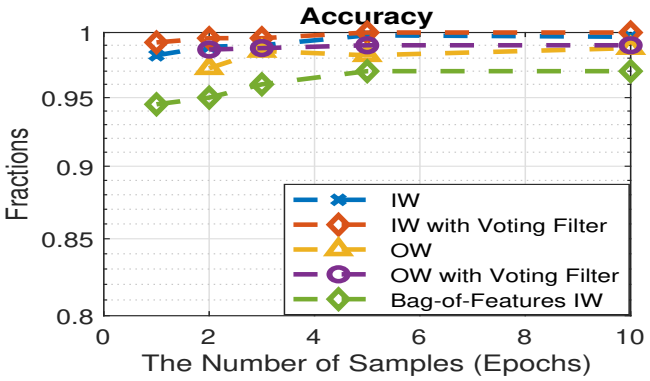


Fig. 11: The accuracy of the systems under different windowing techniques. While 'IW' denotes independent windowing techniques, 'OW' stands for overlapping windows.

In terms of precision, namely, $P = \frac{TP}{TP+TF}$, the performance of the overlapping window system hovers at around 94

% (without a voting filter), though it can be boosted to around 97 % with the assistance of a voting filter. On the other hand, an independent window system has a better precision at around 96 % and 98 % for systems without and with a voting filter.

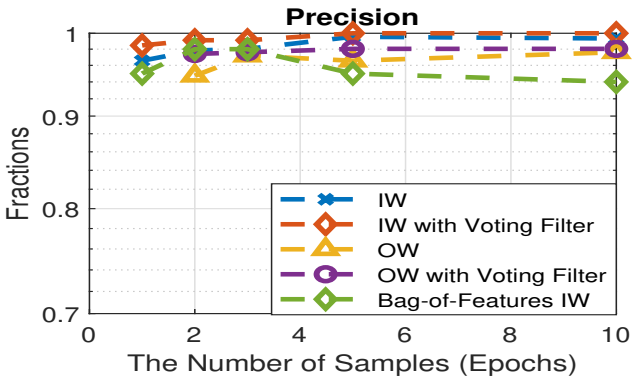


Fig. 12: The precision of the system under multiple windowing conditions. While 'IW' denotes independent windowing techniques, 'OW' stands for overlapping windows.

Fig. 13 suggests that both systems can achieve perfect true positive rates (also known as sensitivity as defined by $TPR = \frac{TP}{TP+FN}$, meaning there is no room for malicious data being correctly identified as legitimate. This indicates the highly secure nature of our system.

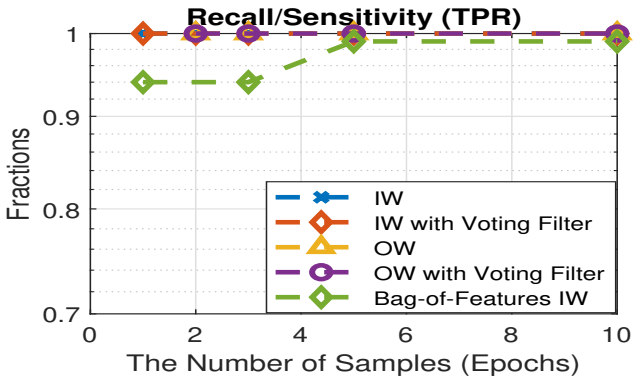


Fig. 13: True Positive Rates (TPR) as a function of epochs under different windowing schemes. While 'IW' denotes independent windowing techniques, 'OW' stands for overlapping windows.

Moving on to Fig 14, one can see the true negative rate as defined by $TNR = \frac{TN}{TN+FN}$ of the systems, suggesting the percentage of legitimate data being correctly identified. Likewise, the figures for independent windows are better than those employing overlapping windows with a margin of around 2 %.

Moreover, Fig 15 depicts the F_1 score of the systems (as defined by $F_1 = 2\frac{Precision \times Recall}{Precision+Recall} = \frac{TP}{TP+\frac{1}{2}(FP+FN)}$, giving the harmonic mean of precision and recall. It turns out that independent windows outperform overlapping windows with a margin of about 2 % at the lower number of epochs $\lambda \leq 5$. The difference is however less noticeable (about 1 %) at a higher number of epochs $\lambda \geq 6$.

While the accuracy, indicating the total number of correct predictions divided by the total number of predictions) is suitable for a well-balanced class of data, it is not ideal

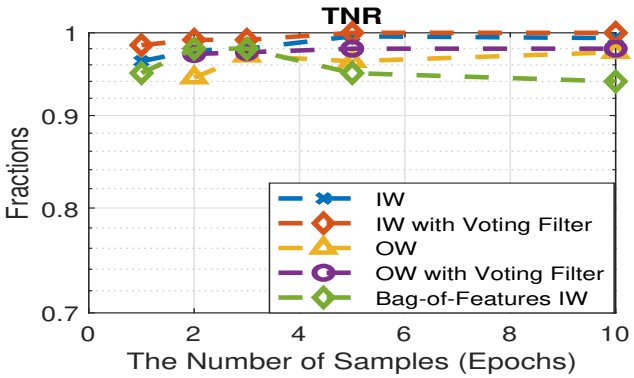


Fig. 14: True Negative Rates (TPR) as a function of epochs under different windowing schemes. While 'IW' denotes independent windowing techniques, 'OW' stands for overlapping windows.

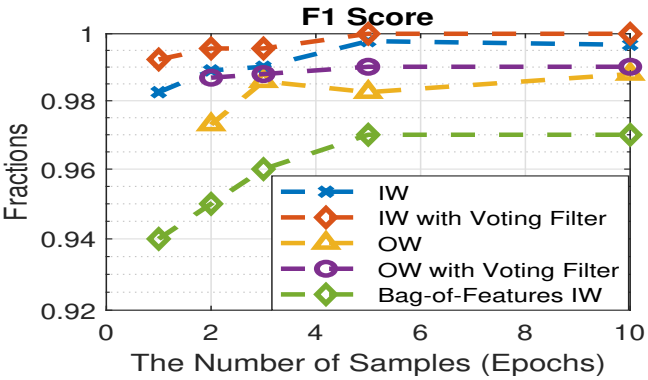


Fig. 15: F1 scores as a function of epochs under different windowing schemes. While 'IW' denotes independent windowing techniques, 'OW' stands for overlapping windows.

for imbalanced data. Thus, we also employed the F_1 score, describing the harmonic mean of precision and recall. It is a more appropriate metric to identify the minority class as in the case of data imbalance.

As such, poor prediction of the minority class will cause an increase in the F_1 score since it keeps the balance between precision and recall. Improved F_1 score can only be achieved if the system identifies more of a certain class correctly. Accordingly, the F_1 score only increases if both the number and quality of prediction improve. In our case, we can be confident with the ability of the system to deal with such a case since the F_1-scores of the system are reasonably high.

Fig. 16 highlights the false negative rates of the system as defined by $FNR = \frac{FN}{TP+FN}$. As can be seen, they can achieve a perfect minimum score regardless of the length of the epochs, confirming the value of TPR in Fig. 13. Again, it means no malicious data being misclassified as legitimate (negative), or the system can perfectly capture all malicious attacks, reconfirming the robustness of the system.

Fig. 17 describes the false-positive rates of the systems $FPR = \frac{FP}{FP+TN}$ as a function of epochs. FPR highlights the number of misidentified legitimate users as malicious. As can be seen, the systems employing independent windows outperform the performance of systems employing overlapping windows with a margin of 4 % while employing a voting filter can suppress the false positive rate by a factor of 2 %. The

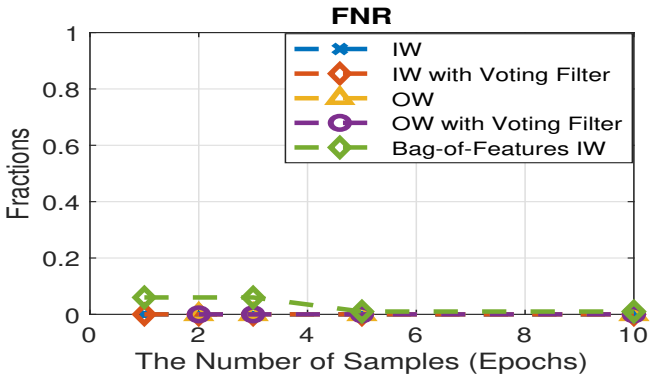


Fig. 16: False Negative Rates (FNR) as a function of epochs under different windowing schemes. While 'IW' denotes independent windowing techniques, 'OW' stands for overlapping windows.

higher the number of epochs, the discrepancy in FPR is more noticeable.

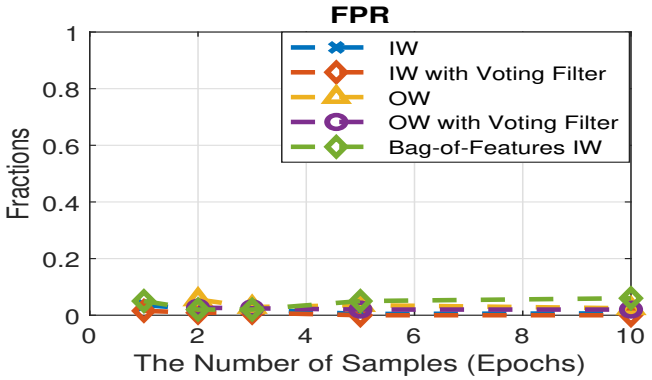


Fig. 17: False Positive Rates (FPR) as a function of epochs under different windowing schemes. While 'IW' denotes independent windowing techniques, 'OW' stands for overlapping windows.

To highlight the relative merits of our intrusion detection algorithm, we also conducted a rigorous comparative study with respect to the bag of visual words or bag-of-features (BoF) detection algorithm widely used in image processing (see Table VI and Figs. 12-16) for a visual comparison. The algorithm in general contains three major integral parts, namely, feature extraction, codebook generation, and feature vector generation. Interested readers are recommended to study in [30] for details. Our research indicates the superiority of our proposed framework with respect to the BoF detection algorithm widely implemented in image processing.

In Table VII, we discuss the performance of our proposed cyber security systems, namely, the independent window approach supported by a voting filter with $\kappa = 3$ (framework-1) and the independent window approach without a voting filter (framework-2), relative to the performance of the well-known bag-of-feature detection algorithm (BoF), and Support Vector Machine (SVM).

As can be seen, among seven statistical measures, our proposed algorithms demonstrate their superiority. The independent window approach (Framework 1) supported by a voting filter demonstrates its best performance closely followed by the independent window technique without a voting filter (Framework-2).

TABLE VII: Direct one epoch comparison of our proposed security framework in the face of data imbalance, namely, Framework-1 for a system with voting filter $\kappa = 3$, and Framework-2 for the system without a voting filter, with respect to the performance of the Bag-of-Feature (BoF), and Support Vector Machine (SVM)

Parameters	Framework-1	Framework-2	Bag-of-Features	SVM
Accuracy	0.992	0.982	0.945	0.912
TPR	1.000	1.000	0.940	1.000
TNR	0.984	0.964	0.950	0.845
FPR	0.016	0.036	0.050	0.155
FNR	0.000	0.000	0.060	0.000
Precision	0.985	0.966	0.950	0.830
F_1	0.992	0.983	0.940	0.907

Meanwhile, the SVM algorithm performs better with respect to the BoF detection algorithm in two categories, namely, TPR and TNR while BoF outperforms SVM in FPR, TNR, Precision, and F_1. Our proposed system also turns out to be superior compared to the performance of the existing systems as indicated by the highest accuracy, TPR, TNR, precision, and the F_1-score while achieving the lowest FPR and FNR.

VII. Conclusion

We have performed real-time cyber attacks (penetration testing) on Robot Operating Systems (ROS) applied on the GVR-BOT ground vehicle. We collected ROS network traffic data to train the CNN system under both legitimate conditions as well as malicious cyber attacks.

Our research confirms the efficacy, security, and practicality of the proposed cyber-intrusion detection algorithm. In fact, our detection algorithm is highly secure as proven by reasonably high accuracy $\rightarrow 1$, representing TPR (sensitivity) $\rightarrow 1$ and TNR (specificity) $\rightarrow 1$. The system can also achieve a $F_1 \rightarrow 1$ score (indicating near-perfect precision and recall values). Its practicality is also supported by reasonably small FPR $\rightarrow 0$ and reasonably short detection time within 3 consecutive epochs or less. Our proposed framework also performs better compared to the well-known BoF, and SVM detection algorithms widely used for image classification.

For future work, we would like to modify the structure of the CNN algorithm to deal with limited training of data. We are also interested in investigating the efficacy of our intrusion detection system on different robotic platforms, such as unmanned aerial vehicles, whose dynamics are reasonably faster and more complex compared to a ground robot. Under the umbrella of deep learning (supervised and unsupervised) systems, we are also keen to study the relative merits of our CNN intrusion detection algorithm with respect to similar detection techniques such as using evolving type-2 fuzzy systems, that can accommodate the footprint-of-uncertainties.

Acknowledgment

We would like to thank Mr. Timothy W. Pietrzyk, an Electrical Engineer of the Combat Capabilities Development Command (CCDC), Ground Vehicle Robotics (GVR), US Army for his technical advice on the GVR-BOT ground robot.

REFERENCES

[1] H. A. Abbass, E. Petraki, K. Merrick, J. Harvey, and M. Barlow. Trusted autonomy and cognitive cyber symbiosis: Open challenges. *Cognitive Computation*, 8:385–408, December 2016.

[2] G. W. Clark, M. V. Doran, and T. R. Andel. Cybersecurity issues in robotics. In *IEEE Conference on Cognitive and Computational Aspects of Situation Management (CogSIMA)*, pages 1–5, Savannah, GA, USA, 2017. IEEE.

[3] C. G. L. Krishna and R. R. Murphy. A review on cybersecurity vulnerabilities for unmanned aerial vehicles. In *IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*, pages 194–199, Shanghai, China, 2017. IEEE.

[4] R. S. Batth, A. Nayyar, and A. Nagpal. Internet of robotic things: Driving intelligent robotics of future - concept, architecture, applications and technologies. In *4th International Conference on Computing Sciences (ICCS)*, pages 151–160, Jalandhar, India, 2018. IEEE.

[5] L. Romeo, A. Petitti, R. Colella, G. Valecce, P. Boccadoro, A. Milella, and L.A. Grieco. Automated deployment of iot networks in outdoor scenarios using an unmanned ground vehicle. In *IEEE International Conference on Industrial Technology (ICIT)*, pages 369–374, Buenos Aires, Argentina, 2020. IEEE.

[6] F. Santoso, M. A. Garratt, and S. G. Anavatti. State-of-the-art intelligent flight control systems in unmanned aerial vehicles. *IEEE Transactions on Automation Science and Engineering*, 15(2):613–627, February 2018.

[7] N. Goerke, D. Timmermann, and I. Baumgart. Who controls your robot? an evaluation of ros security mechanisms. In *7th International Conference on Automation, Robotics and Applications (ICARA)*, pages 60–66, Prague, Czech Republic, 2021. IEEE.

[8] B. Dieber, B. Breiling, S.Taurer, S. Kacianka, S. Rass, and P. Schartner. Security for the robot operating system. *Robotics and Autonomous Systems*, 98:192–203, 2017.

[9] P. M. Lima, M. V. S. Alves, L. K. Carvalho, and M. V. Moreira. Security of cyber-physical systems: Design of a security supervisor to thwart attacks. *IEEE Transactions on Automation Science and Engineering*, May 2021.

[10] F Santoso. Range-only distributed navigation protocol for uniform coverage in wireless sensor networks. *IET Wireless Sensor Systems*, 5:20–30, 2014.

[11] F. Santoso. A decentralised self-dispatch algorithm for square-grid blanket coverage intrusion detection systems in wireless sensor networks. In *IEEE Vehicular Technology Conference (VTC Fall)*, pages 1–5, San Francisco, CA, USA, 2011. IEEE.

[12] F. Santoso. A new framework for rapid wireless tracking verifications based on optimized trajectories in received signal strength measurements. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(11):1424–1436, November 2015.

[13] F. Santoso and A. Finn. A data-driven cyber–physical system using deep-learning convolutional neural networks: Study on false-data injection attacks in an unmanned ground vehicle under fault-tolerant conditions. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 53(1):346–356, January 2023.

[14] S. Safaoui V. Renganathan, K. Fathian and T. Summers. Spoof resilient coordination in distributed and robust robotic networks. *IEEE Transactions on Control Systems Technology*, 2021.

[15] Z. Qu Y. Joo and T. Namerikawa. Resilient control of cyber-physical system using nonlinear encoding signal against system integrity attacks. *IEEE Transactions on Automatic Control*, 2020.

[16] R. Ma, P. Shi, and L. Wu. Dissipativity-based sliding-mode control of cyber-physical systems under denial-of-service attacks. *IEEE Transactions on Cybernetics*, 51(5):2306–2318, May 2021.

[17] C. Wu, L. Wu, J. Liu, and Z. P. Jiang. Active defense-based resilient sliding mode control under denial-of-service attacks. *EEE Transactions on Information Forensics and Security*, 15:237–249, May 2020.

[18] J. H. Cheon, K. Han, S.-M. Hong, H. J. Kim, J. Kim, S. Kim, H. Seo, H. Shim, and Y. Song. Toward a secure drone system: Flying with real-time homomorphic authenticated encryption. *IEEE Access*, 6:24325–24339, May 2018.

[19] B. Gerkey. Why ROS 2? https://design.ros2.org/, 2017.

[20] A. Durand-Petiteville, E. Le Flecher, V. Cadenat, T. Sentenac, and S. Vougioukas. Tree detection with low-cost three-dimensional sensors for autonomous navigation in orchards. *IEEE Robotics and Automation Letters*, 3(4):3876–3883, 2018.

[21] H. Su, Y. Zhang, J. Li, and Jie Hu. The shopping assistant robot design based on ros and deep learning. In *2nd International Conference on Cloud Computing and Internet of Things (CCIOT)*, pages 173–176, Dalian, China, 2016. IEEE.

[22] C. Korpela, K. Chaney, and P. Brahmbhatt. Applied robotics for installation and base operations for industrial hygiene. In *IEEE International Conference on Technologies for Practical Robot Applications (TePRA)*, pages 1–6, Woburn, MA, USA, 2015. IEEE.

[23] T. Valascho and R. Sosin. GVR-BOT thermal testing. In *Technical Report*. Defence Technical Information Centre, 2018.

[24] R. Dhaouadi and A. A. Hatab. Dynamic modelling of differential-drive mobile robots using lagrange and newton-euler methodologies: A unified framework. *Advances in Robotics and Automation*, 2(2):1–7, August 2013.

[25] E. Ivanjko, Toni Petrinic, and I. Petrovic. Modelling of mobile robot dynamics. 2010.

[26] Z. Sun, S. Balakrishnan, L. Su, A. Bhuyan, P. Wang, and C. Qiao. Who is in control? practical physical layer attack and defense for mmwave-based sensing in autonomous vehicles. *IEEE Transactions on Information Forensics and Security*, 16:3199–3214, April 2021.

[27] F. Farivar, M. S. Haghighi, Al. Jolfaei, and M. Alazab. Artificial intelligence for detection, estimation, and compensation of malicious attacks in nonlinear cyber-physical systems and industrial iot. *IEEE Transactions on Industrial Informatics*, 16(4):2716–2725, 2020.

[28] B. Dieber, R. White, S. TAurer, B. Breiling, G. Caiazza, H. Chrsitensen, and A. Cortesi. *Studies in Computational Intelligence Robot Operating System*, volume 4, chapter Penetration Testing ROS, pages 183–225. Springer, 2020.

[29] B. Dieber, R. White, S. Taurer, B. Breiling, G. Caiazza, H. Christensen, and A. Cortesi. *Penetration Testing ROS*, pages 183–225. Springer International Publishing, Cham, 2020.

[30] E. Nowak, F. Jurie, and B. Triggs. Sampling strategies for bag-of-features image classification. In A. Leonardis, H. Bischof, and A. Pinz, editors, *Computer Vision – ECCV 2006*, pages 490–503, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

Fendy Santoso received the master's degree in electrical and computer systems engineering from Monash University, Melbourne, VIC, Australia, in 2007, and the Ph.D. degree in electrical engineering from the University of New South Wales, Sydney, NSW, Australia, in 2012. Fendy currently holds a position as a Senior Lecturer in Engineering. He was a Research Fellow with the Defense and Systems Institute, UniSA STEM (Science, Technology, Engineering, and Mathematics), The University of South Australia, Adelaide, SA, Australia, and with the Autonomous Systems Laboratory, School of Engineering and Information Technology, UNSW Canberra, ACT, Australia, where he also currently holds a visiting research fellowship. His current research interests include control systems and artificial intelligence with applications in robotics and cyber security. Fendy has successfully made some remarkable contributions to intelligent robotics. He has advocated the concept of adaptive control systems in aerial robotics by innovating the concept of bio-mimetic self-learning by means of fuzzy systems. His research has led to safer and more efficient flights. Fendy was the recipient of the "Distinguished Early Career Travel Fellowship" (under the Vice Chancellor's Fellowship) Award from the University of Wollongong, NSW, Australia. He has been a reviewer for multiple high-impact journals, such as IEEE Trans on Fuzzy Systems, IEEE Trans on Neural Network and Learning Systems, IEEE Transactions on Systems, Man, and Cybernetics: Systems, IEEE Trans on Cybernetics, IEEE Trans on Control Systems Technology, IEEE Trans on Cognitive and Developmental Systems, and IEEE Access, in addition to multiple flagship international conferences.

Anthony Finn graduated from Cambridge University with a Ph.D. in 1989. He is a Professor of Autonomous Systems at the University of South Australia and was Director of the Defense and Systems Institute (DASI) from 2011 - 2019. He has published two books and around two hundred book chapters, journal articles, refereed conference papers, and research reports. Before joining DASI in 2010, Professor Finn worked at Australia's Defense Science and Technology Organization (DSTO) for almost twenty years, the last ten of which were spent leading a team of around forty scientists and engineers conducting research into the defense applications of autonomous and unmanned vehicles. Professor Finn worked as a research consultant for several large organizations in Europe.