SENSELET++: A Low-cost Internet of Things Sensing Platform for Academic Cleanrooms

Beitong Tian[1], Zhe Yang[1], Hessam Moeini[1], Ragini Gupta[1], Patrick Su[2], Robert Kaufman[2], Mark McCollum[2], John Dallesasse[2], Klara Nahrstedt[1]
[1]*Coordinated Science Laboratory,* [2]*Holonyak Micro & Nanotechnology Laboratory*
University of Illinois at Urbana-Champaign, Champaign, USA
{*beitong2,zheyang3,moeini,raginig2,psu8,rbkaufm2,markjmcc,jdallesa,klara*}*@illinois.edu*

***Abstract*—Sensory IoT (Internet of Things) networks are widely applied and studied in recent years and have demonstrated their unique benefits in various areas. In this paper, we bring the sensor network to an application scenario that has rarely been studied - the academic cleanrooms. We design SENSELET++, a low-cost IoT sensing platform that can collect, manage and analyze a large amount of sensory data from heterogeneous sensors. Furthermore, we design a novel hybrid anomaly detection framework which can detect both time-critical and complex non-critical anomalies. We validate SENSELET++ through the deployment of the sensing platform in a lithography cleanroom. Our results show the scalability, flexibility, and reliability properties of the system design. Also, using real-world sensory data collected by SENSELET++, our system can analyze data streams in real-time and detect shape and trend anomalies with a 91% true positive rate.**

***Index Terms*—Internet of Things; Sensor Network; Anomaly Detection**

I. Introduction

Internet of Things (IoT) is made up of various devices embedded with sensors, actuators and software which have the ability to connect to each other or to the Internet. By combining IoT devices together with an automated system, we can build an IoT system to automatically collect and analyze information and create outputs for different given tasks. Many smart IoT systems are designed and deployed in different areas such as agriculture, healthcare and building automation for the purpose of monitoring and control process [1]–[3]. However, there are only a few IoT works (e.g., [4]) that focus on embedding IoT systems into **academic cleanroom laboratories** to enable a sensing platform that would provide scalable, evolvable, secure and safe capabilities in a dynamic cleanroom environment.

Generally, cleanroom labs are critical environments including a variety of scientific instruments for semiconductor fabrication and manufacturing of chips and are characterized by various physical control factors such as temperature, humidity, airflow, and airborne particles to ensure seamless operation of high-end instruments. When compared to industrial cleanrooms, academic cleanrooms have unique challenges and requirements: (a) highly diverse research tasks conducted by researchers, (b) highly diverse users with knowledge expertise ranging from novice students to experienced faculty and lab managers, (c) highly diverse scientific equipment ranging from microscopes such as Scanning Electronic Microscope (SEM) to fume-hoods and pumps, (d) highly diverse missions ranging from teaching undergraduate students to operating scientific instruments in graduate research to develop new chips, (e) diverse usage of equipment and machinery in terms of their frequency and setups since students come and go and set instruments into different states as they conduct experiments, and (f) constrained budget to maintain and repair these cleanroom instruments.

Furthermore, it is challenging to design IoT systems for academic cleanrooms that have the following requirements:

1. Environmental monitoring: Any violation of strictly controlled environmental parameters such as particle density and humidity may cause experiment failures. Using the semiconductor experiment as an example, under a high particle density environment, dust particles are more likely to fall on the silicon wafer and fail the whole semiconductor manufacturing process. Besides the particle density, excess humidity will cause the photoresist process for semiconductors to behave unexpectedly which is highly undesirable as scientific results may become invalid.

2. Instrument monitoring: There are diverse scientific instruments in the academic cleanroom, including core instruments such as the lithography system and auxiliary equipment such as pumps in water cooling systems. An auxiliary equipment failure may cause a chain effect and result in a core instrument failure and a significant repairing expense. A pump failure may lead to a water cooling system's failure and cause expensive instruments to overheat. By monitoring via sensors (IoT devices) some parameters of the mechanical equipment such as surface temperature, we can determine if the equipment such as a pump is in a healthy state. The sensory data can help us predict the failure of instruments and replace the unhealthy instruments ahead to avoid cascading failures. Additionally, we can monitor safety-critical instruments such as fume-hood and HVAC systems to make sure that toxic gas does not undermine the safety of researchers.

3. Security monitoring: We need to monitor (a) safety in cleanrooms since students work with chemicals, (b) reliability, integrity and availability of measurements since scientific

This research was funded by the NSF (award number 1827126). The opinions, findings and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the view of the NSF.

instruments may interfere with sensing infrastructure, and (c) entrances of cleanrooms to detect violated access to cleanrooms for regulating the access control and protecting the property in cleanrooms. Since privacy is not a first-class object in academic cleanrooms, we only consider security, safety, reliability, and availability of sensory data in this work.

Based on these requirements, we aim to address the following challenges in academic cleanrooms:

- **Scale and Heterogeneity:** Due to the large scale and heterogeneity of sensors for various environmental and instrument monitoring tasks, we need to (a) design new integrative hardware methods to connect a large number of commodity sensors with different interfaces and output formats to edge devices in a highly reliable manner and (b) solve the power problem.
- **Flexibility and Evolvability:** The environmental and instrument monitoring requirement will keep changing and updating due to dynamic changes of cleanrooms' settings, which requires our sensory platform to evolve and expand with new kinds of sensors and algorithms or replace sensors and change the sensor layout.
- **Availability and Reliability:** High power machinery in cleanrooms will interfere with the nearby wireless environment violating instrument and environmental monitoring requirements. For instance, interference will block the WiFi signal and alerts will not be sent out in time. We need to provide availability guarantees of the sensory data stream. The IoT system should be designed in a reliable way to extend the life-time.
- **Effective and Efficient Anomaly Detection:** We need to carefully design an anomaly detection framework to satisfy security monitoring requirements and to detect meaningful anomalies from a large amount of sensory data streams with the consideration of performance and latency.

To address the above challenges, we present SENSELET++, an end-to-end low-cost and real-time IoT sensing platform for smart cleanrooms with characteristics of scalability, flexibility, reliability, availability, and integrity. To enable *scalability*, we design a new set of interfaces and connectors for sensors and edge devices based on 1-Wire protocol [5]. The new design allows us to connect and power tens of heterogeneous sensors to one edge device in a *reliable* way. We design hardware and software of SENSELET++ in a highly *modular* fashion so that we can easily add new kinds of sensors and algorithms without large modification of the existing system, providing flexibility and evolvability of the platform. Furthermore, we carefully design the control software module running in the edge device to realize the *plug-and-play* feature to increase the *flexibility*. We separate the wireless part and sensing part so that we can place the wireless part in a low interference area of the cleanroom to secure *availability*.

We design two paths for achieving *effective* and *efficient* anomaly detection. The quick path will detect critical anomalies i.e., abnormal measurements (e.g., abnormal critical measurements around water pumps, unusual fume-hood temperature increase) with rule-based algorithms and run on edge devices to minimize the latency between the generation of anomalies and sending out alerts. The slow path will detect non-time-critical anomalies where latency is not the first-class object such as micro-climate anomalies and run on the cloud server. We design a Singular Spectrum Analysis (SSA) [6] based anomaly detection framework to extract meaningful information from a large amount of sensory data. We make the following contributions:

1) **SENSELET++ Design:** We present design of SENSELET++, a low-cost, real-time, inclusive, evolvable (flexible), scalable, and secure IoT system for academic cleanrooms. The design includes a unique hardware-software co-design architecture to increase the IoT system's scalability, reliability, availability, and extensibility. We carefully address each of these considerations in our design and to the best of our knowledge, this is the first work towards the design and implementation of a scalable end-to-end IoT sensing and monitoring platform for academic cleanroom environments.
2) **Anomaly Detection Framework:** We design an online light-weight anomaly detection framework which can automatically detect abnormal events of changes in cleanroom's micro-climate, safety and security from heterogeneous sensory data streams. Advanced anomaly detection techniques are applied to real-time environmental and contextual sensor data to identify sensor faults/variations and environmental property fluctuations which significantly help researchers and lab managers in academic cleanrooms.
3) **Implementation and Evaluation:** We validate SENSELET++ in a semiconductor cleanroom. During the past several months, we have extensively evaluated our system and showed that the system meets our design goals. The system has generated valuable information from the collected data which helps the cleanroom administrator and researchers to have new, useful findings of the cleanroom.

The paper is organized as follows. Section II surveys the related work. Section III provides an overview of SENSELET++ with its a) system architecture and data flow, and b) system design responding to existing challenges in the academic cleanrooms. Section IV describes our anomaly detection algorithms that provide integrity and improve performance in SENSELET++. Section V presents the experimental results from the deployed SENSELET++ in a real academic cleanroom. The paper concludes in Section VI.

II. Related Work

In order to understand the need for an IoT infrastructure in scientific cleanrooms, we explored the existing IoT sensing solutions and their challenges in critical environments, current sensory data acquisition systems, and sensor data analytics in a networked system, discussed in detail as follows:

A. Indoor environmental monitoring in critical environments

Several previous works [7]–[10] have been steered towards application-centric design of indoor air monitoring systems

and CO2 sensors in cleanrooms for industries such as pharmaceuticals, semiconductors, nuclear waste management and logistics. However, these systems either require reconfiguration in existing ventilation or air-conditioning units of the room or are only limited to inventory and stock management capabilities.

There are several commercial environmental monitoring systems [11], [12] available on the market. These systems can provide basic monitoring functionalities. However, it is hard to add new kinds of sensors and anomaly detection algorithms which are not supported by these systems. Also, their high price is economically infeasible for academic IoT systems.

B. Data acquisition, management and modelling

Different data acquisition systems have been largely deployed in IoT applications such as smart energy [13], smart homes [14], agriculture [15], and airports [16] where sensors are deployed to collect real-time data followed by different processing techniques (such as rule-based approach) to generate periodic notifications and provide device management schedules or surveillance. 4CeeD [17] focused on designing a cloud-based data acquisition system in academic laboratories that facilitates data collection, data sharing and data curation services, and Miras [18] proposed a novel resource management framework for such systems.

C. Data analysis and Anomaly detection

Recent studies [19]–[21] examine different machine learning methods (regression, LSTM, ANN, Isolation forest) on wireless traffic data for anomaly detection used in a variety of applications such as intrusion detection, fraud detection, data leakage, sensor data tampering, link failures, and sensor faults. However, they are restricted to univariate sensor data analysis. In a situation with a more complex and critical environment such as cleanrooms, a thorough analysis and investigation are required to identify anomalous events/variations under different types of sensors (i.e. multi-variate sensor data analysis or sensor data fusion). Moreover, algorithms such as ANN, LSTM and Isolation Forest are very computationally expensive. They are not suitable for a constrained network of edge computing devices. Different statistical models such as ARMA [22] and Bayesian Changepoint [23] have also been explored to address the problem of overfitting in machine learning based anomaly detection algorithms and reduce the false positive alarms efficiently. Another component of this work has been done in contextual-based anomaly detection frameworks [24] for environmental sensors, wherein the algorithm is made contextually aware by using the meta-information (temporal or spatial) associated with data points. While the contextually aware algorithm was scalable and adaptable for real-time detection, it required extensive sensor profiling with a large amount of historical values collected from the same sensors.

III. SENSELET++

In this section, we will discuss hardware and network architecture in SENSELET++ (Sec. III-A) and introduce our system design overcoming the existing challenges for building an IoT sensing platform in academic cleanrooms (Sec. III-B).

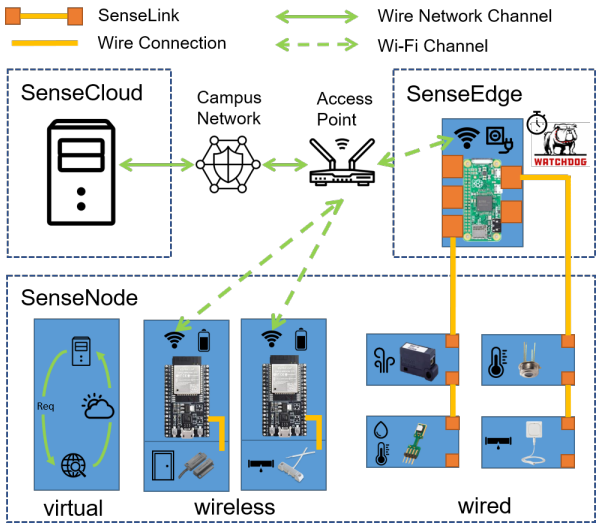

Fig. 1. SENSELET++ hardware and network architecture

A. System Architecture and Data Flow

Main components of SENSELET++ and their relations in the whole architecture are illustrated in Figure 1.

SenseNode is a device used to generate single or multiple sensory data streams. We introduce three types of SenseNode in our design to support various sensing tasks. *Wired* SenseNodes are sensors used to collect various physical parameters in the cleanrooms. These sensors are directly connected to the SenseEdge by wires and cables. *Wireless* SenseNodes are sensors directly connected to the SenseCloud through Wi-Fi access points and are mainly used to detect push-based events such as water leakage or door status change in places far from power sockets. Finally, *virtual* SenseNode is a running service in SenseCloud or SenseEdge layer used to collect two categories of data: 1) public data such as weather data from public datasets, and 2) network related statistics such as bandwidth usage of a SenseEdge.

SenseEdge is the controller and a gateway for the wired SenseNodes connected to it. It makes it possible for the low-priced sensors to efficiently transmit their sensory data to the SenseCloud. Moreover, SenseEdge runs a watchdog service that handles various failures and supports the system's reliability.

SenseCloud is a server for collecting, storing and analyzing sensory data. All components of SENSELET++ are protected from external attacks by the campus network firewalls. Wireless SenseNodes and SenseEdges connect to the campus network via nearby Wi-Fi access points while SenseCloud uses Ethernet to connect to this network.

Figure 2 shows data flows in SENSELET++ and illustrates how sensory data collected from different SenseNodes flows and gets processed in the system before being stored and used at SenseCloud. In Figure 2, the sensory data streams from wired SenseNodes first flows into the SenseEdge. SenseEdge uses a publish-subscribe messaging pattern to transfer data. It then publishes the received data to the Message Queuing Telemetry Transport (MQTT) broker [25] which handles mes-

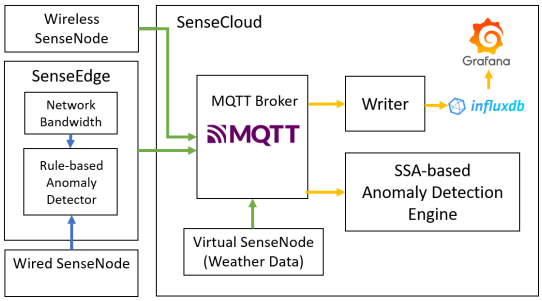

Fig. 2. SENSELET++ data-flow

sage transmission between SenseEdge and SenseCloud. In parallel, a rule-based anomaly detector running on the SenseEdge will also examine the data and send alerts if it detects any anomaly. Wireless and Virtual SenseNodes publish the data to the MQTT broker directly. There exist two running programs on the SenseCloud which subscribe to receive all the messages in the system. The *writer* program is responsible to parse the message and store this parsed message into InfluxDB, which is a high-performance time-series database. Grafana, an open-source data visualization tool, queries the data from InfluxDB and visualizes the data within predefined dashboards. The *anomaly detection engine* also listens to all the published messages and detects anomalies in a timely manner. Next, we will introduce the system design and discuss solutions to handle existing challenges in academic cleanrooms.

B. System Design

We consider different challenges in designing an IoT sensing platform for academic cleanrooms and make sure SENSELET++ is: 1) scalable and flexible, 2) reliable and highly available, and 3) capable of finding anomalies from large-scale sensory data streams with a minimized latency and maximized completeness. We will discuss each of these considerations in detail.

Scalability and Flexibility. We consider both *vertical* and *horizontal* scalability of SENSELET++. Vertical scalability requires the system to easily upgrade the existing components and handle more intensive tasks in the future. We choose Raspberry Pi as the core of SenseEdge because it can easily be replaced and upgraded. The vertical scalability of SenseNodes is demonstrated by the ease of upgrading the sensor with better characteristics such as accuracy or response time. To enable this, we design SenseNodes in a modular way that separates the sensor from other parts of the SenseNode. We then can easily upgrade the sensor when needed.

Horizontal scalability requires that increasing the number of SenseNodes will increase the total costs in a linear or sub-linear function of the number of SenseNodes. This overall cost includes the installation and operational cost of SenseNodes and SenseEdges, bandwidth usage, as well as storage and processing time of SenseEdges and SenseCloud. Due to the lack of a unified and reliable interface needed to connect and power heterogeneous sensors with the edge devices, there exists a scalability bottleneck at the sensor and edge layer. We

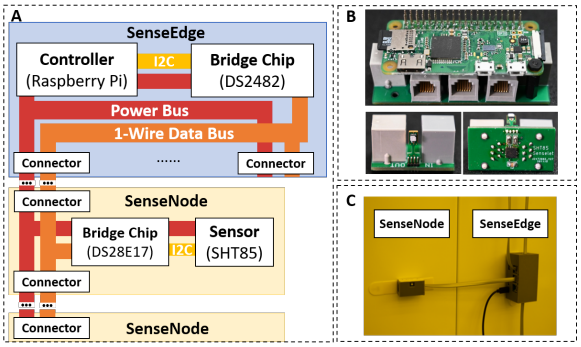

Fig. 3. The diagram of the interface between a SenseEdge and SenseNodes and implementation results. **A.** Data-flow and power-flow between a SenseEdge and a Temperature & Humidity SenseNode and their components. **B.** Front-view of a SenseEdge (Top). Front-view of a SenseNode (Bottom-left). Bottom-view of a SenseNode (Bottom-right) **C.** SenseEdge and SenseNode in production environment.

design a new interface between SenseEdges and SenseNodes as shown in Fig. 3-A to remove this bottleneck.

1) Interconnection interface: In the academic cleanroom, tens of sensors need to be connected to a single edge device to save space and reduce the cost. However, scalability is an issue because the number of interfaces on each edge device limits the number of connected sensors. We design a new interface between the SenseEdge and SenseNodes to provide scalability. The idea is to convert various output types[1] into a uniform one.

We use 1-Wire protocol, a serial protocol with features known as long-range covering (fits our academic cleanroom size: 5m x 10m), simplicity, and low data rate requirements. As shown in Fig. 3-A, each SenseNode has a dedicated bridge chip to convert the sensor's output type into the uniform 1-Wire protocol. The converted 1-Wire data stream is then transmitted to the SenseEdge via the 1-Wire data bus. On the SenseEdge, a bridge chip will convert the 1-Wire data stream back to I2C stream for the reading of Raspberry Pi.

The *power bus* provides 3.3V and 5V power to SenseNodes from the SenseEdge which itself is powered by abundant sockets on the cleanroom walls. Connectors are used to connect SenseEdge and SenseNodes. We use the modular cable to connect two connectors as shown in Fig.3-C. Our new interface allows us to connect heterogeneous sensors to any connectors on the SenseEdge and chain sensors together which increases the scalability and supports different types of sensor network topologies.

2) Plug-and-Play design: Most cheap commodity sensors are not automatically detectable and identifiable at the edge level. Hence, people need to plan ahead what sensors each edge device needs to connect and hard code the software accordingly. This static design forces users to rearrange connected sensors and redesign the software when a system adjustment is needed, which is laborious and undesirable. The plug-and-play design enables us to easily add, remove and replace SenseEdges and SenseNodes without any rearrangement

[1]Some sensors output analog voltage. Some sensors support serial communication buses such as I2C or SPI. Here we collectively call them the output type.

of the infrastructure. To make SenseNodes plug-and-play, the SenseEdge needs to (1) detect the plug and unplug events of SenseNodes; (2) identify the metadata of SenseNodes such as the output type, the location and sampling rate; and (3) use the dedicated API to read in and process the sensor data.

The 1-Wire bridge chip used in our new design can provide a unique ID for each SenseNode. We collect the information of currently connected sensors continuously from a folder in the SenseEdge operating system where the 1-Wire bus driver lists all the connected SenseNode's ID. When a new SenseNode plugs in, SenseEdge uses its ID to query the metadata database (which has the metadata for all the SenseNodes we manufactured). This metadata then will be used by the SenseEdge to update the sensor membership list. A thread will read the data, detect the anomaly, and then publish the data to the SenseCloud for each sensor in the membership list. When the SenseNode is unplugged, it will be removed from the membership list and related resources will be recycled.

Reliability and Availability. We take reliability as a very important goal during the system design because a reliable design can protect the system from various adverse factors which are common in the complex academic cleanroom environment or minimize the cost after the occurrence of failures. We increase the reliability of our system by: (1) protecting the system from possible adverse factors ahead; (2) making the system automatically recovers after failures.

Some of the adverse factors can simply be avoided by improving the system design. *Interference* from instruments is the first one. Some instruments in cleanrooms can cause electronic-magnetic interference resulting in wireless packets loss between SenseEdges and other devices. This communication failure can delay or even lose urgent alert messages which is highly undesirable because it violates safety and security requirements. We solve the problem by measuring the interference level at each location of the cleanrooms and generate a map of interference regions, to move the SenseEdge away from these regions. The wired communication between SenseNodes and the SenseEdge is robust to the interference and SENSELET++ can safely collect data from SenseNodes in high interference areas. However, the design of traditional wireless sensor network infrastructure, where the wireless interface and the sensor of each sensing node are in a whole, cannot easily deal with the interference as our solution does. Another adverse factor that can be avoided is *Hardware faults*. Due to the high-frequency usage of academic cleanrooms, in a long-term deployment, there is a high probability for our system to suffer from hardware faults like sensor or circuit defects due to poor or loose protections. In the event of such hardware faults, we will lose valuable sensor measurements. Thus, to increase the reliability of the hardware, we have designed: (1) circuit boards to support and connect electronic components; (2) connectors to connect devices via cables in a highly reliable way; (3) closures for SenseEdges and SenseNodes to protect sensors and circuit board from water, dust and accidentally touching from users in cleanrooms. The implementation of these designs is illustrated in Fig. 3-B&C.

The above failure prevention methods do not guarantee the reliability and we need an *automatic system recovery* solution to recover devices after the occasional failures happen. Our system can be seen as a distributed system where each SenseEdge is a distributed node. After a node fails, we need to detect the failure, fix the error and restart the node which is a non-trivial process. A watchdog previously introduced in [4] helps SENSELET++ to automatically reboot the device when a failure occurs in the system. A timer will time out and invoke a reboot if some hardware or software failures block the program. This mechanism is found to be effective in the past few months of deployment, and the system recovers from failures autonomously with no human intervention.

IV. Anomaly Detection in SENSELET++

There are different types of *time-critical* and *non-time-critical* sensing events in academic cleanrooms. We first consider time-critical events and introduce a critical anomaly detection pipeline in Sec. IV-A. We then consider non-time-critical events and introduce our SSA-based anomaly detection algorithms in Sec. IV-B to detect anomalies in these events.

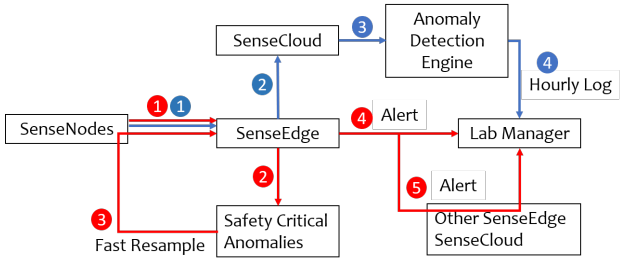

Fig. 4. Anomaly detection data-flow

A. Critical Anomaly Detection

It is challenging to extract useful information with a large amount of data in a real-time manner. We observed critical anomalies are usually simple to be detected, so detecting them does not require advanced algorithms and high computing power, instead, minimizing the latency between the occurrence of the anomaly and sending out the alert is the first priority. Since the sensor reading time is relatively static and cannot be fully optimized, the only opportunity to optimize the latency is in data transmission and data analysis. We add an anomaly detector for critical anomalies on SenseEdges to reduce the delay as shown in Figure 4. The detailed data flow is shown in the *red* path in Figure 4, we call this path the fast path. We keep algorithms used in the SenseEdge as simple as possible to reduce the data processing delay. The algorithms we used are threshold-based algorithms. For example, we will identify the environmental temperature reading as an anomaly if its value is above 30°C. The threshold we used are suggested by domain experts. We also add a fast re-sampling step to avoid the false positive because we observed some sensors rarely (once in a week) give extremely high or low readings. We also proactively broadcast the alert to other devices in our system to increase the success rate of sending out the alert.

B. SSA-based Anomaly Detection

Besides critical anomalies, there are interesting or innovative events happening in academic cleanrooms, which can only be found by more complex algorithms on more powerful servers. However, finding those anomalies are not urgent. We call the data flow to find such anomalies as a slow path as shown in the *blue* path in Figure 4. Those findings will be used for helping lab managers and researchers to have a better understanding of the cleanroom's micro-climate and operation. In order to find this useful information, we proposed an anomaly detection framework based on SSA [6].

SSA Algorithm Description. SSA is a method that can decompose a time series into several meaningful components such as trend, periodicity and noise. Given a time series, where N is the number of samples:

$$F = \{f_1, f_2, ..., f_N\}\ (N \geq 2) \tag{1}$$

The **first step** of SSA is to form a set of lagged column vectors X_i from F and use these vectors to build a trajectory matrix $\mathbf{X}$. Let L be the length of each lagged vector, $2 \leq L \leq N/2$, and $K = N - L + 1$, the total number of lagged vectors. The lagged vectors and the trajectory matrix are:

$$\begin{aligned} X_i &= \{f_i, f_{i+1}, ..., f_{L+i-1}\}^T\ (1 \leq i \leq K) \\ \mathbf{X} &= [X_1, ..., X_K] \end{aligned} \tag{2}$$

The **second step** of SSA is to decompose the trajectory matrix $\mathbf{X}$ with singular value decomposition (SVD). After applying SVD, the trajectory matrix can be written into a combination of d elementary matrices:

$$\mathbf{X} = \mathbf{X}^{(1)} + \mathbf{X}^{(2)} + ... + \mathbf{X}^{(d)} \tag{3}$$

where $\mathbf{X}^{(i)} = \sigma_i U_i V_i^T$, $i = 1, ..., d$, and d is the rank of $\mathbf{X}$. The collection of $\{\sigma_i, U_i, V_i\}$ is called the ith eigentriple of the SVD, where σ_i is the singular value indicating the importance of this eigentriple and U_i and V_i are corresponding left and right singular vectors.

The **third step** is to use diagonal averaging to reconstruct a time series component $\tilde{F}_i$ from elementary matrix $\mathbf{X}^{(i)}$, where:

$$\tilde{F}_i(n) = \begin{cases} \frac{1}{n}\sum_{j=1}^{n} \mathbf{X}^{(i)}_{j,n-j+1} & \text{if } 1 \leq \text{n} < \text{L} \\ \frac{1}{L}\sum_{j=1}^{L} \mathbf{X}^{(i)}_{j,n-j+1} & \text{if L} \leq \text{n} \leq \text{K} \\ \frac{1}{N-n+1}\sum_{j=n-K+1}^{N-K+1} \mathbf{X}^{(i)}_{j,n-j+1} & \text{if K} < \text{n} \leq \text{N} \end{cases} \tag{4}$$

$\mathbf{X}^{(i)}_{a,b}$ means the element at row a and column b of elementary matrix $\mathbf{X}^{(i)}$.

According to the linear nature of diagonal averaging and math deduction:

$$F = \tilde{F}_1 + \tilde{F}_2 + ... + \tilde{F}_d \tag{5}$$

In the **last step**, we calculate the w-correlation [26] and automatically group $\tilde{F}_i$s into trend, periodicity and noise components based on the correlation factor. After this step, the original time series F can be written as:

$$F \approx \tilde{F}^{(Trend)} + \tilde{F}^{(Periodicity)} + \tilde{F}^{(Noise)} \tag{6}$$

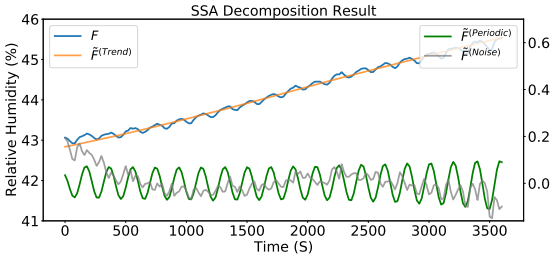

Fig. 5. SSATriple example of a 1-hour segment of humidity time series

Online Anomaly Detection Framework. To make the SSA algorithm work on data streams, we segment each sensor stream with a sliding window with the width of W and the step size of S and get the segment of data $F_{s,w}$, where s is the index of the sensor and w is the index of the sliding window. We apply SSA algorithm on $F_{s,w}$ to get a SSATriple $SSA_{s,w} = \{\tilde{F}^{(Trend)}_{s,w}, \tilde{F}^{(Periodicity)}_{s,w}, \tilde{F}^{(Noise)}_{s,w}\}$, which is the building block for our anomaly detection framework. As an example, Fig. 5 shows the SSATriple of a segmented data stream from a humidity sensor.

The idea of our anomaly detection algorithm is to find how similar a target SSATriple, $SSA^{target}_{s,w}$ is to one or several reference SSATriples, which represent the past or averaged behavior. If they differ to some degree, we can conclude there is an anomaly. Following this idea, the immediate problem to be solved is: *How to choose the reference SSATriples?*

Intuitively, we want to compare the current reading with the past reading. So, we choose SSATriple calculated in the last sliding window as the reference SSATriple, denoted as $SSA^{temporal}_{s,w}$. After we find proper reference SSATriples, the second problem is: *How to compare the target SSATriple with reference SSATriples and find various anomalies?* We notice there is not a one-size-fits-all solution for this question because comparing different components of an SSATriple or using different comparison methods will lead to different types of anomalies. Hence, we categorize anomalies into 2 groups which are most important to cleanrooms. Below we describe each of these two types of anomaly:

Fig. 6. Two kinds of anomaly we are interested in: **(A)** Short-Period Shape Anomaly; **(B)** Long-Period Shape Anomaly; **(C)** Trend Anomaly;

From the collected data, we can find two kinds of **Shape Anomaly** as shown in Fig.6-A&B. The shape anomaly usually has a short duration (**A**) and is similar to noise. The sensor reading will come back to normal after the anomaly disappears. Sometimes we can observe long-period shape anomaly (**B**) which has a unique pattern and are highly desired to be detected. The trend of the sensor reading can change dramatically as depicted in Fig.6-C. We name it **Trend Anomaly**.

We calculate shape and trend anomaly scores every time when the sliding window slides one step ahead. Given the target SSATriple of the target time series:

$$SSA^{target}_{s,w} = \{\tilde{F}^{(Trend)}_{s,w}, \tilde{F}^{(Periodicity)}_{s,w}, \tilde{F}^{(Noise)}_{s,w}\} \tag{7}$$

and its temporal correlated SSATriple:

$$SSA_{s,w}^{temporal} = \{\tilde{F}_{s,w-1}^{(Trend)}, \tilde{F}_{s,w-1}^{(Periodicity)}, \tilde{F}_{s,w-1}^{(Noise)}\} \quad (8)$$

The shape anomaly score is the norm of the last S values of the noise component of the target time series, where S is the step size of the sliding window. From the experiment result, the shape anomaly score is good at detecting short-term shape anomalies.

$$AS_{Shape} = \left\| \tilde{F}_{s,w}^{(Noise)}[(W-S):W] \right\|_2 \quad (9)$$

To detect long-period shape anomalies and trend anomalies, we calculate the trend anomaly score by normalizing the Euclidean distance between the trend components of two SSATriples.

$$AS_{Trend} = \frac{1}{Mean(\tilde{F}_{s,w}^{(Trend)})} \left\| \tilde{F}_{s,w}^{(Trend)} - \tilde{F}_{s,w-1}^{(Trend)} \right\|_2 \quad (10)$$

An anomaly is identified if the anomaly score is above a predefined threshold.

V. Experimental Validation

In this section, we discuss the results of experiments we have conducted to validate SENSELET++ and verify the effectiveness of our design. We first introduce the test-beds we built to perform experiments. To evaluate SENSELET++ in a real environment, we implement and deploy it in an academic semiconductor lithography cleanroom in Holonyak Micro and Nanotechnology Laboratory (HMNTL) at the University of Illinois at Urbana-Champaign. We deployed 16 SenseNodes and 4 SenseEdges in this cleanroom to: 1) test the system reliability (V-B), 2) check the latency and effectiveness of critical anomaly detection module (V-C), and 3) collect data for evaluating our SSA based anomaly detection framework (V-D). Table I lists hardware details for all the devices placed in the cleanroom. The SenseEdge uses *Raspberry Pi Zero W*, which has a 1GHz, single-core CPU and 512MB RAM and runs a Debian operating system. The SenseCloud in this setting is a desktop computer with Intel(R) Core(TM) i7-2600 CPU @ 3.40GHz and 4 GB memory running Ubuntu 16.04. Developed programs in the SenseEdge and SenseCloud are written in Python and the network environment used by this test-bed is the UIUC campus network. Part of the deployment is shown in Fig. 7.

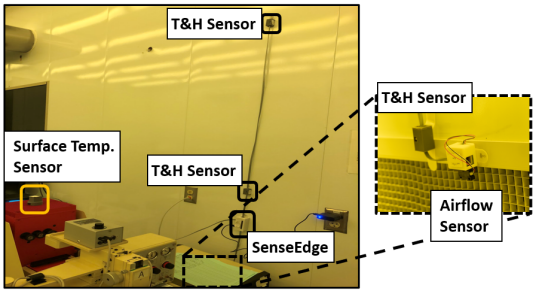

Fig. 7. Part of the SENSELET++ deployment. Three temperature and humidity (T&H) SenseNodes, one surface temperature SenseNode, and one airflow SenseNode are connected to a single SenseEdge device.

The second test-bed is a home-based setup to: 1) test the scalability of SenseEdge (V-A), and 2) evaluate the performance of the SSA-based anomaly detection (V-D). The SenseCloud in this test-bed uses AMD Ryzen 5 3600 6-Core Processor @ 3.60GHz and has 16 GB memory.

A. Scalability of SenseEdge

We connect a different number of I2C-based SenseNodes (from 1 to 10) to a SenseEdge and monitor changes in CPU and bandwidth usages to verify its scalability. We record the metrics of each setting every second and for a period of one minute, with a 0.5 Hz sampling rate and the result is shown in Fig. 8. We use I2C based SenseNodes here because they are more resource-consuming than other types of SenseNodes. The result suggests that the CPU and bandwidth overhead increases linearly with the number of SenseNodes and remains at a low level. A scalable SenseEdge also guarantees the real sampling

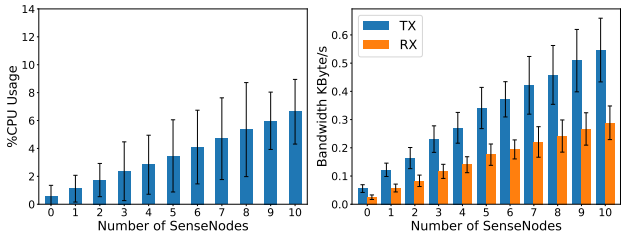

Fig. 8. %CPU usage and bandwidth usage of a SenseEdge when it is connected with different numbers of SenseNodes.

rate close to the required sampling rate. We set the required sampling rate to 0.5Hz, which is a quite fast sampling rate and faster than many sensors' response time. We connected ten I2C-based SenseNodes to one edge device and recorded the time interval between two valid readings of each SenseNode. The result illustrated in Fig. 9 verifies that the average real sampling rate is close to 0.5Hz.

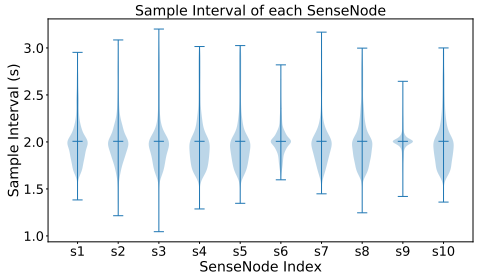

Fig. 9. The distribution of real sample intervals of each SenseNode when the expected interval is 2 seconds. The data is collected from one SenseEdge connected with ten I2C SenseNodes in one hour.

TABLE I
SENSELET++ Hardware Details

Name	*Main Components*	*Count*	*Price ($)*
Temp. & Humidity	SHT85, DS28E17	8	35
Airflow	D6F-V03A1, DS2438	3	30
Surface Temp.	MLX90614ESF, DS28E17	2	20
Magnetic Door	GF19002, DS2413	2	5
Water Leakage	RCHWES4/U, DS2438	1	25
SenseEdge	RaspberryPi 0-W, DS2482	4	25

B. Robustness to the Wireless Interference

We test the wireless interference in cleanrooms and check our design if it can easily increase the robustness to any potential interference. We set up one SenseEdge in the fume-hood (close to possible wireless interference) and another one right outside the fume-hood where the point-to-point distance between these two SenseEdges is less than 1 meter. To control the variables, we don't connect any sensors to these SenseEdges and also make sure both are linked to the same socket on the wall and have connected to the same wireless access point. Each SenseEdge keeps sending data to the SenseCloud to generate network traffic. We record the network traffic bandwidth of each SenseEdge from 4 am to 2 pm which covers the closing and opening time of the cleanroom. Figure 10 shows the results of this experiment.

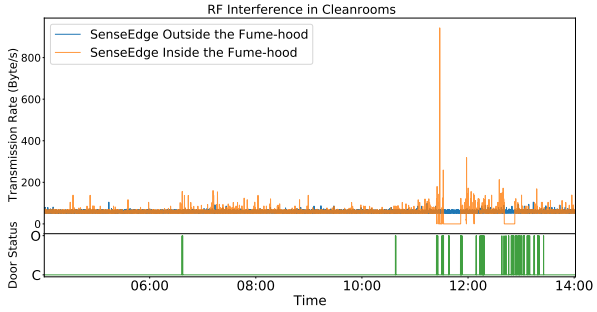

Fig. 10. Top graph shows the transmission rate of two SenseEdges. Bottom graph shows the door sensor data of the cleanroom which can indicate the occupancy of the cleanroom.

From the results in Fig. 10, we can find: (1) The interference only happens at working hours, which can prove the wireless interference is caused by some operations conducted in the cleanroom. (2) The SenseEdge outside the fume-hood is robust to the RF interference where its bandwidth is stable and never drops to zero. This can prove that by taking the advantage of our design, we can simply move the SenseEdge away from the interference to avoid such interference. On the contrary, the SenseEdge in the fume-hood which represents the traditional wireless sensor has an approximate 0.6 hour downtime in the 10-hour experiment.

C. Critical Anomaly Detection

In this subsection, we test the accuracy and latency of our critical anomaly detection pipeline. Critical anomalies are rare in the real world, then we manually trigger critical events around SenseNodes in order to gather enough data to validate the design. We emulate four events for four different kinds of sensors: (1) Fire events for the temperature sensor; (2) Overheat events for the surface temperature sensor; (3) Water leakage for the water leakage sensor; (4) Door open events for the door sensor. End-to-end latency is averaged over ten event invocations, recording the number of cases when the system successfully detects the event. We define end-to-end latency as the time when the SenseEdge invokes a sensor reading until the alert arrives at the user's server. Because some events such as fire cannot be emulated in cleanrooms, we run this experiment using the home-based test-bed. We increase the network transmission delay accordingly (54 ms) to bring our results closer to those measured values in the cleanroom. Table II shows results in this experiment. It verifies the rule-based algorithm is very effective and all critical events are successfully detected. We also notice the end-to-end latency introduced by the sensor reading, data processing, and network transmission is fairly low which demonstrates our system can detect critical anomaly detection in real-time.

TABLE II
CRITICAL ANOMALY DETECTION RESULTS

Event	*Alert Rule*	*Success Rate*	*Latency* Mean ± Std. (s)
Fire	$>$ 30°C	10 / 10	0.23 ± 0.09
Overheat	$>$ 70°C	10 / 10	0.14 ± 0.04
Water Leak	if True	10 / 10	0.29 ± 0.04
Door Open	if True	10 / 10	0.09 ± 0.03

D. SSA-based Anomaly Detection

We randomly choose a 5-days long subset of our dataset including two humidity time series and two temperature time series. We preprocess the data by averaging the data in each 10s interval to remove noise. After the preprocessing, each time series in the test dataset contains 43200 samples with a sampling interval of 10s. The SenseCloud will read in the time series continuously to emulate the online anomaly detection procedure and output anomaly detection results which will be compared with the ground truth. To obtain the ground truth of anomalies, we visually inspect and label each time series. We find similar patterns to the sample patterns shown in Fig. 6 and label them with the corresponding anomaly types. We label each anomalous event across a period of time and call the period *anomaly period*. We choose sliding window width W, sliding window step size S, and L, which is the size of trajectory matrix $\mathbf{X}$ used in equation 2 based on the characteristics of observed anomalies in cleanrooms. For each combination of anomaly type and physical property, we choose a threshold based on the historical observations. The parameters used in the experiment are listed in table III.

We use *number of false positives* and *number of false negatives* as our metrics to validate our algorithms. False-positive means our algorithm falsely detected non-exist anomalies. False-negative indicates our algorithm missed some anomalous events. When the anomaly identified by our algorithm hits an *anomaly period*, we consider the anomaly is successfully detected. The result is shown in table IV. x/y means our algorithms successfully find x anomalies out of y labeled anomalies. False-negative count is equal to $y - x$. From the result, we can find our algorithm has a high hit rate for both shape and trend anomalies and have only a few false positives. We carefully re-examine our dataset and find some false positive reports are real anomalies but are missed during the data labeling process. Figure 11 shows the detection result of the time series of a humidity sensor placed in the fume-hood. From the figure, we can find detected anomalies are highly close to patterns that people will find interesting. The average running time for each sliding window of each data stream is 68 ms with the parameters in table III. Considering the current

TABLE III
SSA-BASED ANOMALY DETECTOR SETTING

Parameter Name & Symbol	*Value*
Sliding Window Width W	180 samples (30 minutes)
Sliding Window Step S	30 samples (5 minutes)
Trajectory Matrix Size L	60 samples (10 minutes)
Anomaly Shape Threshold	1.5 (humidity); 0.125 (temperature)
Trend Change Threshold	0.15 (humidity); 0.08 (temperature)

TABLE IV
SSA-BASED ANOMALY DETECTION RESULT

Data Stream	*Shape Anomaly*	*Trend Anomaly*	*False Positive*
Humidity 1	5/6	11/11	2
Humidity 2	1/2	9/11	1
Temperature 1	0/0	7/8	8
Temperature 2	14/14	4/4	1
Total	20/22	31/34	12

step size of the sliding window is 5 minutes, the SenseCloud used in the home-based test-bed can process about 4000 data streams before the updating deadline, which demonstrates the scalability of our anomaly detection algorithm. Because anomalies detected by SSA are not time-critical, the 5-minute delay caused by the time-window step size is reasonable.

VI. CONCLUSION

We presented SENSELET++ for academic cleanrooms. SENSELET++ helped the cleanroom managers and researchers better understand the operation details of the cleanroom and allowed us to discover new challenges in sensor networks within challenging academic scientific environments such as their wireless interference, and other anomalies.

REFERENCES

[1] S. R. J. Ramson, S. Vishnu, and M. Shanmugam, "Applications of internet of things (iot) – an overview," in *2020 5th International Conference on Devices, Circuits and Systems (ICDCS)*, pp. 92–95, 2020.

[2] L. Catarinucci, D. de Donno, L. Mainetti, L. Palano, L. Patrono, M. L. Stefanizzi, and L. Tarricone, "An iot-aware architecture for smart healthcare systems," *IEEE Internet of Things Journal*, vol. 2, no. 6, pp. 515–526, 2015.

[3] M. F. Othman and K. Shazali, "Wireless sensor network applications: A study in environment monitoring system," *Procedia Engineering*, vol. 41, pp. 1204–1210, 2012. International Symposium on Robotics and Intelligent Sensors 2012 (IRIS 2012).

[4] K. Nahrstedt, Z. Yang, T. Yu, P. Su, R. Kaufman, I. Shan, S. Konstanty, M. McCollum, and J. Dallesasse, "Senselet: Distributed sensing infrastructure for improving process control and safety in academic cleanroom environments," *GetMobile: Mobile Computing and Communications*, vol. 24, pp. 12–16, 09 2020.

[5] D. Awtrey and D. Semiconductor, "Transmitting data and power over a one-wire bus," *Sensors-The Journal of Applied Sensing Technology*, vol. 14, no. 2, pp. 48–51, 1997.

[6] J. B. Elsner and A. A. Tsonis, *Singular spectrum analysis: a new tool in time series analysis*. Springer Science & Business Media, 2013.

[7] Ii Jin Kim, Sang Do Han, Hi Dock Lee, and Jin Suk Wang, "Fabrication and design of an auto ventilation controller using micro gas sensor for the clean room/building environment," in *SENSORS, 2005 IEEE*, pp. 3 pp.–, 2005.

[8] M. Loomans, P. Molenaar, H. Kort, and P. Joosten, "Energy demand reduction in pharmaceutical cleanrooms through optimization of ventilation," *Energy and Buildings*, vol. 202, p. 109346, 2019.

[9] G. Marques and R. Pitarma, "An internet of things-based environmental quality management system to supervise the indoor laboratory conditions," *Applied Sciences*, vol. 9, no. 3, 2019.

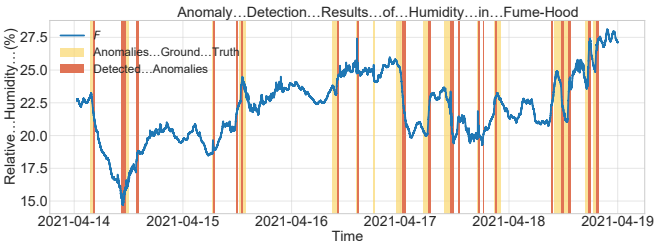

Fig. 11. Blue line shows the humidity in the fume-hood. Yellow regions represent the ground truth anomaly periods. Red regions show the detected anomalies by our algorithm.

[10] T. Seco, J. Bermudez, J. Paniagua, and J. A. Castellanos, "Dynamic and heterogeneous wireless sensor networks for virtual instrumentation services: Application to perishable goods surveillance," in *2011 IEEE Eighth International Conference on Mobile Ad-Hoc and Sensor Systems*, pp. 849–854, 2011.

[11] "Data center power management, DCIM software, and KVM-over-IP." https://www.raritan.com/.

[12] "Network technologies inc." https://www.networktechinc.com/environment-monitor-5d.html.

[13] Hyun-jae Yoo, J. Seo, M. Shin, and H. Suh, "Study of data acquisition and communication equipment for micro-grid system," in *2009 IEEE 13th International Symposium on Consumer Electronics*, pp. 671–675, 2009.

[14] M. Kanai and M. Inoue, "Home information management system with automatic acquisition of consumer electronics information," in *2013 IEEE International Symposium on Consumer Electronics (ISCE)*, pp. 261–262, 2013.

[15] A. Cardozo, A. Yamin, R. Souza, P. Davet, J. Lopes, and C. Geyer, "Sensing and actuation in iot: An autonomous rule based approach," in *2016 IEEE 13th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, pp. 355–360, 2016.

[16] H. Yao, H. Wu, D. He, and M. Wu, "A fusion method of multiple sensors data on panorama video for airport surface surveillance," in *2018 IEEE 15th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, pp. 500–504, 2018.

[17] P. Nguyen, S. Konstanty, T. Nicholson, T. O'Brien, A. Schwartz-Duval, T. Spila, K. Nahrstedt, R. H. Campbell, I. Gupta, M. Chan, K. Mchenry, and N. Paquin, "4ceed: Real-time data acquisition and analysis framework for material-related cyber-physical environments," in *2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*, pp. 11–20, 2017.

[18] Z. Yang, P. Nguyen, H. Jin, and K. Nahrstedt, "Miras: Model-based reinforcement learning for microservice resource allocation over scientific workflows," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, pp. 122–132, IEEE, 2019.

[19] L. Njilla, L. Pearlstein, X. W. Wu, A. Lutz, and S. Ezekiel, "Internet of things anomaly detection using machine learning," in *2019 IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*, pp. 1–6, 2019.

[20] G. R. Abuaitah and B. Wang, "Data-centric anomalies in sensor network deployments: analysis and detection," in *2012 IEEE 9th International Conference on Mobile Ad-Hoc and Sensor Systems (MASS 2012)*, vol. Supplement, pp. 1–6, 2012.

[21] Y. Liu, Z. Pang, M. Karlsson, and S. Gong, "Anomaly detection based on machine learning in iot-based vertical plant wall for indoor climate control," *Building and Environment*, vol. 183, p. 107212, 2020.

[22] J. Qi, Y. Chu, and L. He, "Iterative anomaly detection algorithm based on time series analysis," in *2018 IEEE 15th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, pp. 548–552, 2018.

[23] R. G. d. S. Ramos, P. R. L., and J. V. d. M. Cardoso, "Anomalies detection in wireless sensor networks using bayesian changepoints," in *2016 IEEE 13th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, pp. 384–385, 2016.

[24] M. A. Hayes and M. A. M. Capretz, "Contextual anomaly detection in big sensor data," in *2014 IEEE International Congress on Big Data*, pp. 64–71, 2014.

[25] R. A. Light, "Mosquitto: server and client implementation of the mqtt protocol," *Journal of Open Source Software*, vol. 2, no. 13, p. 265, 2017.

[26] N. Golyandina, V. Nekrutkin, and A. A. Zhigljavsky, *Analysis of time series structure: SSA and related techniques*. CRC press, 2001.