

# Data-Driven Intrusion Detection in Vehicles: Integrating Unscented Kalman Filter (UKF) with Machine Learning

Shuhao Bian<sup>a</sup>, Milad Farsi<sup>b</sup>, Nasser L. Azad<sup>c</sup> and Chris Hobbs<sup>d</sup>

*Systems Design Engineering Dep., University of Waterloo, 200 University Ave W, Waterloo, Canada*

**Keywords:** UKF, Machine Learning (ML), Cyber-Physical System (CPS), Advanced Driver Assistance Systems (ADAS).

**Abstract:** In the realm of Cyber-Physical System (CPS), accurately identifying attacks without detailed knowledge of the system's parameters remains a major challenge. When it comes to Advanced Driver Assistance Systems (ADAS), identifying the parameters of vehicle dynamics could be impractical or prohibitively costly. To tackle this challenge, we propose a novel framework for attack detection in vehicles that effectively addresses the uncertainty in their dynamics. Our method integrates the widely used Unscented Kalman Filter (UKF), a well-known technique for nonlinear state estimation in dynamic systems, with machine learning algorithms. This combination eliminates the requirement for precise vehicle modeling in the detection process, enhancing the system's adaptability and accuracy. To validate the efficacy and practicality of our proposed framework, we conducted extensive comparative simulations by introducing Denial of Service (DoS) attacks on the vehicle systems' sensors and actuators.


## 1 INTRODUCTION


Cyber-Physical System (CPS) refers to the integration of computation with physical processes. Therefore, cyber attacks on these systems can cause severe consequences. The reliable operation of Advanced Driver Assistance Systems (ADAS) depends on the accurate functioning of various sensors and power management systems. If these elements are targeted by malicious attackers, passengers, pedestrians and drivers could be exposed to significant safety risks, potentially endangering their lives. This situation underscores the critical necessity for attack detection methods within autonomous driving systems. By identifying potential threats as they occur, the system can initiate appropriate protective actions to safeguard passengers, pedestrians and drivers.


Denial of Service (DoS) attack is one of the most well-known cyber attacks, and it has become more prevalent since 2004 (Mirkovic and Reiher, 2004). These attacks purposefully flood networks with too much traffic, overwhelming systems and compromising service availability. DoS attacks can cause sig-


nificant operational disruptions in Autonomous Vehicles (AVs) and Connected Autonomous Vehicles (CAVs), which rely heavily on continuous and secure communication channels for services such as navigation, real-time traffic updates, and vehicle-to-vehicle (V2V) communication. The absence of connectivity not only affects the safety elements essential to AV operations but also degrades the system's ability to make intelligent decisions on the road. Multiple recent studies can be found in the literature, providing insight into the different classes of attacks and defense mechanisms developed, such as (Naqvi et al., 2022; Marcillo et al., 2022; Al-Jarrah et al., 2019; Banafshehvaragh and Rahmani, 2023; Farsi. et al., 2023).

Concerning vehicle security, there exist various applications of machine learning techniques in the literature for detecting anomalies in different networks. One group of approaches considers the possibility of attacks on communications between vehicles themselves and between vehicles and roadside infrastructure. (Canh and HoangVan, 2023) seeks to build and evaluate a particular attack detection system that employs four specific discriminating features. A collected dataset is then utilized to train and evaluate several machine learning and statistical models, allowing for a comparative examination of their efficacy. The suggested strategy focuses on early detection, allowing for timely and effective countermeasures.

<sup>a</sup>  <https://orcid.org/0009-0000-7518-0972>

<sup>b</sup>  <https://orcid.org/0000-0002-5909-4377>

<sup>c</sup>  <https://orcid.org/0000-0003-1412-7961>

<sup>d</sup>  <https://orcid.org/0009-0007-6175-5449>

Another class of intrusion detection has focused on in-vehicle communications used for exchanging data between different control units of vehicles. (Berger et al., 2018) evaluates various machine learning methods, including deep learning, for in-vehicle intrusion detection systems. In a more recent technique, (Aldhyani and Alkahtani, 2022) implemented deep learning approaches like Convolutional Neural Network (CNN)s and CNN-Long Short-Term Memory (LSTM) hybrid models to detect attacks such as spoofing, flooding, and replay attacks on the Controller Area Network (CAN) bus. Other similar techniques can be found in (Pawar et al., 2022). A comprehensive survey of the techniques presented, in the literature, can be found in (Rajapaksha et al., 2023).

The majority of these approaches focus on network traffic and data package analysis to discover recurring patterns in normal operation during the training stage, and then use them to detect anomalies during the exploitation stage. Therefore, the dynamic aspects of different subsystems in the vehicles receive less attention. On the other hand, dynamic models are the foundation of vehicle design, allowing engineers to predict and optimize the performance of numerous vehicle systems under a variety of operating situations. Exploiting such models for intrusion detection allows us to identify anomalies by continuously comparing real-time data against the predicted normal behavior. Through this proactive strategy, intrusions can be detected early and promptly addressed to reduce potential damage. (Ju et al., 2022) provides a review of such techniques from a control perspective. In line with these approaches, in this study, we employ a rather system dynamics model-based technique as a defense mechanism. Moreover, since the parameters of the vehicle, such as weight, tire conditions, and engine characteristics, can change over time or under different conditions, we have found machine learning techniques particularly advantageous as they offer high adaptability.

The Unscented Kalman Filter (UKF) has been employed successfully in various fields of applications, such as power and automotive systems. In power systems, the UKF contributes to stability and operational integrity by accurately evaluating the state of electrical grids and identifying potential interruptions (Du et al., 2022; Rashed et al., 2022). The UKF has played an important role in improving cyber threat detection in-vehicle systems (Zhang et al., 2021), particularly AVs and CAVs. The UKF improves the system's robustness against DoS attacks by enabling real-time, precise anomaly detection, preventing possible threats from causing harm. In (Živković and Sarić, 2018), the authors employed the UKF to predict and update

state variables from a previously known state to detect False Data Injection (FDI) attacks. They compared the results to those obtained using a common weighted least squares-based state estimation technique. They observed that the state variables under attack significantly deviated between them, which can be used to detect the attack.

In this paper, we describe a novel approach for developing an attack detection system tailored to vehicles. Our proposed approach integrates the UKF with a learning-based module to obtain a resilient adaptive framework. This feature eliminates the requirement for detailed vehicle modeling in the attack detection process, simplifying implementation while retaining accuracy. The framework's effectiveness is strengthened by the use of a Cumulative Sum (CUSUM) algorithm with a sliding window for responsive anomaly detection, and by incorporating the learned dynamics to predict and compare real-time data against expected behavior. By leveraging the UKF's capabilities in handling non-linear dynamics, our proposed algorithm significantly improves the robustness and accuracy of intrusion detection in cyber-physical systems, particularly in ADAS. We conducted extensive simulations using CARLA, a simulation platform commonly used in autonomous driving research, to evaluate the efficacy and feasibility of our system. This made it possible for us thoroughly to test our structure in practical settings and make sure it can operate dependably in actual situations.

The rest of the paper is organized in the following order: In Section 2, we formulate the problem by assuming a DoS attack on the actuator. Section 3 presents the attack detection framework, introducing the main algorithm. In the following Section 4, we discuss the detailed simulation results.

## 2 PROBLEM FORMULATION

The system model is described by the following equations

$$\begin{aligned} x^{k+1} &= f(x^k, u^k), \\ y^k &= h(x^k), \end{aligned} \quad (1)$$

where  $x^k \in \mathbf{R}^n$ , represents the state space of the vehicle, and  $u^k \in \mathbf{R}^p$ , denotes the control input.  $y^k \in \mathbf{R}^m$  is the measurement vector. Moreover,  $f: \mathbf{R}^n \times \mathbf{R}^p \rightarrow \mathbf{R}^n$  represents the state transition function that defines the next state  $x^{k+1}$  based on the current state  $x^k$  and the control input  $u^k$ . The function  $h: \mathbf{R}^n \rightarrow \mathbf{R}^m$  denotes the measurement function that maps the current state  $x^k$  to the measurement vector  $y^k$ .  $k$  gives the time step.

Next, we will model the DoS attack on various elements of the control loop.

## 2.1 Attack Model

Considering that the attacks are implemented in the cyber layer, we construct the cyber attack in a discrete space, as shown below.

### 2.1.1 Actuator Attack

To model the effect of the attack, the dynamics in (1) are modified as

$$x^{k+1} = \begin{cases} f(x^k, a^k), & k \in \alpha, \\ f(x^k, u^k), & k \notin \alpha, \end{cases} \quad (2)$$

where we denote the set of time steps during which the attack is active using  $\alpha$ . Moreover, we assume that DoS can arbitrarily affect a subset of components of  $u^k$ . This can be further described by

$$a^k[i] = \begin{cases} 0, & i \in \Gamma_a, \\ u^k[i], & \text{otherwise}, \end{cases} \quad (3)$$

where  $\Gamma_a$  denotes the set of indices corresponding to the actuators that are affected by the attack, and  $i$  ranges over the set  $\Gamma_a \subseteq \{1, 2, \dots, p\}$ . Accordingly, a particular attack strategy can be represented by choosing nonempty sets  $\alpha$  and  $\Gamma_a$ , which specify the time of the attack and the indices of targeted actuators, respectively.

### 2.1.2 Sensor Attack

In this section, we define the sensor attack in a similar fashion. We modify the measurements relation given in equation (1) to reflect the DoS attack on some specific set of sensors. The equation

$$y^k = \begin{cases} y_a^k, & k \in \beta, \\ h(x^k), & k \notin \beta, \end{cases} \quad (4)$$

switches the measurements to  $y_a^k$  that may be manipulated by the anomaly, at different time steps specified by set  $\beta$ . Moreover, based on a particular attack pattern, a list of targeted sensors is given by  $\Gamma_s$ , with the attacked components being set to zero. This is shown below

$$y_a^k[i] = \begin{cases} 0, & i \in \Gamma_s, \\ y^k[i], & \text{otherwise}, \end{cases} \quad (5)$$

where  $\Gamma_s \subseteq \{1, 2, \dots, m\}$  denotes the set of indices corresponding to the components of  $y^k$  that are blocked by the attacker. Here,  $m$  refers to the dimension of the measurement vector.

## 2.2 Vehicle Model

In order to demonstrate the performance of the developed approach, we use the Kalman filter as a baseline approach for comparison in Section 4. It should be noted that we aim to obtain a sample-based technique, meaning that we treat the dynamics as a black box without using the analytical model. However, since the dynamic model are essential for running the Kalman Filter, we concisely present the equations of the vehicle in what follows.

According to (Takahama and Akasaka, 2018), a vehicle dynamics model can be obtained as below. The longitudinal dynamics of the vehicle is given by

$$M_{vehicle} \dot{v}_h = M_{vehicle} a_f - r_{travel}, \quad (6)$$

where  $M_{vehicle}$  is the mass of the vehicle,  $a_f$  is the traction force converted to acceleration and  $r_{travel}$  is the travel resistance. The model of the  $r_{travel}$  can be described as

$$r_{travel} = r_{air}(v_h^2) + r_{roll}(v_h) + r_{accel}(\dot{v}_h) + r_{grad}(\theta), \quad (7)$$

where  $r_{air} = \frac{1}{2} \rho C_d A v_h^2$  is the air drag,  $\rho$  is the air density,  $C_d$  is the drag coefficient,  $A$  is the frontal area of the vehicle, and  $v_h$  is the vehicle speed. The rolling resistance is given by  $r_{roll} = C_r M_{vehicle} g v_h$ , where  $C_r$  and  $g$  denote the rolling resistance coefficient and the acceleration due to gravity, respectively.

Then, the acceleration resistance is  $r_{accel} = M_{vehicle} \dot{v}_h$ , where  $\dot{v}_h$  is the acceleration of the vehicle. Finally,  $r_{grad} = M_{vehicle} g \sin(\theta)$ , where  $\theta$  is the slope angle of the road.

## 3 ATTACK DETECTION FRAMEWORK

In this section, we present the required components to construct the attack detection framework.

### 3.1 Unscented Kalman Filter

In the UKF, it is not necessary to know the detailed model. The UKF uses sigma points to sample the input and obtain the corresponding output. It is a method similar to the Monte Carlo approach but requires only a small number of sigma points. Next, we will provide a detailed introduction to the UKF.

#### 3.1.1 Unscented Transformation (UT)

As described by (Wan and Van Der Merwe, 2000), the UT is a technique developed for the generation of

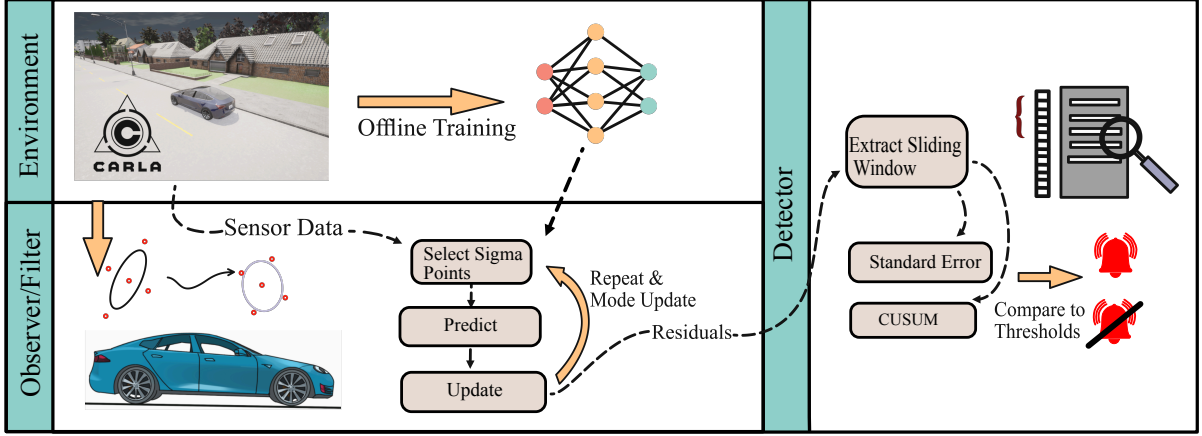


Figure 1: The flow chart of the algorithm.

sigma points that are capable of undergoing nonlinear transformations expressed as  $f(x^k, u^k)$ . This approach is particularly valuable when dealing with a multitude of random vectors, each residing in an  $n$ -dimensional space ( $x^k \in \mathbf{R}^n$ ), characterized by a mean  $\bar{x}^k$  and a covariance matrix  $P^k$ .



Figure 2: CARLA Environment.

### 3.1.2 Select Sigma Points

The sigma points  $\chi_i^k$  are chosen such that they capture the true mean and covariance of the random variable  $x^k$ , enabling an accurate propagation through the nonlinear function  $f(x^k, u^k)$ . This selection ensures that the mean and covariance of the sigma points match  $\bar{x}^k$  and  $P^k$ , respectively, providing an effective mechanism for estimating the statistical properties of  $x^{k+1}$ . The sigma points are selected according to

$$\begin{cases} \chi_0^k &= \bar{x}^k, \\ \chi_i^k &= \bar{x}^k + (\sqrt{(n+\lambda)P^k})_i \quad i = 1, \dots, n, \\ \chi_i^k &= \bar{x}^k - (\sqrt{(n+\lambda)P^k})_i \quad i = n+1, \dots, 2n, \\ W_0^{(m)} &= \frac{\lambda}{n+\lambda}, \\ W_0^{(c)} &= \frac{\lambda}{n+\lambda} + (1 - \phi^2 + \beta), \\ W_i^{(m)} &= W_i^{(c)} = \frac{1}{2(n+\lambda)} \quad i = 1, \dots, 2n. \end{cases} \quad (8)$$

According to (Wang et al., 2023),  $\lambda$  is a key parameter calculated using  $\lambda = \phi^2(n + \kappa) - n$ ,  $\phi$  deter-

mines the dispersion degree of  $\sigma$  points, and  $\kappa$  is typically  $3 - n$ .  $\beta$  is for integrating prior knowledge about  $x$ 's distribution.

### 3.1.3 Predict

The predicted state and covariance are

$$\begin{aligned} \bar{x}^{k+1|k} &= \sum_{i=0}^{2n} W_i^{(m)} \chi_i^{k+1|k}, \\ P^{k+1|k} &= \sum_{i=0}^{2n} W_i^{(c)} (\chi_i^{k+1|k} - \bar{x}^{k+1|k}) (\chi_i^{k+1|k} - \bar{x}^{k+1|k})^T, \\ \bar{y}^{k+1|k} &= \sum_{i=0}^{2n} W_i^{(m)} \mathcal{Y}_i^{k+1|k}, \\ P_{yy}^{k+1} &= \sum_{i=0}^{2n} W_i^{(c)} (\mathcal{Y}_i^{k+1|k} - \bar{y}^{k+1|k}) (\mathcal{Y}_i^{k+1|k} - \bar{y}^{k+1|k})^T, \\ P_{xy}^{k+1} &= \sum_{i=0}^{2n} W_i^{(c)} (\chi_i^{k+1|k} - \bar{x}^{k+1|k}) (\mathcal{Y}_i^{k+1|k} - \bar{y}^{k+1|k})^T, \end{aligned} \quad (9)$$

where  $\chi_i^{k+1|k} = f(\chi_i^k, u^k)$ ,  $\mathcal{Y}_i^{k+1|k} = h(\chi_i^k, u^k)$ , for  $i = 0, \dots, 2n$ .

### 3.1.4 Update

The update step can be defined as

$$\begin{aligned} K^{k+1} &= P_{xy}^{k+1} (P_{yy}^{k+1})^{-1}, \\ r^{k+1} &= y^{k+1} - \bar{y}^{k+1|k}, \\ x^{k+1} &= \bar{x}^{k+1|k} + K^{k+1} r^{k+1}, \\ P^{k+1} &= P^{k+1|k} - K^{k+1} P_{xy}^{k+1T}, \end{aligned} \quad (10)$$

where  $K$  is the Kalman gain,  $r$  is the residual. These values are updated when sensors provide new measurements.



**Data:** Sensor signal, Input signal  
**Result:** Attack State:  $f_{ad}$  (0 for no attack, 1 for attack detected)

*Initialisation:* ;  
 Initialize the cumulative sum,  $s_1, s_2 \leftarrow 0$  ;  
 Initialize the attack detection flag,  $f_{ad} \leftarrow 0$  ;  
 Initialize the threshold value,  $threshold$  ;  
 Define the window size  $N$  ;  
 Initialize an empty list  $Q$ , to store the last  $N$  residuals ;  
*Detection Loop:* ;  
**while** each new sensor signal input  $y$  **do**  
     Perform the selecting sigma points step;  
     Perform the prediction step of UKF to estimate the next state ;  
     Perform the update step of UKF with signal  $y$  to obtain residual  $r$  ;  
     Append  $r$  to  $Q$  ;  
     Update the UKF model // using equation (11) ;  
     **if** Size of  $Q > N$  **then**  
         | Remove the oldest residual from  $Q$  ;  
     **end**  
     Update cumulative sum using residuals within  $Q$ :  $s_1 \leftarrow \sum Q$  // using equation (12) ;  
     Update standard error using residuals within  $Q$ :  $s_2 \leftarrow std(Q)$  // using equation (13) ;  
     **if**  $abs(s_1) > threshold_1$  **and**  $s_2 > threshold_2$  **then**  
         |  $f_{ad} \leftarrow 1$  // attack detected immediately upon detection ;  
     **else**  
         |  $f_{ad} \leftarrow 0$  ;  
         | Continue monitoring ;  
     **end**  
**end**

Algorithm 1: CUSUM Algorithm with Sliding Window for Attack Detection.

### 3.2 Machine Learning (ML)

Obtaining the parameters of a vehicle might often be infeasible or economically prohibitive. In the update phase of the UKF, we employ ML techniques to predict the vehicle's subsequent state.

For this purpose, we selected the multi-layer perceptron (MLP) network. This neural network receives control commands ( $u$ ) — encompassing the throttle-brake and steering angle — as well as the vehicle's current state, which includes velocity, angular velocity, and acceleration, as its input. The network is designed to output the vehicle's acceleration for the next

time step. Designed to predict the vehicle's acceleration at the next time step, it was necessary to account for the delay between the input commands and the resultant state changes. To address this, according to (Xu et al., 2019), we selected a model output that reflects the vehicle's acceleration 50 milliseconds after the input, ensuring the training data adequately captures the dynamics of the system.

The MLP is structured with three distinct layers: an input layer comprising 5 units, a hidden layer containing 20 units, and a single-unit output layer. The activation function utilized within the hidden layer is the Rectified Linear Unit (ReLU).

With this predicted acceleration, we can determine the vehicle's upcoming velocity using Newton's second law, thereby integrating ML predictions seamlessly into the UKF's update mechanism for enhanced estimation accuracy.

### 3.3 Model Update Methods

In order to make the detector adaptive, we deployed an algorithm that can make the detector update automatically. In the proposed detector, we updated the model of the vehicle when it got new sensor signals. To make sure the model is not updated when the attacker implements the DoS attack on the car model and is updated when the car is running normally, we update the learning rate according to

$$l = 1 - \frac{1}{1 + e^{-S_{rate} \|r\|_2}}. \quad (11)$$

Here,  $l$  denotes the learning rate,  $r$  represents the residual of the car,  $S_{rate}$  is the scale of the residual, and  $\|\cdot\|_2$  denotes the  $L_2$  norm of a vector.

### 3.4 Attack Detection Method

As a widely used attack detector in many literature (Liu et al., 2019), CUSUM is selected as the detector. In (Van Eykeren et al., 2012), they also calculated a moving window average for the residual. In practice, there is often a gap between theoretical models and engineering applications. Therefore, we introduce  $W_{r1}$  and  $W_{r2}$  to fine-tune the target parameters. Additionally, we used two tests to detect whether the target is under attack.

Test 1 is shown by

$$s_1^k = \sum_{i=k-N+1}^k W_{r1} r^i, \quad (12)$$

where  $r$  is the residual updated by the UKF,  $W_{r1}^T \in \mathbf{R}^m$ ,  $N$  is the length of the sliding window.

Test 2 is described as

$$s_2^k = \sum_{i=k-N+1}^k (r^i - \bar{r}) W_{r2} (r^i - \bar{r})^T, \quad (13)$$

where  $\bar{r}$  is the average of the sliding window for the residuals,  $W_{r2} \in \mathbf{R}^{m \times m}$  is the weight of the residual.

Then  $s_1$  and  $s_2$  will be compared with two thresholds,  $t_1$  and  $t_2$ , respectively. Then the alarm is triggered according to

$$A = \begin{cases} S_p, & s_1 > t_1 \text{ and } s_2 > t_2, \\ S_n, & s_1 \leq t_1 \text{ or } s_2 \leq t_2. \end{cases} \quad (14)$$

As illustrated in Figure 1, the trained model derived from historical data is subsequently integrated into the UKF. The mechanism of the detector is detailed in Algorithm 1.

## 4 SIMULATION RESULTS

To demonstrate the efficacy of the proposed approach under DoS attacks, we chose CARLA (Dosovitskiy et al., 2017) as our simulator, which can simulate real-world dynamics and generate sensor signals in real-time. This allows for accurate and timely responses in the simulation environment. Moreover, to highlight the superiority of the proposed method, we compare it with the traditional Kalman filter.

All neural network training was conducted in Python on the Ubuntu operating system. The hardware configuration used included an AMD Ryzen 9 processor with 16 cores, clocked at 3.40 GHz, and 64GB of RAM. The Kalman filter and UKF were implemented using the FilterPy library (Labbe, 2024), while the neural network was developed with PyTorch (Paszke et al., 2019).

In the following subsection, we detail the data preprocessing methods, the implementation of attack scenarios, the simulation parameters and the resulting simulation outcomes.

### 4.1 Data Preprocessing

Both manual driving data and autonomous driving data generate command signals (throttle, brake, and steering angle), which can be used as training data. To obtain representative data, we used manual control to generate data in the CARLA environment, as shown in Figure 2.

In order to achieve better training results, we need to ensure that the data are valid by removing all outliers.

The brake and throttle inputs are combined and normalized into a unified control signal ranging from 0 to 1 according to

$$u = \frac{T - B + 1}{2}, \quad (15)$$

where  $u$  is the unified control signal,  $T \in [0, 1]$  is the throttle input,  $B \in [0, 1]$  is the brake input,  $T_{max}$  is the maximum throttle value and  $B_{min}$  is the minimum brake value. When  $u = 0$ , it represents full braking, and when  $u = 1$ , it represents full throttle.

### 4.2 Attack Implementation

We implemented the attack at 20 seconds in the simulation. To validate the algorithm, we chose a DoS attack signal in the form of a Pulse Width Modulation (PWM) structure. When the signal is 1, it indicates that the monitor is blocked.

### 4.3 Parameters

To have a comparison we chose a typical Kalman filter. The parameters of equation (7) is shown as table 1.

Table 1: Parameters of Kalman filter.

Name	Value
$r_{air}$	68.9 N
$r_{roll}$	271.6 N
$r_{grad}$	0 N

If there is no slope,  $r_{grad} = 0$ . Therefore, the total travel resistance  $r_{travel}$  is given by  $r_{travel} = r_{air} + r_{roll} + r_{grad} = 340.5N$ . In order to make the detector more accurate, we have  $W_{r1} = [1 \ 0.01 \ 0]^T$ ,

$W_{r2} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0.01 & 0 \\ 0 & 0 & 0 \end{bmatrix}$  and we focus on the acceleration residual to detect the attack.

The learning parameters are detailed in Table 2. The optimizer used is Adam (Kingma and Ba, 2014).

Table 2: Parameters of Machine Learning.

Name	Value
Learning rate	0.001
Train size/Total	0.8
Epochs	1000
MLP batch size	64
MLP epochs	1000

To improve the clarity of the results, we simplified the model, the value of  $t_2$  is based on  $t_1$ , maintaining a fixed proportional relationship as shown by

$$t_2 = \gamma t_1, \quad (16)$$

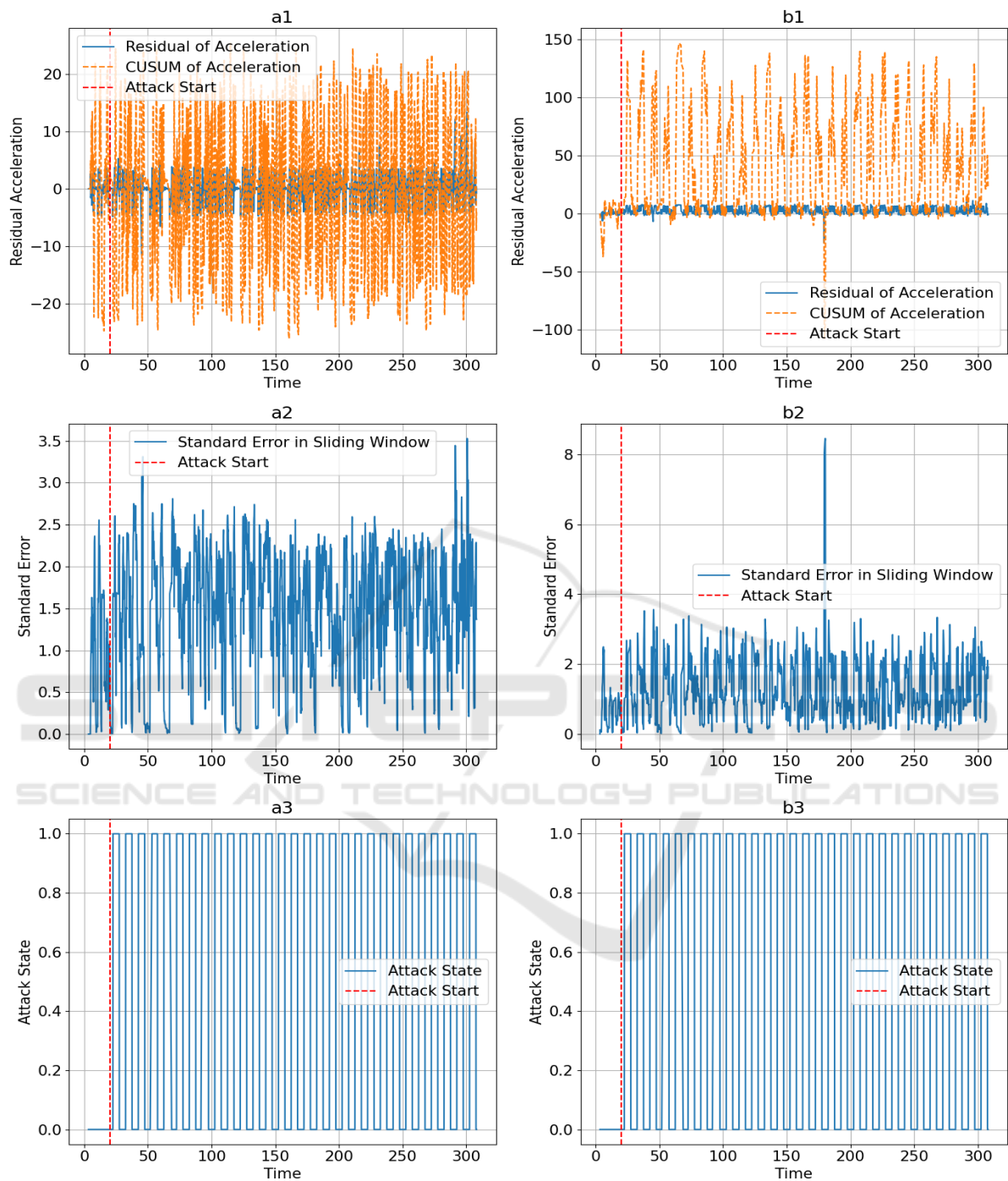


Figure 3: The results of the attack detector are as follows: for the Kalman filter, *a1* represents the residual and CUSUM of residuals in the sliding window, *a2* denotes the standard error of residuals in the sliding window, and *a3* indicates the attack state; for the proposed method, *b1* signifies the residual and CUSUM of residuals in the sliding window, *b2* refers to the standard error of residuals in the sliding window, and *b3* denotes the attack state of the UKF.

where  $\gamma$  is a tuning parameter. In this simulation,  $\gamma$  is set to 0.04.

## 4.4 Results

In this study, the performance of the Kalman filter and UKF for detecting attacks in autonomous vehicles was evaluated. The experiments were conducted under two scenarios: actuator attack and sensor attack.

### 4.4.1 Actuator Attack

To validate the detector, we use four metrics to measure the performance: false positive alarm rate, true positive alarm rate, false negative alarm rate and true negative alarm rate.

As shown in Figure 4, despite the threshold changes from 10 to 25, the true positive alarm rate and true negative alarm rate of the proposed method are significantly higher than those of the Kalman filter when using the same threshold.

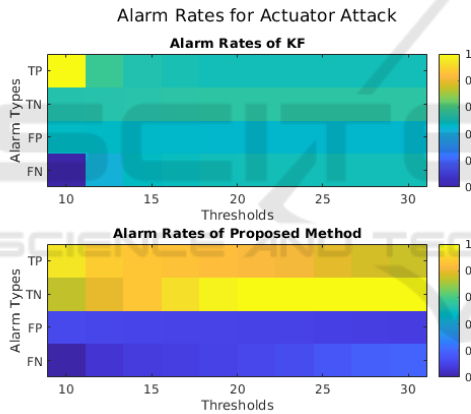


Figure 4: The metrics of different detectors are defined as follows: the true positive rate (TP), the true negative rate (TN), the false positive rate (FP), and the false negative rate (FN).

To achieve a more precise quantification of the research data, we utilized the F1-score (Bishop and Nasrabadi, 2006). As shown in Figure 5, the F1-score of the Kalman filter shows a decreasing trend, while the F1-score of the UKF first increases and then decreases. Based on this pattern, we selected the appropriate thresholds for the Kalman filter and the proposed method. The thresholds are 10 for the Kalman filter and 13.33 for the proposed method.

Table 3 presents the optimal thresholds and corresponding values for the Kalman filter and the proposed method. In this table, all metrics of the proposed method significantly outperform those of the Kalman filter.

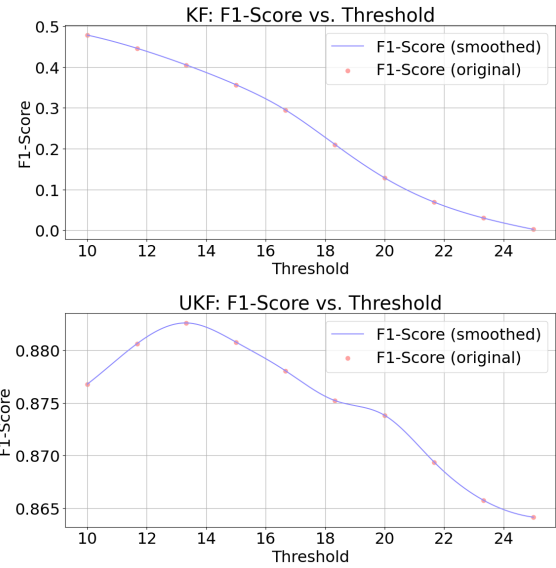


Figure 5: F1 Score of Kalman Filter and the Proposed Method.

Table 3: Optimal Thresholds and Confusion Matrix Values for Kalman Filter and the Proposed Method.

Metric	Kalman filter	Ours
Optimal Threshold	10.0	13.33
F1-Score	0.4784	0.8826
True Positives (TP)	42.88%	93.58%
False Positives (FP)	36.41%	18.48%
True Negatives (TN)	63.59%	81.52%
False Negatives (FN)	57.12%	6.42%

### 4.4.2 Sensor Attack

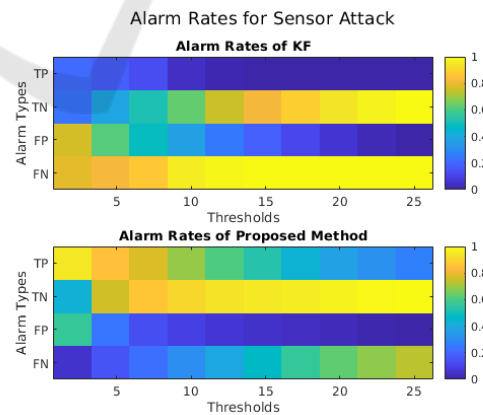


Figure 6: The metrics of different detectors are defined as follows: the true positive rate (TP), the true negative rate (TN), the false positive rate (FP), and the false negative rate (FN).

We obtained similar results for Attack Detection (AD) on the sensor, as shown in Figure 6. Both the true positive and true negative rates of the proposed UKF



method are significantly higher than those of the Kalman filter when choosing the best F1-Score.

## 5 CONCLUSION

In conclusion, we proposed a novel framework for attack detection in vehicles with unknown systems. We exploited a learning-based model to predict and compare observations against expected behavior. Therefore, compared to the Kalman filter, the proposed approach based on the UKF is capable of detecting DoS attacks from the sensor and actuator without prior knowledge of the system parameters. Accordingly, by exploiting UKF's capabilities in handling nonlinearity, our proposed algorithm demonstrated a significant advantage over the traditional Kalman filter for detecting DoS attacks on sensors and actuators. In detail, through extensive simulations of our proposed algorithm, we observed that our method outperforms the Kalman filter by demonstrating substantial results in both true positive alarm rate and true negative alarm rate. Enhancing the filtering design for vehicle intrusion detection can be the primary focus of future research. By precisely simulating intricate dynamics, investigating robust particle filters may improve anomaly identification even further and strengthen vehicle security. Moreover, the presented framework can be extended to other types of attacks such as FDI.

## ACKNOWLEDGEMENT

We gratefully acknowledge the financial support provided by BlackBerry and the Natural Sciences and Engineering Research Council of Canada (NSERC). Furthermore, we acknowledge BlackBerry QNX's assistance and cooperation with this study. Their assistance has been crucial to the accomplishment of this task.

## REFERENCES

- Al-Jarrah, O. Y., Maple, C., Dianati, M., Oxtoby, D., and Mouzakitis, A. (2019). Intrusion detection systems for intra-vehicle networks: A review. *Ieee Access*, 7:21266–21289.
- Aldhyani, T. H. and Alkahtani, H. (2022). Attacks to autonomous vehicles: A deep learning algorithm for cybersecurity. *Sensors*, 22(1):360.
- Banafshehvaragh, S. T. and Rahmani, A. M. (2023). Intrusion, anomaly, and attack detection in smart vehicles. *Microprocessors and Microsystems*, 96:104726.
- Berger, I., Rieke, R., Kolomeets, M., Chechulin, A., and Kotenko, I. (2018). Comparative study of machine learning methods for in-vehicle intrusion detection. In *International Workshop on Security and Privacy Requirements Engineering*, pages 85–101. Springer.
- Bishop, C. M. and Nasrabadi, N. M. (2006). *Pattern recognition and machine learning*, volume 4. Springer.
- Canh, T. N. and HoangVan, X. (2023). Machine learning-based malicious vehicle detection for security threats and attacks in vehicle ad-hoc network (vanet) communications. In *2023 RIVF International Conference on Computing and Communication Technologies (RIVF)*, pages 206–211. IEEE.
- Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., and Koltun, V. (2017). Carla: An open urban driving simulator.
- Du, D., Zhu, M., Li, X., Fei, M., Bu, S., Wu, L., and Li, K. (2022). A review on cybersecurity analysis, attack detection, and attack defense methods in cyber-physical power systems. *Journal of Modern Power Systems and Clean Energy*, 11(3):727–743.
- Farsi., M., Bian., S., Azad., N. L., Shi., X., and Walenstein., A. (2023). An efficient resilient mpc scheme via constraint tightening against cyberattacks: Application to vehicle cruise control. In *Proceedings of the 20th International Conference on Informatics in Control, Automation and Robotics - Volume 1: ICINCO*, pages 674–682. INSTICC, SciTePress.
- Ju, Z., Zhang, H., Li, X., Chen, X., Han, J., and Yang, M. (2022). A survey on attack detection and resilience for connected and automated vehicles: From vehicle dynamics and control perspective. *IEEE Transactions on Intelligent Vehicles*, 7(4):815–837.
- Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Labbe, R. (2024). filterpy: Kalman filters and other optimal and non-optimal estimation filters in python. <https://github.com/rlabbe/filterpy>. Accessed: 20 June 2024.
- Liu, Q., Mo, Y., Mo, X., Lv, C., Mihankhah, E., and Wang, D. (2019). Secure pose estimation for autonomous vehicles under cyber attacks. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 1583–1588. IEEE.
- Marcillo, P., Tamayo-Urgilés, D., Valdivieso Caraguay, Á. L., and Hernández-Álvarez, M. (2022). Security in v2i communications: a systematic literature review. *Sensors*, 22(23):9123.
- Mirkovic, J. and Reiher, P. (2004). A taxonomy of ddos attack and ddos defense mechanisms. *ACM SIGCOMM Computer Communication Review*, 34(2):39–53.
- Naqvi, I., Chaudhary, A., and Kumar, A. (2022). A systematic review of the intrusion detection techniques in vanets. *TEM Journal*, 11(2):900.
- Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. (2019). Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.
- Pawar, Y. S., Honnavalli, P., and Eswaran, S. (2022). Cyber attack detection on self-driving cars using machine

- learning techniques. In *2022 IEEE 3rd Global Conference for Advancement in Technology (GCAT)*, pages 1–5. IEEE.
- Rajapaksha, S., Kalutarage, H., Al-Kadri, M. O., Petrovski, A., Madzudzo, G., and Cheah, M. (2023). Ai-based intrusion detection systems for in-vehicle networks: A survey. *ACM Computing Surveys*, 55(11):1–40.
- Rashed, M., Kamruzzaman, J., Gondal, I., and Islam, S. (2022). False data detection in a clustered smart grid using unscented kalman filter. *IEEE Access*, 10:78548–78556.
- Takahama, T. and Akasaka, D. (2018). Model predictive control approach to design practical adaptive cruise control for traffic jam. *International Journal of Automotive Engineering*, 9(3):99–104.
- Van Eykeren, L., Chu, Q., and Mulder, J. (2012). Sensor fault detection and isolation using adaptive extended kalman filter. *IFAC Proceedings Volumes*, 45(20):1155–1160.
- Wan, E. A. and Van Der Merwe, R. (2000). The unscented kalman filter for nonlinear estimation. In *Proceedings of the IEEE 2000 adaptive systems for signal processing, communications, and control symposium (Cat. No. 00EX373)*, pages 153–158. Ieee.
- Wang, C., Chen, L., Li, L., Yan, Y.-H., Sun, J., Yu, C., Deng, X., Liang, C.-P., Zhang, X.-L., and Peng, W.-M. (2023). Combining unscented kalman filter and wavelet neural network for anti-slug. *Petroleum Science*, 20(6):3752–3765.
- Xu, J., Luo, Q., Xu, K., Xiao, X., Yu, S., Hu, J., Miao, J., and Wang, J. (2019). An automated learning-based procedure for large-scale vehicle dynamics modeling on baidu apollo platform. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5049–5056. IEEE.
- Zhang, D., Lv, C., Yang, T., and Hang, P. (2021). Cyber-attack detection for autonomous driving using vehicle dynamic state estimation. *Automotive Innovation*, 4:262–273.
- Živković, N. and Sarić, A. T. (2018). Detection of false data injection attacks using unscented kalman filter. *Journal of Modern Power Systems and Clean Energy*, 6(5):847–859.