Received: 17 June 2018 | Revised: 14 January 2018 | Accepted: 13 February 2019

DOI: 10.4218/etrij.2018-0261

ORIGINAL ARTICLE

WILEY ETRI Journal

# Temporal and spatial outlier detection in wireless sensor networks

**Hoc Thai Nguyen**[1] | **Nguyen Huu Thai**[2]

[1]Department of Automation, Vietnam National University of Agriculture, Hanoi, Vietnam

[2]Faculty of Electrical Engineering, Vinh University of Technology Education, Vinh, Vietnam

**Correspondence**
Hoc Thai Nguyen, Department of Automation, Vietnam National University of Agriculture, Hanoi, Vietnam.
Email: nguyenthaihoc@vnua.edu.vn

Outlier detection techniques play an important role in enhancing the reliability of data communication in wireless sensor networks (WSNs). Considering the importance of outlier detection in WSNs, many outlier detection techniques have been proposed. Unfortunately, most of these techniques still have some potential limitations, that is, (a) high rate of false positives, (b) high time complexity, and (c) failure to detect outliers online. Moreover, these approaches mainly focus on either temporal outliers or spatial outliers. Therefore, this paper aims to introduce novel algorithms that successfully detect both temporal outliers and spatial outliers. Our contributions are twofold: (i) modifying the Hampel Identifier (HI) algorithm to achieve high accuracy identification rate in temporal outlier detection, (ii) combining the Gaussian process (GP) model and graph-based outlier detection technique to improve the performance of the algorithm in spatial outlier detection. The results demonstrate that our techniques outperform the state-of-the-art methods in terms of accuracy and work well with various data types.

**KEYWORDS**
Gaussian process, Hampel Identifier, outlier detection, spatial outlier, temporal outlier

## 1 | INTRODUCTION

Wireless sensor networks have become a promising technology that can be used in a variety of applications related to data acquisition and data monitoring. They provide a vast amount of online information in the deployment field. However, due to constraints on the signal processing and communication capabilities [1–5] of WSNs, some unusual data (called outliers) may result from sensor malfunction, process disturbances, human-related errors, and/or a sudden change in the state of the environment [6–10]. The outliers might seriously affect the accuracy of data analysis, which leads to model misspecification. Therefore, outlier detection is one of the most important preprocessing steps in any data analytical application [11–14]. This has stimulated many researchers in both temporal and spatial outlier detection [15–19].

In outlier detection, the Hampel Identifier (HI) is the most widely used and efficient outlier identifier [15]. In the HI algorithm, the median and the median absolute deviation (MAD) of a moving window with a size of $(2L + 1)$ are calculated, where $L$ is the number of observations before and after the current observation. The thresholds for outlierness evaluation are calculated according to where the generated parameter is in the range from 0 to 5. Any observation falling outside the range of such thresholds is identified as an outlier and is replaced by the median value of the data window. However, the HI algorithm still reveals its limitations with a highly autocorrelated data process. More precisely, it may fail to detect outliers due to the strong autocorrelation [16].

Additionally, in the HI algorithm, the standard deviation estimates are replaced by the MAD from the median.

However, this MAD scale estimator can behave badly with coarsely quantized data [18].

To the best of our knowledge, most of the existing outlier detection methods are still mainly designed for detecting outliers in offline data, and there is little research on identification of outliers in streaming data. This may seriously affect the accuracy of real-time decision-making. Moreover, by detecting outliers in streaming data successfully, the energy consumption, memory usage as well as the running time complexity can be significantly reduced.

In this paper, we exploit the sensor reading over a period of time for temporal outlier detection and the readings from 2-hop neighbor nodes of the candidate node for spatial outlier detection. The contributions of our approaches are twofold: (a) modifying the HI algorithm to achieve high accuracy identification rate in temporal outlier detection, (b) combining the GP model and the graph-based outlier detection technique to improve the performance of the algorithm in spatial outlier detection. It should be noted that our proposed methods are computationally simple. Therefore, they are suitable for applications with energy and computational constraints. Our algorithms achieve better performances in terms of accuracy identification rate compared to the HI algorithm and some other algorithms.

The remainder of the paper is organized as follows. In Section 2, we classify the outlier detection methods and briefly summarize some performance indices. In Section 3, we define the problem statements and provide a background for our network model. Section 4 describes our algorithms. The performance metrics of our algorithms are analyzed in Section 5. Finally, Section 6 concludes our work and suggests some potential directions for future research.

## 2 | RELATED WORKS

Outlier detection methods have become one of the primary concerns in WSNs. There have been many efforts on improving the efficiency of such methods, which can be categorized into: (a) parametric (statistical based) methods and (b) nonparametric methods [20]. The former [21–23] assumes a stochastic distribution for observations. It labels the observations as outliers if there is a significant difference between the observations and the model assumptions. Unfortunately, it may fail to identify outliers in high-dimensional datasets. Concretely, the complexity and inaccuracy of estimation increase gradually with the multidimensional distributions of data points [24]. The latter is known as a graph-based outlier detection method, which consists of the density-based method, the distance-based method, and the clustering-based method [24–28]. It outperforms the former when using the distances among points to detect outliers. It computes either the dimensional distance between points [26,29] or the densities of local neighborhoods [25] to detect outliers in a dataset. However, these methods may be bounded due to deterioration of the high-dimensional data [30]. Therefore, in Ref. [30], the authors proposed angle-based outlier detection (ABOD) for high-dimensional data. They further proved that the angles are more stable than the distances in high-dimensional data. The main idea of ABOD is a comparison of the angles between pairs of distance vectors to detect outliers in datasets. ABOD has some advantages and disadvantages. The advantages of ABOD are that it achieves high performance, especially in the case of high-dimensional data, and that it does not require any specific parameter as in the case of typical methods. Nevertheless, it still has some limitations for using the angle-based outlier factor (ABOF) to detect outliers. The three disadvantages of the ABOD algorithm are as follows: (a) owing to the long time complexity $\mathcal{O}(n^3)$, where $n$ is the size of a dataset, the ABOD algorithm may not suitable for outlier detection in large datasets [31]; (b) it achieves a low accuracy identification rate with boundary points [32]; and (c) it does not rely on any parameter, which influences the algorithm performance [33]. To overcome the aforementioned drawbacks of outlier detection methods, we propose new outlier detection methods that can detect both temporal outliers and spatial outliers. Similar to the research [34,35], the following performance indices have been used for evaluating the effectiveness of outlier detection methods:

1. There are four possible outcomes when detecting outliers [35].
   a. Type 1 (Normal points-NP): The inliers that are identified as inliers (by the detection method).
   b. Type 2 (False detection points-FP): The inliers that are identified as outliers.
   c. Type 3 (Miss detection points-MP): The outliers that are identified as inliers.
   d. Type 4 (Correct detection points-CP): The outliers that are identified as outliers.
2. Let TDR denote the true-positive rate, which measures the fraction of outliers that are correctly identified. Obviously,

$$\mathrm{TPR} = \frac{\mathrm{CP}}{\mathrm{CP} + \mathrm{MP}}. \tag{1}$$

3. Let FPR denote the false-positive rate, which measures the fraction of inliers that are misidentified as outliers. Obviously,

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{NP}}. \tag{2}$$

4. Let IR denotes the identification rate as follows.

$$\mathrm{IR} = \frac{\mathrm{CP}}{\max\{\mathrm{FP}+\mathrm{CP}, \mathrm{MP}+\mathrm{CP}\}}. \quad (3)$$

## 3 | PROBLEM STATEMENT AND BACKGROUND

Before presenting the network model, we first specify the basic assumptions of our WSN model.

### 3.1 | Basic assumptions

1. We consider a sensor network with $N$ homogeneous sensor nodes, which are distributed uniformly in the area of interest.
2. The sensor nodes are energy constrained and are stationary after deployment.
3. At the current sensing round $k$th, each sensor node collects (a) sensing data from its 2-hop neighbor nodes and (b) its $k$ previous adjacent sensing values. The received data are used for outlier detection at each node before being sent to the base station.
4. The sensor nodes know in advance the geographic position of the network nodes that are used to detect the event areas.
5. The storage capacity of each sensor node is large enough to store $k$, its adjacent sensed values (from time instant $t_k$ in the current sensing round $k$th to the time instant $t_1$ in the previous sensing round), and data values at time instant $t_k$ from its 2-hop neighbor nodes.
6. Two sensor nodes are considered as neighbors if they are located inside their radio transmission areas.

### 3.2 | Network model

Let $y_i(t)$, $i = 1, \ldots, N$ denote the measurement of sensor node $S_i$ at time $t$. In [36], $y_i$ was assumed the 0–1 binary variable for detecting outliers. This attempt may face some drawbacks [37], which results in a high rate of false alarm during outlier detection. Therefore, in this paper, $y_i$ represents the actual measurement of the parameter of interest. Each sensor node stores $k$, its previous sensing data (from time instant $t_1$ to time instant $t_k$), and the sensed data from 2-hop neighbor nodes at time instant $t_k$. By implementing our proposed algorithms, there are three possible outcomes of the reading $y_i(t_k)$, as follows:

1. Normal reading (if the difference between $y_i(t_k)$ and $y_i(t_j)$, $(k - L \le j < k)$ is not significant).
2. Temporal outlier (if the difference between $y_i(t_k)$ and $y_i(t_j)$, $(k - L \le j < k)$ is significant).
3. Spatial outlier (if the difference between $y_i(t_k)$ and 2-hop neighbor nodes values is significant).

All types of outlier nodes are defined below.

**Definition 1** A node is called an outlying node at time instant $(t_k)$ if its measurement value $y_i(t_k)$ is a temporal outlier and its 2-hop neighbor nodes values are normal readings.

**Definition 2** An event area contains $m$ nodes whose measurements are temporal outliers and satisfy $|y_i(t_k) - \mu_i| \le \theta, i = 1, \ldots, m$, where $\theta$ is a predefined threshold, and $\mu_i$ is the predicted value.

**Definition 3** An outlier area contains $m$ nodes whose measurements are spatial outliers and satisfy $|y_i(t_k) - \mu_i| > \theta$.

It is natural to ask how to obtain the optimal value of $\theta$. To answer this question, in the following sections, we will show how to obtain the best possible performance of our outlier detection algorithms with the optimal value of $\theta$.

In order to understand fully the appearance of outliers in WSNs, Figure 1 depicts an example of some outlier types. Figure 1A illustrates the network structure with $N = 20$ sensor nodes. These nodes are randomly deployed in the area of interest. In Figure 1B, the light blue circles represent the normal nodes, the green circles represent the event nodes, and the black circles indicate the outlying sensor nodes (faulty nodes). These figures raise the problems of identifying outliers in real time and distinguishing between outlier areas and event areas. At time instant $t_{k-2}$ (Figure 1A), the sensor data are in the normal range. However, in Figure 1B, there are some changes in the network. More precisely, the measurements of sensor nodes $S_2$, $S_5$, and $S_7$ indicate that an event occurred in this event area. Additionally, the measurements of sensor nodes $S_4$, $S_6$, and $S_{18}$ indicate that they are in the outlier area. The working nodes and the outlying nodes in the network may change over time. This becomes evident when we compare the measurements of nodes between time instant $t_{k-1}$ and time instant $t_k$ as shown in Figure 1B and 1C, respectively.

## 4 | THE PROPOSED APPROACH

We apply the modified Hampel Identifier algorithm for classifying three types of readings, that is, normal reading, event reading, and temporal outlier, for each sensor node. Based on this classification, we propose a new effective method for outlier detection in streaming data. Our proposal not only achieves a high identification rate but also distinguishes between event areas and outlier areas.

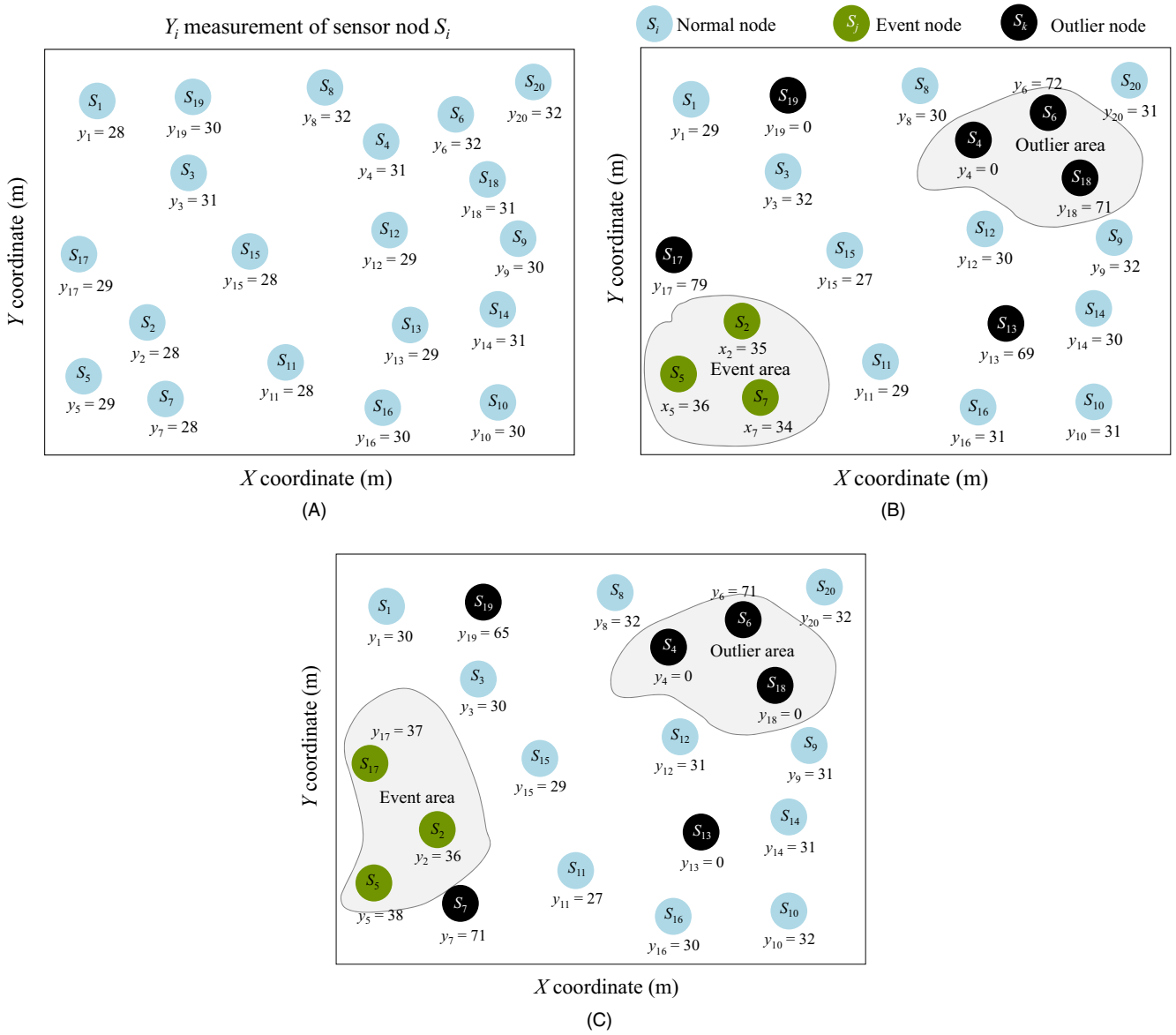


**FIGURE 1** Measurements of sensor nodes in the network. (A) Measurements of sensor nodes at time instant $t_{k-2}$. (B) Measurements of sensor nodes at time instant $t_{k-1}$. (C) Measurements of sensor nodes at time instant $t_k$

## 4.1 | Efficient communication technique in data collection

It is well known that one of the limitations of a wireless sensor network is the limited storage capability. Buffer overflow may occur at any time if a sensor node receives too much data from its neighbor nodes without data aggregation or data filtering. To tackle this problem, we propose an effective communication technique based on data filtering, which significantly reduces the number of transmissions. Our main idea in presenting an energy efficient communication technique in WSNs is partially inspired by our previous work in [38].

Given $N$ sensor nodes in the field of interest. To initialize the data filtering process, each sensor node broadcasts a "HELLO" message to the network using the controlled flooding method. This "HELLO" message consists of node ID number ($S_i$), location information ($l_i$ = (latitude$_i$, longitude$_i$)), hop ID number (HOP$_i$), and measurement values ($y_i$), (Figure 2). It is worth emphasizing that the hop ID number is set to 2 (HOP$_s$ = 2) at source node $S_s$. When a


**FIGURE 2** Format of the "HELLO" message

sensor node $S_d$ receives a "HELLO" message, it checks the value of the hop ID number. If this number is larger than 0, the node $S_d$ decreases this number by one. Then, the node $S_d$ forwards this message to its neighbor nodes.

To avoid loop data transmission (a message sent and received by the same node), the node must check the node ID number in the message before receiving it. The node accepts the message if the node ID number in that message is different from its node ID number and it comes from one of its 2-hop neighbor nodes. Obviously, the number of transmissions is significantly decreased by this technique. As shown in Figure 1A, sensor node $S_{13}$ has received data only from its 2-hop neighbor nodes (i.e., $\{S_{10}, S_{11}, S_{14}$, and $S_{16}\}$). The received data will be used for outlier detection before sending it to the base station.

## 4.2 | Temporal outlier detection techniques

### 4.2.1 | The Hampel Identifier Algorithm

This algorithm identifies the measurement $(y_i(t_k))$ of node $S_i$ as an outlier or inlier based on $k$ previous readings of that node. It is assumed that the observations come from iid random variables with common distribution $\mathcal{N}\left(\mu, \sigma^2\right)$. The details of this algorithm are illustrated in the following steps.

- **Step 1.** Choose the width of the moving window $L$, $\left\{0 < L \le \left\lceil \frac{(k-1)}{2} \right\rceil \right\}$ from $k$ observations.
- **Step 2.** Compute the median value: $y_0 = \tau(y) = \text{median}(y[(i-L):(i+L)])$;
- **Step 3.** Estimate the standard deviation for each observation:

$$\Gamma_0 = \Gamma(y) = H \times \text{median}(|y_{i-L:i+L} - y_0|) \tag{4}$$

where $H = 1.4826$ as given in [39];

- **Step 4.** Identify $y_i$ as a temporal outlier if

$$|y_i - y_0| > t_0 \times \Gamma_0$$

where $2 \le t_0 \le 5$.

One of the advantages of the HI algorithm is its ability to detect local outliers in a moving window. Unfortunately, the MAD scale estimator in HI is zero if more than half of the observations are identical. Furthermore, the authors [16] have claimed that if the input is an independent and identically distributed process, the HI algorithm may fail to detect the outliers due to highly autocorrelated data. As a matter of fact, it is hard to find the suitable threshold for each process model.

To overcome the aforementioned limitations and to improve the robustness in outlier detection, we propose a new algorithm based on modifying the HI. The superior performance of this method is illustrated in the following sections.

### 4.2.2 | The modified Hampel Identifier algorithm

For each $\alpha, \beta \in \mathcal{R}$, the measurement $y_i$ of sensor node $S_i$ is an outlier inside a dataset $y$ if and only if the rescaled measurement $\alpha y_i + \beta$ is an outlier inside the rescaled dataset $\alpha y + \beta$ [39]. The equivalence equation is given as follows.

$$\begin{aligned} &|y_i - \tau(y)| > t_0 \times \Gamma(y) \Leftrightarrow \\ &|\alpha y_i + \beta - \tau(\alpha y + \beta)| > t_0 \times \Gamma(\alpha y + \beta). \end{aligned} \tag{5}$$

Furthermore, the authors in [39] proved that $y_i$ is a temporal outlier if and only if

$$|y_i - \tau(y)| > t_0 \times \Gamma(y), \tag{6}$$

where

$$\begin{cases} y_i \to \alpha y_i + \beta, \\ \tau(\alpha y + \beta) = \alpha \tau(y) + \beta, \\ \Gamma(\alpha y + \beta) = |\alpha| \Gamma(y). \end{cases} \tag{7}$$

Based on these analyses, we propose the Temporal Outlier Detection (TOD) method using the modified HI as in Algorithm 1.

**Algorithm 1** TOD algorithm

**Input:** $y, \alpha, \beta$
**Output:**
+ a list of outliers $Y_{TO}$
+ a number of outliers $O$
**Initiation:** $Y_{TO} = \emptyset; H = 1.4826; t_0 = 3; L = 5; n = length(y); O = 0;$

**procedure** TOD_ALGORITHM($y, \alpha, \beta$)
  **for** ($i \leftarrow L+1$ to $n-L$) **do**
    Consider the moving data window:
      $W_i^L = \{\hat{y}_{i-L}, ..., \hat{y}_i, ..., \hat{y}_{i+L}\}$.
    Calculate:
      $y_0 \leftarrow \tau(y)$
      $\Gamma_0 \leftarrow \Gamma(y)$
      $y_i \leftarrow \alpha y_i + \beta$
      $\tau(\alpha x + \beta) \leftarrow \alpha \tau(y) + \beta$
      $\Gamma(\alpha y + \beta) \leftarrow |\alpha| \Gamma(y)$.
    **if** ($|y_i - \tau(y)| > t_0 . \Gamma(y)$) **then**
      $Y_{TO} \leftarrow Y_{TO} \cup y_i$
      $O \leftarrow O + 1;$
      $y_i \leftarrow MAD(W_i^L)$.
    **end if**
  **end for**
**end procedure**

Similar to that in the HI algorithm, we assume that all the observations and historical records of the sensing processes can be represented using stationary stochastic models.

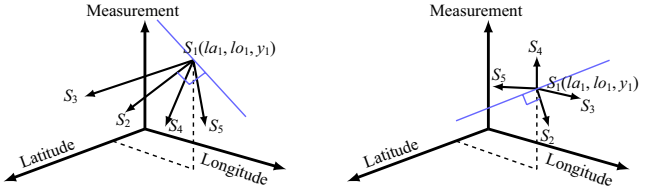


**FIGURE 3** Intuition of (distance, direction)-based outlier detection. (A) Measurement of $S_1$ is an outlier. (B) Measurement of $S_1$ is an inlier

Therefore, our proposed TOD algorithm can be used for outlier detection in these models.

## 4.3 | Spatial outlier detection algorithm

The TOD algorithm identifies the temporal outliers with a high accuracy identification rate. However, it may miss some true spatial outliers. Furthermore, the spatial data are critical for classifying the event areas and outlier areas. Therefore, in this section, we propose a new spatial outlier detection method, which detects the outlying nodes and outlier areas inside a spatial dataset with a high identification rate and within a short execution time.

### 4.3.1 | Basic idea of spatial outlier detection

As mentioned in the previous sections, there may not be sufficient information to identify the spatial outliers if we use only the distances between points and the angles between pairs of distance vectors. In this section, we combine the advantages of the angle-based method and the distance-based method to propose a new spatial outlier detection method. The main idea of this method is illustrated in Figure 3.

We take an example of a WSN with five sensor nodes, which are randomly deployed in the sensing field. Figure 3A shows that the measurement of node $S_1$ is an outlier because the distances between $S_1$ and its 2-hop neighbor nodes $\{S_2, S_3, S_4, S_5\}$ are long. Additionally, vectors $\left\{\overrightarrow{S_1S_3}, \overrightarrow{S_1S_4}, \overrightarrow{S_1S_5}\right\}$ lie on the same side of the plane with vector $\overrightarrow{S_1S_2}$. In contrast, Figure 3B shows that the measurement of node $S_1$ is an inlier because the distances between this node and its neighbor nodes are short. Additionally, vectors $\left\{\overrightarrow{S_1S_4}, \overrightarrow{S_1S_5}\right\}$ lie on the opposite side of the plane with vector $\overrightarrow{S_1S_2}$. Based on these observations, we state the following remarks.

*Remark 1* Consider a WSN in a three-dimensional space, where the horizontal and vertical axes represent the latitude and longitude of the nodes, respectively. The third dimension represents the measurement of the nodes. The measurement of a node is an outlier at a time instant if the distances between this node to all its 2-hop neighbor nodes are long. In addition, the vectors from this node to its 2-hop neighbor nodes lie on the same side of the plane.

### 4.3.2 | Direction-based Spatial Outlier Detection (SOD)

We present a direction-based outlier detection technique in our spatial outlier detection algorithm. To facilitate our proposed SOD algorithm, we first introduce the following theorem, which constructs the SOD algorithm.

**Theorem 1** *Let* $S_i(la_i, lo_i, y_i)$, $(i = 1{:}N)$ *denote the information (including the latitude, longitude, and measurement) of node* $S_i$. *Let* $S_j(la_j, lo_j, y_j)$, $S_k(la_k, lo_k, y_k)$, $\{j, k = 1:N, i \neq j \neq k\}$ *denote the information of two nodes* $S_j$, $S_k$ *among the 2-hop neighbor nodes of node* $S_i$. *The vectors* $\{\overrightarrow{S_iS_j}, \overrightarrow{S_iS_k}\}$ *lie on the opposite sides of a plane if and only if:*

$$\begin{aligned}&(la_j - la_i)(la_k - la_i) + (lo_j - lo_i)(lo_k - lo_i)\\ &+ (y_j - y_i)(y_k - y_i) < 0.\end{aligned} \tag{8}$$

*Proof* From three points $S_i(la_i, lo_i, y_i)$, $S_j(la_j, lo_j, y_j)$, and $S_k(la_k, lo_k, y_k)$, $\{i, j, k = 1{:}N, i \neq j \neq k\}$, we always have two different vectors:

$$\overrightarrow{S_iS_j} = \left((la_j - la_i), (lo_j - lo_i), (y_j - y_i)\right) \text{ and}$$
$$\overrightarrow{S_iS_k} = \left((la_k - la_i), (lo_k - lo_i), (y_k - y_i)\right).$$

In Ref. [40], the dot product of two vectors $\left\{\overrightarrow{S_iS_j}, \overrightarrow{S_iS_k}\right\}$ satisfies the following equations:

$$\begin{aligned}\overrightarrow{S_iS_j}.\overrightarrow{S_iS_k} &= (la_j - la_i)(la_k - la_i)\\ &+ (lo_j - lo_i)(lo_k - lo_i)\\ &+ (y_j - y_i)(y_k - y_i)\end{aligned} \tag{9}$$

and

$$\overrightarrow{S_iS_j}.\overrightarrow{S_iS_k} = \| S_iS_j \| \| S_iS_k \| \cos\left\langle \widehat{\overrightarrow{S_iS_j}, \overrightarrow{S_iS_k}} \right\rangle. \tag{10}$$

From (9) and (10), we have

$$\| S_iS_j \| \| S_iS_k \| \cos\left\langle \widehat{\overrightarrow{S_iS_j}, \overrightarrow{S_iS_k}} \right\rangle < 0 \tag{11}$$

or

$$\cos\left\langle \widehat{\overrightarrow{S_iS_j}, \overrightarrow{S_iS_k}} \right\rangle < 0 \tag{12}$$

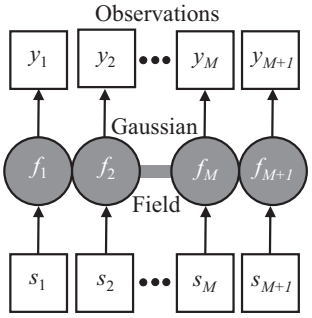


**FIGURE 4** Graphical model of the GP for regression. Squares represent observed variables and circles represent unknowns [42]

or

$$\frac{\pi}{2} < \left\langle \widehat{\overrightarrow{S_iS_j}, \overrightarrow{S_iS_k}} \right\rangle < \frac{3\pi}{2}. \tag{13}$$

It means that the two vectors $\left\{\overrightarrow{S_iS_j}, \overrightarrow{S_iS_k}\right\}$ lie on the opposite sides of a plane. ■

### 4.3.3 | Gaussian process

We propose a Gaussian process-based method to predict the unobserved variables in the collected dataset. Hence, we use a stationary Gaussian Process (GP) to model the measurement of sensor nodes. Then, this information is utilized to estimate the measurement uncertainty at different locations of the sensor nodes. These predicted values are used for spatial outlier detection.

Let $s = (s_1, s_2, \ldots, s_N)$ denotes the location of $N$ nodes, where $s_i = (\text{latitude}_i, \text{longitude}_i)$. Let $y_i$ denotes the measurement of sensor node $S_i$. Since sensor node $S_i$ has $(M-1)$ 2-hop neighbor nodes, where $1 < M \leq N$, we take into account $M$ nodes only. Hence, we have to look for a mapping from input space $s = [s_1, s_2, \ldots, s_M]$ to output space $y = [y_1, y_2, \ldots, y_M]$. The spatial correlation function between $s_i$ and $y_i$ is given in [41]. In that research, the signal strength at node $S_p$, which is received from node $S_q$, is calculated as follows:

$$y_{pq} = P_r - P_0 - 10\eta \log_{10}\left(\frac{d_{pq}}{d_0}\right) + X_\sigma \tag{14}$$

where $P_r$ is the power of the received signal, $P_0$ is a reference power at a reference distance $d_0$, $d_{pq}$ is the distance between $S_p$ and $S_q$, $\eta$ is the path loss index, and $X_\sigma$ is a zero-mean Gaussian random variable with standard deviation $\sigma$.

Let $f(s)$ denotes the function modeled by the GP, which is given as follows:

$$f(s) \sim \text{GP}\left(m(s), k(s_p, s_q)\right), \tag{15}$$

where $m(s) = E\left[f(s)\right]$ is the mean function of the distribution at $s$, and $k(s_p, s_q)$ is the kernel function of the covariance function, $s_p, s_q \in s$ (refer [42] for more details). The graphical model of the GP for regression is given in Figure 4.

We assume that the covariance function is a squared exponential (SE), and it can be written as follows.

$$k(d_{pq}) = \sigma_0^2 \exp\left(-\frac{d_{pq}^2}{2\lambda^2}\right) \tag{16}$$

where $d_{pq} = \| s_p - s_q \|$ is the Euclidean distance between $s_p$ and $s_q$, and the two hyperparameters $\sigma_0$ and $\lambda$ are the amplitude and length scale, respectively.

For convenience, let $s_{1:M}$ refers to the set of $M$ data points. Suppose that a set of localizations $s_{1:M}$ has its corresponding approximate measurements $\{f(s_1), f(s_2), \ldots, f(s_M)\}$, which are denoted as $f_{1:M}$. It turns out that each $f(s_i)$ corresponds to a data point $s_i$ with probability $p(s_i)$. Therefore, the training data for the GP are defined as follows: $C_{1:M} = \{(s_1, f(s_1)), (s_2, f(s_2)), \ldots, (s_M, f(s_M))\}$. We assume that $C_{1:M}$ follows the GP model whose initial mean function is zero and the covariance function is given by the kernel function as follows:

$$K = \begin{bmatrix} k(s_1, s_1) & \cdots & k(s_1, s_M) \\ \vdots & \ddots & \vdots \\ k(s_M, s_1) & \cdots & k(s_M, s_M) \end{bmatrix}. \tag{17}$$

After training the GP with $C_{1:M}$, the new measurement of node $S_{M+1}$ will be predicted by the GP model. The set of predicted data $(s_{M+1}, f(s_{M+1}))$ is calculated using the GP properties as follows:

$$\begin{bmatrix} f_{1:M} \\ f_{M+1} \end{bmatrix} = \mathcal{N}\left(\begin{bmatrix} m(s_{1:M}) \\ m(s_{M+1}) \end{bmatrix}, \begin{bmatrix} K & k \\ k^T & k(s_{M+1}, s_{M+1}) \end{bmatrix}\right) \tag{18}$$

where $k = [k(s_{M+1}, s_1), k(s_{M+1}, s_2), \ldots, k(s_{M+1}, s_M)]$. By applying Bayes' theorem, we can calculate the approximate probability of the predicted measurement value $f_{M+1}$ from location $s_{M+1}$ as follows:

$$P\left(f_{M+1} | C_{1:M}, s_{M+1}\right) = \mathcal{N}\left(\mu_M(s_{M+1}), \sigma_M^2(s_{M+1})\right) \tag{19}$$

where

$$\begin{cases} \mu_M(s_{M+1}) = kK^{-1}f_{1:M}, \\ \sigma_M^2(s_{M+1}) = k(s_{M+1}, s_{M+1}) - kK^{-1}k^T. \end{cases} \tag{20}$$

### 4.3.4 | Lack of information in GP training

As described above, each sensor node has only measurements from $(M-1)$ in its 2-hop neighbor sensor nodes. In

**TABLE 1** The observed data in Ref. [47]

| $y_1$ | $y_2$ | $y_3$ | $y_4$ | $y_5$ | $y_6$ | $y_7$ | $y_8$ | $y_9$ | $y_{10}$ |
|---|---|---|---|---|---|---|---|---|---|
| 22.6 | 28.8 | 26.8 | 81.5 | 19.1 | 15.2 | 24.1 | 23.6 | 9.1 | 79.5 |
| $y_{11}$ | $y_{12}$ | $y_{13}$ | $y_{14}$ | $y_{15}$ | $y_{16}$ | $y_{17}$ | $y_{18}$ | $y_{19}$ | $y_{20}$ |
| 18.6 | 78.8 | 23.1 | 11.9 | 20.1 | 20.3 | 17.3 | 25.8 | 14.1 | 26.5 |

some cases, this may not provide sufficient information for the GP training process. To tackle this problem, each sensor node should estimate some new data points within its sensing area. We call such nodes, which lack information, as candidate nodes $S_c$. The estimation process is implemented using the two following steps.

- **Step 1.** $S_c$ finds a source node $S_s$ among its 2-hop neighbor nodes. The location of $S_s$ is determined using $s_s = \text{argmax}_{s_i \in s_{1:M-1}} f(s_i)$.
- **Step 2.** Based on the received signal strength in (14), $S_c$ estimates $V$ more measurements within its sensing area. The measurements of these $V$ locations $s_e$, $\{e = 1 : V\}$ are calculated as follows:

$$f_{(s_e)} = P_r - P_0 - 10\eta \log_{10}\left(\frac{d_{es}}{d_0}\right) + X_\sigma \tag{21}$$

where $d_{es} = \| s_e - s_s \|$ is the Euclidean distance between $S_e$ and $S_s$.

### 4.3.5 | Spatial outlier detection

As presented in [43–45], the authors applied the value of the posterior mean $\mu_M$ in the GP to predict the anomaly. After training the GP, a measurement $y_{M+1}$ is an outlier in the given dataset if its posterior mean is larger than the mean of the other measurements; otherwise, it is an inlier. Furthermore, the posterior variance of the anomaly $\sigma^2(s_{M+1})$ is higher than that of the variance of the normal. The question is how much deviation of $\mu_M$ and $y_{M+1}$ is suitable for identifying an outlier. To answer this question, we use the Neyman-Pearson hypothesis [46] to find the optimal value of $\theta$.

$$\theta_{\text{opt}} := \min_{\theta} \left\{ P\left( |y_i(t_k) - \mu_i| \geq \theta \right) \right\}. \tag{22}$$

Based on the aforementioned analyses, we propose a new outlier detection algorithm using the GP. The procedure for the SOD algorithm is as follows.

**Algorithm 2** SOD algorithm

**Input:** $M, s_{i_{1:M}}$
**Output:**
+ predicted dataset $(s_{M+1}, f(s_{M+1}))$
+ list of spatial outlier $Y_{SO}$
**Initial:** List of checked nodes $L_c = \emptyset$, list of nodes in the area $L_a = \{S_1, ..., S_M\}$

**procedure** SOD_ALGORITHM($M, s_{i_{1:M}}$)
  **while** $(L_c < L_a)$ **do**
    $S_c \leftarrow MAX(|latitute_i|, |longtitute_i|, |f(s_i)|), i = 1, ..., M$
    **if** $(S_c \notin H)$ **then**
      $L_c \leftarrow L_c \cup S_c$.
      Construct training data from $(M - 1)$ 2-hop neighbor nodes of the candidate node $S_c$.
      $C_{1:M} = \left\{ \left(s_1, f(s_1)\right), \left(s_2, f(s_2)\right), ..., \left(s_M, f(s_M)\right) \right\}$
      Estimate $V$ measurements within the sensing range of $S_c$.
      $C_{1:V} = \left\{ \left(s_1, f(s_1)\right), \left(s_2, f(s_2)\right), ..., \left(s_V, f(s_V)\right) \right\}$
      Train the GP based on the dataset $\left\{C_{1:M}, C_{1:V}\right\}$, and then predict the measurement $\left(f_{M+1} \left| s_{M+1} = s_c \right.\right)$ at the position of candidate node $S_c$.
      **if** $(\left| y_{M+1} - \mu(s_{M+1}) \right| \geq \theta$ and the vectors between $S_c$ and its 2-hop neighbor nodes lie on the same side of the plane) **then**
        $Y_{SO} \leftarrow Y_{SO} \cup y_i$.
        $O = O + 1$.
        Replace $y_i$ by its estimated value.
      **else**
        Mark $y_i$ as an inlier.
      **end if**
    **end if**
  **end while**
**end procedure**

## 5 | PERFORMANCE EVALUATION

To evaluate the performance of our algorithms, we conducted experiments on both synthetic and real-world datasets. The comparison among algorithms TOD, SOD, HI, and ABOD was made by (a) analyzing the results when using these

algorithms to detect outliers in the dataset, (b) testing the generated dataset, and (c) testing a real-world dataset.

## 5.1 | Numerical analysis on the observed data in Ref. [47]

Let us consider the observed data in [47]. In that study, the authors have identified three outliers (81.5, 79.5, and 78.8) among the observations. This result is used for comparison with the outcomes of the HI algorithm and TOD algorithm (Table 1).

Firstly, we detect the outliers in the observed data by using HI with *window width* $L = 5$ and $t_0 = 3$. All the parameters in the HI algorithm are given in Table 2. As we can see, the Hampel Identifier detects only two outliers, that is, $y_4$, and $y_{12}$, in the observed data, while $y_{10}$ is detected as an inlier.

As mentioned above, due to "a symmetric view on dispersion," and a highly autocorrelated dataset, HI may fail to identify the outliers. Therefore, $y_{10}$ is identified as an inlier. The reason is that two outliers (i.e., $y_{10}$, $y_{12}$) occur in a short time. More concretely, at $y_{10}$, we have $|y_10 - y_0| = 55.9$ and $t_0 \times S_0 = 64.49$. Obviously, $|y_{10} - y_0| < t_0 \times S_0$. Hence, $y_{10}$ is identified as an inlier while it is actually an outlier.

In order to compare the accuracy of detecting outliers between the HI algorithm and the TOD algorithm, we apply the TOD algorithm with $(\alpha = 1.2, \beta = 0.8)$. Using these setting parameters, the TOD algorithm identifies all three outliers correctly. Table 3 shows the results of our algorithm.

## 5.2 | Numerical analysis on detecting outliers in the monthly series of the Italian Industrial Production Index from 1981 to 1996

In this section, we apply the HI algorithm and TOD algorithm to detect the outliers in the monthly series of the Italian Industrial Production Index from 1981 to 1996, which is available in the *tsoutlier R* package [48]. Table 4 shows the investigated data series, which has been analyzed by many authors [49,50]. Therefore, it is really useful for comparing the HI algorithm and TOD algorithm. In the HI algorithm, we set $L = 5$, $t_0 = 3$. In the TOD algorithm, we set $(\alpha = 1.0, \beta = 1.0)$. The results are given in Table 4. The results demonstrate that our proposed TOD algorithm detects 16 of 18 outliers correctly. However, for the HI algorithm, 18 of 34 detected outliers are FP.

**TABLE 2** Outlier detection by HI

| Data process | | $y_0$ | $\lvert y_i - y_0 \rvert$ | $t_0 \times \Gamma_0$ | Outlier |
|---|---|---|---|---|---|
| $y_1$ | 22.6 | - | - | - | N/A[a] |
| $y_2$ | 28.8 | - | - | - | N/A |
| $y_3$ | 26.8 | 26.8 | 0 | 18.68 | No |
| $y_4$ | 81.5 | 26.8 | **54.7** | **34.25** | **Yes** |
| $y_5$ | 19.1 | 24.1 | 5.0 | 12.01 | No |
| $y_6$ | 15.2 | 23.6 | 8.4 | 14.23 | No |
| $y_7$ | 24.1 | 19.1 | 5.0 | 20.02 | No |
| $y_8$ | 23.6 | 23.6 | 0 | 37.36 | No |
| $y_9$ | 9.10 | 23.6 | 14.5 | 22.24 | No |
| **$y_{10}$** | **79.5** | **23.6** | **55.9** | **64.49** | **No** |
| $y_{11}$ | 18.6 | 23.1 | 4.5 | 62.27 | No |
| $y_{12}$ | 78.8 | 23.1 | **55.7** | **49.82** | **Yes** |
| $y_{13}$ | 23.1 | 20.1 | 3.0 | 13.34 | No |
| $y_{14}$ | 11.9 | 20.3 | 8.4 | 12.45 | No |
| $y_{15}$ | 20.1 | 20.1 | 0 | 12.45 | No |
| $y_{16}$ | 20.3 | 20.1 | 0.2 | 12.45 | No |
| $y_{17}$ | 17.3 | 20.1 | 2.8 | 12.45 | No |
| $y_{18}$ | 25.8 | 20.3 | 5.5 | 23.13 | No |
| $y_{19}$ | 14.1 | - | - | - | N/A |
| $y_{20}$ | 26.5 | - | - | - | N/A |

[a]N/A: Not available.
All the bold values are predefined as outliers, and otherwise (normal values) are inliers.

**TABLE 3** Outlier detection by TOD

| Data process | | $y_0$ | $\lvert y_i - y_0 \rvert$ | $t_0 \times \Gamma_0$ | Outlier |
|---|---|---|---|---|---|
| $y_1$ | 22.6 | - | - | - | N/A[a] |
| $y_2$ | 28.8 | - | - | - | N/A |
| $y_3$ | 26.8 | 32.96 | 0 | 18.72 | No |
| $y_4$ | 81.5 | 32.96 | **65.64** | **34.33** | **Yes** |
| $y_5$ | 19.1 | 29.72 | 6.00 | 22.29 | No |
| $y_6$ | 15.2 | 29.12 | 10.08 | 20.06 | No |
| $y_7$ | 24.1 | 23.72 | 6.00 | 20.06 | No |
| $y_8$ | 23.6 | 29.12 | 0 | 37.45 | No |
| $y_9$ | 9.10 | 29.12 | 17.40 | 22.29 | No |
| **$y_{10}$** | **79.5** | **29.12** | **67.08** | **64.64** | **Yes** |
| $y_{11}$ | 18.6 | 28.52 | 5.40 | 62.42 | No |
| $y_{12}$ | 78.8 | 28.52 | **66.84** | **49.93** | **Yes** |
| $y_{13}$ | 23.1 | 24.92 | 3.60 | 13.37 | No |
| $y_{14}$ | 11.9 | 25.16 | 10.08 | 12.48 | No |
| $y_{15}$ | 20.1 | 24.92 | 0 | 12.48 | No |
| $y_{16}$ | 20.3 | 24.92 | 0.24 | 12.48 | No |
| $y_{17}$ | 17.3 | 24.92 | 3.36 | 12.48 | No |
| $y_{18}$ | 25.8 | 25.16 | 6.60 | 24.52 | No |
| $y_{19}$ | 14.1 | - | - | - | N/A |
| $y_{20}$ | 26.5 | - | - | - | N/A |

[a]N/A: Not available.
All the bold values are predefined as outliers, and otherwise (normal values) are inliers.

**TABLE 4** Outlier detection in the Italian industrial production index 1981 to 1996

| Time | Jan | Feb | Mar | Apr | May | June | July | Aug | Sept | Oct | Nov | Dec |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1981 | 86.3 | 87.6 | 96.3 | 90.4 | 90.4 | **94.4**[a] | 95.2 | **36.6**[a,b] | 96.1 | 95.6 | 92.8 | 77.3 |
| 1982 | 85.1 | 86.8 | 96.9 | 90.5 | 88.5 | 87.9 | 90.2 | **36.9**[a,b] | 92.3 | 88.0 | 86.8 | 77.1 |
| 1983 | 80.7 | 82.3 | 92.6 | 79.5 | 87.0 | **86.7**[a] | **84.8**[a] | **38.4**[a,b] | 90.8 | 87.7 | 89.5 | **74.6**[a,b] |
| 1984 | 85.4 | 85.3 | 92.6 | 78.9 | 93.3 | 90.0 | 88.6 | **43.0**[a,b] | 89.3 | **97.5**[a] | 89.7 | **73.7**[a] |
| 1985 | 84.1 | 87.3 | 92.7 | 84.5 | 93.9 | 88.4 | 93.9 | **39.1**[a,b] | 91.4 | 96.5 | 89.7 | 77.0 |
| 1986 | 86.2 | 88.9 | 92.6 | 93.6 | 92.4 | 91.8 | **99.0**[a] | **37.5**[a,b] | 97.9 | 101.0 | 91.3 | 83.3 |
| 1987 | 82.9 | 90.9 | 102.2 | 95.3 | 96.0 | **100.5**[a] | **100.5**[a] | **39.3**[a,b] | 100.3 | 103.0 | 99.1 | 86.5 |
| 1988 | 89.4 | 99.8 | 109.5 | 94.2 | 104.2 | 106.1 | 100.8 | **46.9**[a,b] | 107.2 | 104.3 | 106.3 | 93.3 |
| 1989 | 98.3 | 101.4 | 109.0 | 97.5 | 107.5 | 110.3 | **104.2**[a] | **50.3**[a,b] | 108.1 | 112.2 | 109.5 | 89.8 |
| 1990 | 101.2 | 101.8 | 112.6 | 98.1 | 110.3 | 105.8 | 109.0 | **51.1**[a,b] | 103.6 | **113.8**[a] | 104.6 | **88.2**[a,b] |
| 1991 | 102.9 | 99.6 | 105.9 | 97.6 | 108.6 | 103.1 | 110.9 | **46.3**[a,b] | 106.9 | **112.8**[a] | 105.3 | 89.1 |
| 1992 | 98.1 | 102.8 | **111.5**[a] | 102.4 | **103.1**[a] | 109.7 | 111.1 | **43.4**[a,b] | 104.1 | 107.1 | 105.4 | 87.8 |
| 1993 | 89.3 | 98.7 | 111.3 | 98.5 | 102.4 | 105.7 | 103.9 | **45.0**[a,b] | 104.5 | 101.9 | 104.1 | 92.9 |
| 1994 | 90.2 | 99.4 | **113.3**[a] | **97.9**[a] | 110.6 | 112.4 | **108.8**[a] | **52.6**[a,b] | **112.9**[a] | 109.2 | 111.7 | 99.3 |
| 1995 | 102.8 | 107.6 | 123.6 | 98.9 | 117.5 | 117.7 | 113.4 | **58.5**[a,b] | 114.1 | 117.8 | 115.5 | 96.6 |
| 1996 | 106.1 | 111.3 | 115.6 | 103.5 | 115.3 | 110.1 | **118.1**[a] | **52.0**[a,b] | 110.7 | 118.2 | 108.1 | 93.6 |

[a]Outlier detected by HI.
[b]Outlier detected by TOD.
All the bold values are predefined as outliers, and otherwise (normal values) are inliers.

### 5.3 | Numerical analysis on detecting spatial outliers

In this section, the performance of the SOD algorithm is analyzed and compared with some typical methods in spatial outlier detection. We briefly summarize a typical method of spatial outlier detection, which will serve as a reference for evaluating the performance of our proposed SOD algorithm. In this paper, each sensor node $S_i$ is treated as a point in three-dimensional space, that is, $S_i$(latitude, longitude, measurement). We use the pairs of points between the candidate node and its 2-hop neighbor nodes to construct vectors. The angle-based outlier factor (ABOF) is the variance over the angles between these vectors.

#### 5.3.1 | How to use GP in spatial outlier detection

In this section, we use GP to detect outliers in the network shown in Figure 1. At the time instant $t_k$, the measurement of the candidate node $S_{13}(6.8, 3.2)$ is $y_{13} = 0$, while the received data from its 2-hop neighbor nodes are $y_{10} = 32$; $y_{11} = 27$; $y_{14} = 31$; $y_{16} = 30$. Therefore, we have four measurements $y = [32, 27, 31, 30]$ of these nodes with the corresponding locations $s = \{S_{10}(8, 1.2), S_{11}(4, 2), S_{14}(6.2, 3.6), S_{16}(6, 1)\}$. Now, we train the GP with two input vectors $s_{\text{latitude}} = [8, 4, 6.2, 6]$, $s_{\text{longitude}} = [1.2, 2, 3.6, 1]$ and an output vector $y = [32, 27, 31, 30]$. After training, we use the latitude input vector and longitude input vector to estimate the measurements of the candidate nodes, $\mu_{\text{latitude}}(s_{13})$ and $\mu_{\text{longitude}}(s_{13})$, respectively. The final estimated measurement of candidate node is the mean of $\mu_{\text{lattitude}}(s_{13})$ and $\mu_{\text{longtitude}}(s_{13})$ $\mu(s_{13}) = 0.5\left(\mu_{\text{latitude}}(s_{13}) + \mu_{\text{longitude}}(s_{13})\right)$.

We start with the latitude input vector $s_{\text{latitude}} = [8, 4, 6.2, 6]$, and the output vector $y = [32, 27, 31, 30]$. We choose $\sigma_0 = 0.3$ as in (16), and then calculate a covariance matrix using (17):

$$K = \begin{bmatrix} 1.690 & 1.631 & 1.677 & 1.675 \\ 1.631 & 1.690 & 1.671 & 1.675 \\ 1.677 & 1.671 & 1.690 & 1.689 \\ 1.675 & 1.675 & 1.698 & 1.690 \end{bmatrix}.$$

We get $k = [1.684 \quad 1.660 \quad 1.688 \quad 1.687]$ and $k(s_{13}, s_{13}) = 1.69$. From (20), we have

$$\begin{cases} \mu_{\text{latitude}}(s_{13}) = 31.328, \\ \sigma^2_{\text{latitude}}(s_{13}) = 3.73 \times 10^{-4}. \end{cases} \tag{23}$$

Now, we consider the longitude input vector $s_{\text{latitude}} = [1.2, 2, 3.6, 1]$, and the output vector $y = [32, 27, 31, 30]$. We have

$$\begin{cases} \mu_{\text{longitude}}(s_{13}) = 31.467, \\ \sigma^2_{\text{longitude}}(s_{13}) = 3.73 \times 10^{-4}. \end{cases} \tag{24}$$

From (23) and (24), we have $\mu(s_{13}) = 31.397$, which satisfies equation $|y_{M+1} - \mu(s_{M+1})| \geq \theta$. We conclude that it is a spatial outlier at the time instant $t_k$.

### 5.3.2 | Numerical analysis on synthetic data

We apply our SOD algorithm and ABOD algorithm to test the generated datasets, in which some outliers are added using the exponential distribution. More precisely, we consider a WSN with $N = 100$ sensor nodes, which are randomly deployed in the sensing field in the area $100 \times 100$ ($m^2$).

In the training phase, at each sensor node $S_i$, $i = 1 : N$, we generate a random time series data with a size of $k = 1{,}000$ using an autoregressive $AR(Z = 3)$ as in the following model:

$$y_{i,k} = \sum_{j=1}^{Z} a_j y_{i,k-j} + v_k, \tag{25}$$

where the autoregressive coefficients are set as an annual mean minimum temperature model in Ref. [51]: $a_1 = 0.4457$; $a_2 = 0.1662$; $a_3 = 0.2266$ and $v_k$ is iid subject to $\mathcal{N}(0, 1)$. The observation is $\widehat{y_{i,k}} = y_{i,k} + \chi_k$, where $\chi_k$ is an added outlier.

In the testing phase, we apply outlier detection algorithms to test the dataset. We generate $T$ outliers and add them randomly into the dataset, which is distributed exponentially with parameter $\lambda = 1$. The comparison of the identification rate between our SOD algorithm and the ABOD is given in Table 5. The portion of outliers is the proportion of the number of predefined outliers over the size of the dataset. The simulation results demonstrate that the identification rate of all the algorithms will be decreased corresponding to the increase in the number of predefined outliers or the portion of outliers. However, with higher values of CP, the SOD algorithm has higher identification rates compared to the ABOD algorithm.

As expected, 76% of the outliers are identified correctly by our algorithm, while the HI algorithm identified 72%, as shown in Table 5. With a high IR value, the SOD algorithm can correctly detect the outliers in time-series data, which is useful for real-time decision-making.

### 5.3.3 | Numerical analysis on real data

We investigated the effectiveness of our algorithm by applying it to a real dataset from Intel Berkeley Research lab between February 28 and April 5, 2004 [52]. The sensor nodes were arranged in a field according to the diagram shown in Figure 5. As depicted in Figure 5, the coordinates of the sensor nodes are given relative to the upper right corner of the lab. For a period of 31 seconds, each Mica2Dot sensor node recorded the temperature, humidity, light, and voltage values. Thus, over 2.2 million readings were collected by these sensor nodes. However, we only focus on detecting outliers in the dataset generated on March 6, 2004 for the temperature and humidity values by sensor nodes $S_1$, $S_2$, $S_3$, $S_{33}$, $S_{34}$, $S_{35}$, and $S_{37}$. We call them the IBRL dataset. The simulation results are shown in Figures 6 and 7.

**TABLE 5** Comparison of the identification rate in outlier detection

| Outlier | | Angle-based approach | | | SOD algorithm | | |
|---|---|---|---|---|---|---|---|
| % | T | OD | CP | IR | OD | CP | IR |
| 5 | 50 | 58 | 42 | 0.72 | 53 | 47 | 0.89 |
| 5 | 100 | 105 | 92 | 0.88 | 112 | 96 | 0.86 |
| 10 | 100 | 120 | 93 | 0.78 | 110 | 91 | 0.83 |
| 10 | 200 | 225 | 185 | 0.82 | 210 | 179 | 0.85 |
| 15 | 150 | 170 | 124 | 0.73 | 160 | 131 | 0.82 |
| 15 | 300 | 361 | 265 | 0.73 | 332 | 262 | 0.79 |
| 20 | 200 | 246 | 190 | 0.77 | 229 | 185 | 0.81 |
| 20 | 400 | 487 | 342 | 0.70 | 452 | 324 | 0.72 |
| 50 | 500 | 592 | 450 | 0.76 | 565 | 429 | 0.76 |
| 50 | 1,000 | 1,246 | 876 | 0.70 | 1,245 | 865 | 0.69 |
| 75 | 750 | 815 | 552 | 0.68 | 851 | 534 | 0.63 |
| 75 | 1,500 | 1,824 | 1,124 | 0.62 | 1,719 | 1,210 | 0.70 |
| 100 | 1,000 | 1,342 | 762 | 0.57 | 1,321 | 861 | 0.65 |
| 100 | 2,000 | 2,671 | 1,586 | 0.59 | 2,408 | 1,517 | 0.63 |
| **Average** | **733** | | | **0.72** | | | **0.76** |

Table 6 presents some events in the network. The source of an event is the node that has the longest distance between its measurements and the measurements of its neighbor nodes in an event area.

It should be noted that sensor node $S_{37}$ is located in the kitchen, where we expect a higher temperature and a lower humidity. However, the temperature captured from 08:54:20 to 14:14:05 on March 6, 2004 by sensor node $S_{34}$ is higher than the sensing data of any of its neighbor nodes $S_1$, $S_2$, $S_3$, $S_{33}$, $S_{34}$, $S_{35}$, and $S_{37}$. Therefore, sensor node $S_{34}$ becomes the

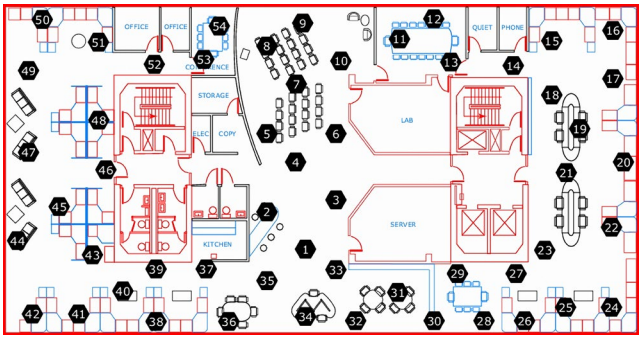


**FIGURE 5** The sensor deployment in the Intel Berkeley Research Lab [52]

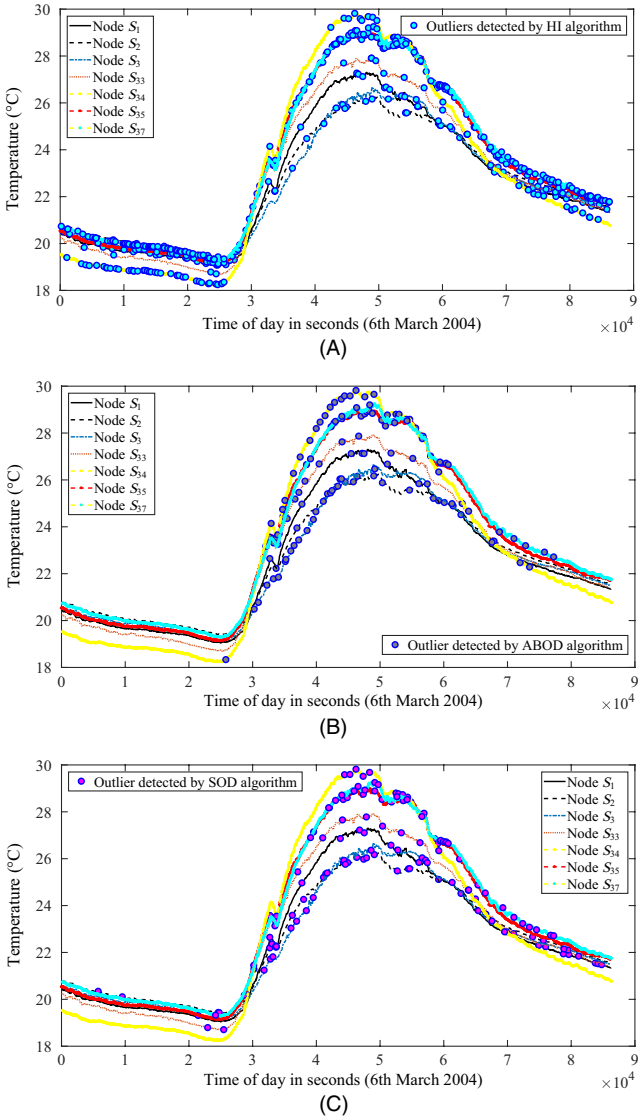


**FIGURE 6** Outlier detection in temperature values (A) Outlier detection by HI (B) Outlier detection by ABOD (C) Outlier detection by SOD

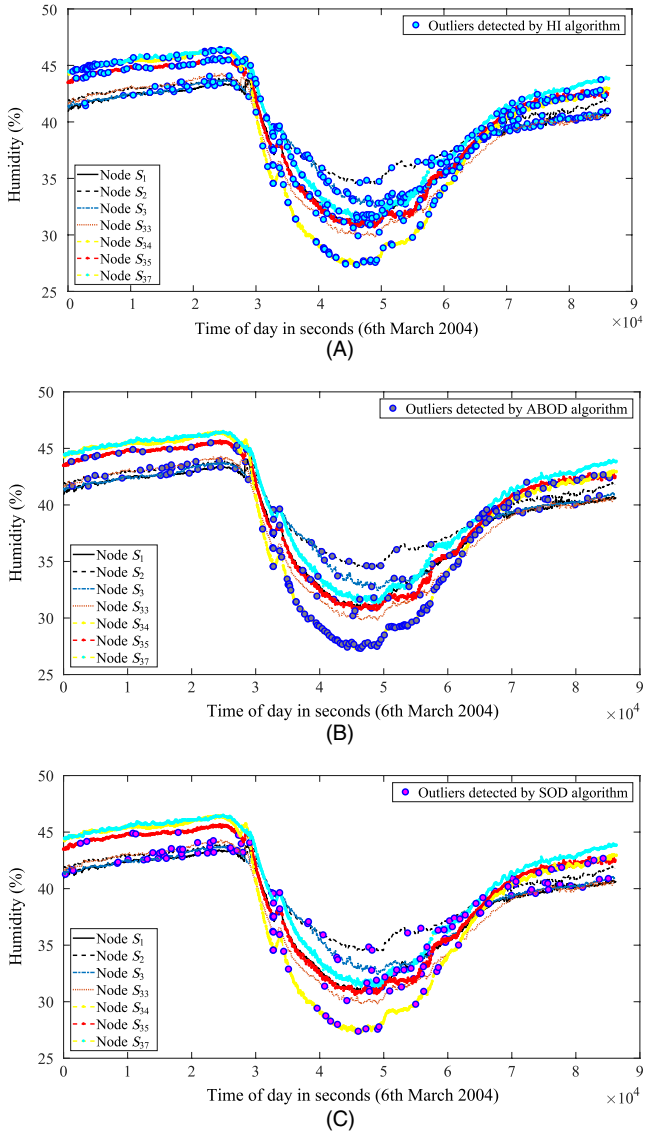


**FIGURE 7** Outlier detection in humidity values (A) Outlier detection by HI (B) Outlier detection by ABOD (C) Outlier detection by SOD

source of a temperature event during this time. From Figures 6A and 7A, it can be seen that a large number of outliers are detected by the HI algorithm, and these outlier points occur over time. It means that the HI algorithm identifies only unusual changes in the time series. This technique ignores the locations of the nodes, from where the sensing data are collected. As a result, the HI algorithm fails to detect some event areas that occur in the network. Furthermore, as mentioned above, the HI algorithm may encounter some limitations while working with highly autocorrelated data processes. Therefore, most of these outliers detected by the HI algorithm are false detection points. This is the main reason behind the low accuracy identification rate of the HI algorithm. To overcome this limitation, the ABOD algorithm and our proposed SOD algorithm consider the location of the generated data. As a result, these algorithms achieve better performance than the HI algorithm. The results are depicted in Figures 6B, 6C and 7B, 7C. Obviously, in the ABOD algorithm and SOD algorithm, the outliers appear when the measurements change among its neighbor nodes. Moreover, by combining the advantages of the angle-based method and the distance-based method, our proposed SOD algorithm overcomes the limitations of these algorithms (i.e., being bounded in high-dimensional data, or a dataset having both dense and sparse dimensions). Hence, as can be seen in Figures 6C and 7C, the number of CP points in our algorithm is higher than that in the other two algorithms, and the SOD algorithm can work well with high-dimensional data. However, there are still some

**TABLE 6** Event areas in the network

| Event area | Source of event | Occurring time (time in second) | |
|---|---|---|---|
| | | From | To |
| 1. Temperature events | | | |
| $\{S_{33}, S_{34}, S_{35}, S_{37}\}$ | $S_{34}$ | 08:54:20 | 09:56:20 |
| $\{S_{34}, S_{35}, S_{37}\}$ | $S_{34}$ | 10:07:37 | 14:14:05 |
| $\{S_1, S_2, S_3, \}$ | $S_1$ | 08:22:07 | 09:12:00 |
| $\{S_1, S_{33}\}$ | $S_{33}$ | 09:47:27 | 14:14:06 |
| $\{S_2, S_{33}\}$ | $S_{33}$ | 09:45:24 | 11:42:20 |
| $\{S_1, S_3\}$ | $S_3$ | 14:45:47 | 15:09:07 |
| 2. Humidity events | | | |
| $\{S_1, S_2, S_3, S_{33}, S_{34}, S_{35}, S_{37}\}$ | $S_{34}$ | 07:49:16 | 09:22:36 |
| $\{S_1, S_2, S_3, S_{33}, S_{35}, S_{37}\}$ | $S_{33}$ | 09:22:37 | 11:27:25 |
| $\{S_1, S_3, S_{33}, S_{35}, S_{37}\}$ | $S_{33}$ | 11:27:26 | 16:35:45 |

**TABLE 7** AUC scores of ABOD and SOD algorithms on IBRL dataset

| Outlier ratio (%) | ABOD | SOD |
|---|---|---|
| 5 | 0.873 ± 0.0128 | 0.862 ± 0.0235 |
| 25 | 0.915 ± 0.0261 | 0.918 ± 0.0156 |
| 50 | 0.926 ± 0.0048 | 0.930 ± 0.0076 |
| 75 | 0.958 ± 0.0170 | 0.942 ± 0.0217 |
| 90 | 0.957 ± 0.0239 | 0.951 ± 0.0173 |

MP points in both ABOD and SOD algorithms. In this case, we should use both TOD and SOD algorithms to improve the accuracy identification rate. These simulation results indicate the robustness of the methods in the presence of spatial data. The temporal outlier detection methods can similarly be used in many applications that require identification of outliers in the data across time (e.g., abrupt changes in the stock market, anomalies in customer transactions, and network violations). These methods are very useful for detecting early anomalies in streaming data. Unfortunately, the temporal outlier detection methods do not take spatial sequences into account, and therefore, they cannot identify pattern changes over space. In contrast, the spatial outlier detection methods detect anomalies in the data of sensor nodes, whose values are significantly different from those of its spatial neighbor nodes. These methods are widely used in localization and tracking applications. However, the temporal behaviors are not captured in spatial outlier detection methods. Moreover, the accuracy identification rate of spatial outlier detection methods is sensitive to environmental noises. Therefore, we should choose the best suitable method depending on the specific data type and the requirements of the applications.

### 5.3.4 | Evaluation through ROC analysis

This section describes the use of the receiver operating characteristic (ROC) [34] analysis to evaluate the performance of the algorithms. In this analysis, the performance metric is the area under the ROC curve (AUC). Therefore, the larger the AUC is, the more accurate the method is. We evaluated our proposed SOD algorithm on the IBRL dataset. While using the IBRL dataset, we found that the original data files did not provide any information about outliers. Therefore, similar to the research by [53,54], we generated a "labeled dataset" from the IBRL dataset by the two following steps: (a) use the SOD algorithm to detect all outliers and then replace them by their estimated values (normal values), (b) generate some outliers, whose values fall outside the normal data range and then inject them into the dataset in a uniform distribution. It should be noted that the labeled dataset in this case is defined as a dataset in which some labels have been tagged (e.g., the size, values, and positions of outliers). The size of the dataset is fixed, and we vary the number of outliers from 5% to 90% of the total data points. We again compare the outcome of our algorithm with that of the ABOD algorithm for ranking the accuracy identification rate. The simulation results are given in Table 7. It can be seen that the AUC of the SOD algorithm is quite similar to that of the ABOD algorithm. The best values of AUC obtained by the algorithms vary from 0.862 to 0.957. It also illustrates that the AUC of the algorithms increases with the outlier ratios. This means that with higher number of outliers in the dataset, the number of CP points is likely to increase.

## 6 | CONCLUSIONS

We have introduced a new threshold-based real-time outlier detection algorithm. The threshold parameter was optimized according to the Neyman-Pearson lemma. The new method achieves real-time outlier detection because of its algorithmic simplicity. We performed extensive numerical analyses, which proved that the proposed method provides better statistical results for outlier detection than the typical method (HI algorithm). Our algorithms are suitable for online outlier detection in any applications that require accurate analysis and real-time decision-making. Our algorithms achieve high performance with a low-energy consumption and small storage requirement. In our future work, we are planning to conduct more research on the early detection of anomalies in healthcare monitoring applications.

## ORCID

*Hoc Thai Nguyen* https://orcid.org/0000-0002-9617-3605
*Nguyen Huu Thai* https://orcid.org/0000-0002-0025-1884

## REFERENCES


1. G. Treplan, L. Tran-Thanh, and J. Levendovszky, *Energy efficient reliable cooperative multipath routing in wireless sensor networks*, World Acad. Sci. Eng. Technol. **68** (2010), 1366–1371.
2. J. Levendovszky et al., *Fading-aware reliable and energy efficient routing in wireless sensor networks*, Comput. Commun. **33** (2010), S102–S109.
3. M. Carlos-Mancilla, E. López-Mellado, and M. Siller, *Wireless sensor networks formation: approaches and techniques*, J. Sensors **2016** (2016), 1–18.
4. A. Saifullah et al., *Enabling reliable, asynchronous, and bidirectional communication in sensor networks over white spaces*, in Proc. ACM Conf. Embedded Netw. Sens. Syst., Delft, Netherlands, Nov. 2017, pp. 9:1–14.
5. H. Jawad et al., *Energy-efficient wireless sensor networks for precision agriculture: a review*, Sensors **17** (2017), no. 8, 1781:1–45.
6. W. Li et al., *Defective sensor identification for wsns involving generic local outlier detection tests*, IEEE Trans. Signal Inf. Process. Netw. **2** (2016), no. 1, 29–48.
7. Z. Feng et al., *A new approach of anomaly detection in wireless sensor networks using support vector data description*, Int. J. Distrib. Sens. Netw. **13** (2017), no. 1, 155014771668616.
8. A. De Paola et al., *Adaptive distributed outlier detection for wsns*, IEEE Trans. Cybern. **45** (2015), no. 5, 902–913.
9. L. Martí et al., *Anomaly detection based on sensor data in petroleum industry applications*, Sensors **15** (2015), no. 2, 2774–2797.
10. X. Liu et al., *Fault tolerant complex event detection in wsns: a case study in structural health monitoring*, IEEE Trans. Mob. Comput. **14** (2015), no. 12, 2502–2515.
11. K. Singh and S. Upadhyaya, *Outlier detection: applications and techniques*, Int. J. Comp. Sci. Issue (IJCSI) **9** (2012), no. 1, 307.
12. C. O'Reilly et al., *Anomaly detection in wireless sensor networks in a non-stationary environment*, IEEE Commun. Surveys Tutorials **16** (2014), no. 3, 1413–1432.
13. H. Aguinis, W. F. Cascio, and R. S. Ramani, *Science's reproducibility and replicability crisis: International business is not immune*, J. Int. Bus. Stud. **48** (2017), 653–663.
14. A. Abid, A. Kachouri, and A. Mahfoudhi, *Outlier detection for wireless sensor networks using density-based clustering approach*, IET Wirel. Sensor Syst. **7** (2017), no. 4, 83–90.
15. F. R. Hampel, *A general qualitative definition of robustness*, Ann. Math. Stat. **42** (1971), 1887–1896.
16. H. Liu, S. Shah, and W. Jiang, *On-line outlier detection and data cleaning*, Comput. Chem. Eng. **28** (2004), no. 9, 1635–1647.
17. H. H. W. J. Bosman et al., *Spatial anomaly detection in sensor networks using neighborhood information*, Inf. Fusion **33** (2017), 41–56.
18. R. K. Pearson, *Outliers in process modeling and identification*, IEEE Trans. Control Syst. Technol. **10** (2002), no. 1, 55–63.
19. G. Zheng et al., *Contextual spatial outlier detection with metric learning*, in Proc. ACM SIGKDD Int. Conf. Knowledge Discovery Data Mining, Halifax, Canada, Aug. 2017, pp. 2161–2170.
20. I. Ben-Gal, Outlier detection, *Data mining and knowledge discovery handbook*, Springer, New York, USA, 2005, pp. 131–146.
21. A. M. Said, P. D. D. Dominic, and I. Faye, *Data stream outlier detection approach based on frequent pattern mining technique*, Int. J. Bus. Inf. Syst. **20** (2015), no. 1, 55–70.
22. M. S. Uddin et al., *Online bad data detection using kernel density estimation*, in IEEE Power Energy Soc. General Meeting, Denver, CO, USA, July 2015, pp. 1–5.
23. M. Solaimani et al., *Statistical technique for online anomaly detection using spark over heterogeneous data from multi-source vmware performance data*, in IEEE Int. Conf. Big Data (Big Data), Washington, DC, USA, Oct. 2014, pp. 1086–1094.
24. C. C. Aggarwal and P. S. Yu, *Outlier detection for high dimensional data*, ACM SIGMOD Record **30** (2001), 37–46.
25. M. M. Breunig et al., *Lof: identifying density-based local outliers*, ACM SIGMOD Record **29** (2000), 93–104.
26. E. M. Knox and R. T. Ng, *Algorithms for mining distancebased outliers in large datasets*, in Proc. Int. Conf. Very Large Data Bases, New York, USA, 1998, pp. 392–403.
27. L. Tran, L. Fan, and C. Shahabi, *Distance-based outlier detection in data streams*, Proc. VLDB Endowment **9** (2016), no. 12, 1089–1100.
28. A. Christy, G. Meera Gandhi, and S. Vaithyasubramanian, *Cluster based outlier detection algorithm for healthcare data*, Procedia Comput. Sci. **50** (2015), 209–215.
29. S. Ramaswamy, R. Rastogi, and K. Shim, *Efficient algorithms for mining outliers from large data sets*, ACM SIGMOD Record **29** (2000), no. 2, 427–438.
30. H.-P. Kriegel et al., *Angle-based outlier detection in high-dimensional data*, in Proc. ACM SIGKDD Int. Conf. Knowledge Discovery Data Mining, Las Vegas, NV, USA, Aug. 2008, pp. 444–452.
31. J. Han, J. Pei, and M. Kamber, *Data mining: concepts and techniques*, 3rd ed., Morgan Kaufmann, Elsevier, Waltham, MA, 2011.
32. X. Li, J. Lv, and Z. Yi, *An efficient representation-based method for boundary point and outlier detection*, IEEE Trans. Neural Netw. Learn. Syst. **29** (2018), no. 1, 51–62.
33. W. Ruan et al., *Tagfall: Towards unobstructive fine-grained fall detection based on uhf passive rfid tags*, in Proc. EAI Int. Conf. Mobile Ubiquitous Syst.: Comput., Netw. Services, Coimbra, Portugal, July 2015, pp. 140–149.
34. T. Fawcett, *An introduction to roc analysis*, Pattern Recogn. Lett. **27** (2006), no. 8, 861–874.
35. S. Haque, M. Rahman, and S. Aziz, *Sensor anomaly detection in wireless sensor networks for healthcare*, Sensors **15** (2015), no. 4, 8764–8786.
36. B. Krishnamachari and S. Iyengar, *Distributed bayesian algorithms for fault-tolerant event region detection in wireless sensor networks*, IEEE Trans. Comput. **53** (2004), no. 3, 241–250.
37. W. Weili et al., *Localized outlying and boundary data detection in sensor networks*, IEEE Trans. Knowl. Data Eng. **19** (2007), no. 8, 1145–1157.
38. H. T. Nguyen et al., *Efficient approach for maximizing lifespan in wireless sensor networks by using mobile sinks*, ETRI J. **39** (2017), no. 3, 353–363.
39. R. K. Pearson, *Mining imperfect data: dealing with contamination and incomplete records*, vol. 93, SIAM, Philadelphia, PA, USA, 2005.
40. E. Lengyel, *Mathematics for 3D game programming and computer graphics*, Cengage Le arning, Boston, MA, USA, 2012.
41. H. T. Nguyen and L. János, *Position location technique in non-line-of-sight environments for wireless sensor networks*, J. Comput. Sci. Cybern. **32** (2016), no. 2, 93–111.

42. C. E. Rasmussen, Gaussian processes in machine learning, *Summer School on Machine Learning*, Springer, 2003, pp. 63–71.
43. M. Kemmler et al., *One-class classification with Gaussian processes*, Pattern Recogn. **46** (2013), no. 12, 3507–3518.
44. R. Ramirez-Padron, B. Mederos, and A. J. Gonzalez, *Novelty detection using sparse online gaussian processes for visual object recognition*, in Int. FLAIRS Conf., St. Pete Beach, FL, USA, May 2013, pp. 124–129.
45. M. Smith et al., *Maritime abnormality detection using Gaussian processes*, Knowl. Inf. Syst. **38** (2014), no. 3, 717–741.
46. J. Neyman and E. S. Pearson, *On the problem of the most efficient tests of statistical hypotheses*, Philos. Trans. R. Soc. London. Series A **231** (1933), 289–337.
47. L. Davies and U. Gather, *The identification of multiple outliers*, J. Am. Stat. Assoc. **88** (1993), no. 423, 782–792.
48. J. López and M. J. López,Package `tsoutliers' (2017).
49. L. Grossi and M. Riani, Robust time series analysis through the forward search, *Compstat*, Springer, 2002, pp. 521–526.
50. R. K. Pearson et al., *Generalized hampel filters*, EURASIP J. Adv. Signal Proc. **2016** (2016), no. 1, 87:1–18.
51. M. M. Smadi and F. S. Mjalli, *Forecasting air temperatures using time series models and neural-based algorithms*, J. Math. Stat. **3** (2007), 44–48.
52. C. Guestrin et al., *Intel lab data*, J. Math. Stat. (2016), http://db.lcs.mit.edu/labdata/labdata.html.
53. T. Zhang et al., *Bayesian-optimization-based peak searching algorithm for clustering in wireless sensor networks*, J. Sensor Actuator Netw. **7** (2018), no. 1, 2:1–19.
54. S. Siripanadorn et al., *Anomaly detection using self-organizing map and wavelets in wireless sensor networks*, in Proc. WSEAS Int. Conf. Appl. Comput. Sci., Japan, Oct. 2010, pp. 291–297.

## AUTHOR BIOGRAPHIES

**Hoc Thai Nguyen** received the BEng and MEng degrees in electrical engineering and automation from the Vietnam National University of Agriculture, Hanoi, Vietnam, in 2006 and 2010, respectively, and the PhD degree in electrical engineering and informatics from Budapest University of Technology and Economics, Budapest, Hungary, in 2018. Since 2007, he has been working at the Department of Automation, Vietnam National University of Agriculture, Hanoi, Vietnam, where he has developed various smart and precision agriculture systems. He is currently involved in the development of wireless sensor networks, mobile sensor networks and robotics, Gaussian processes, machine learning, and artificial neural networks. He can be contacted at: Mail: thaihocme@gmail.com; nguyenthaihoc@vnua.edu.vn.

**Nguyen Huu Thai** received the BEng degree in electrification and power supply and the MEng degree in automation engineering from the Thainguyen University of Technology, Thainguyen, Vietnam in 2001 and 2005, respectively, and the PhD degree in control engineering and automation from the Hanoi University of Science and Technology, Hanoi, Vietnam, in 2015. Since 1996, he has been working at the Faculty of Electrical Engineering, Vinh University of Technology Education. His main researches include nonlinear adaptive control, fuzzy systems, neural networks, wireless sensor networks, mobile sensor networks, and robotics. He can be contacted at: Mail:thainguyenktv@gmail.com.