Journal of Robotics and Control (JRC)
Volume 6, Issue 1, 2025
ISSN: 2715-5072, DOI: 10.18196/jrc.v6i1.24345

IoT-Based Classroom Temperature Monitoring and Missing Data Prediction Using Raspberry Pi and ESP32

M. A. Navarrete-Sanchez [1], Re. Olivera-Reyna [2], Ro. Olivera-Reyna [3],
R. J. Perez-Chimal [4], J. U. Munoz-Minjares [5*]
[1,2,3,5] Unidad de Ingeniería Eléctrica Plantel Jalpa, Universidad Autónoma de Zacatecas, Zacatecas, México
[4] Posgrado en Ingeniería para la Innovación Tecnológica, Unidad de Ingeniería Eléctrica, Universidad Autónoma de Zacatecas, Zacatecas, México
Email: [1] mnavarrete@uaz.edu.mx, [2] reynel@uaz.edu.mx, [3] roliverar@uaz.edu.mx,
[4] janyluy@gmail.com, [5] ju.munoz@uaz.edu.mx
*Corresponding Author

***Abstract*—This study focuses on accurate temperature monitoring to optimize classroom conditions, enhancing student comfort and performance by providing precise data on temperature dynamics and ensuring reliability through advanced algorithms for handling missing data. Currently, advances in the Internet of Things (IoT) have enabled the development of simple, scalable, and intuitive systems for real-time environmental monitoring. This work presents a novel architecture for monitoring temperature dynamics in an electronic laboratory, leveraging a system of interconnected IoT devices with Wireless Fidelity (WiFi) communication. The system employs an ESP32 microcontroller and DS18B20 temperature sensors placed strategically around the classroom, including near windows and doors, to provide comprehensive data on heat distribution. The ESP32 is a small, low-cost, and powerful electronic chip that acts as the central processor for IoT systems, capable of handling data and connecting to a Wireless Network trough WiFi. While the DS18B20 can be defined as a digital sensor that accurately measures temperature and transmits the data electronically to a connected device. Therefore, the ESP32 microcontroller acts as the central processor, receiving temperature data from the DS18B20 sensors, which are configured to detect and transmit measurements. So, this data is then sent over a secure local WiFi network for real-time monitoring and analysis.**

The proposed system offers several advantages over existing solutions, including cost effective deployment, ease of integration, and real time monitoring. By using a secure local network for communication, it ensures reliable and uninterrupted data transmission. Furthermore, the I-UFIR algorithm was implemented to estimate missing temperature data points, significantly improving the accuracy of temperature readings and providing smoother, more reliable estimations. This system not only demonstrates the feasibility of IoT-based temperature monitoring in educational settings, but also highlights its potential to improve learning environments by optimizing classroom conditions.

***Keywords*—Internet of Things; Temperature; Prediction; I-UFIR; Node-RED**

I. INTRODUCTION

Maintaining optimal classroom temperatures is crucial for ensuring student comfort, focus, and overall well-being. Global warming has led to record high temperatures, which cause discomfort and impact daily activities, including those in classrooms [1]–[3]. Furthermore, uneven heat distribution caused by sunlight, airflow, or occupancy patterns can result in localized temperature variations that standard Heating, Ventilation, and Air Conditioning (HVAC) systems struggle to regulate effectively [4], [5]. These variations not only affect occupant comfort but also lead to inefficiencies in energy usage, as systems often overcompensate to maintain a uniform temperature [6]. Identifying and addressing these disparities through precise temperature monitoring can help optimize climate control strategies and enhance overall energy efficiency. In particular, classrooms in buildings dedicated to electronics and automation courses face additional challenges, as fluctuating temperatures can threaten sensitive materials such as electronics, books, and supplies [7]–[9]. Also, it has been identified that a bad temperature control in classrooms can impact in different aspects as such as cognitive performance, health, behavioral issues, and learning [10]. Therefore, it is essential to continuously monitor the thermal conditions of classrooms to maintain health, enhance productivity, and ensure safety.

Continually monitoring temperature through a network of sensors can achieve several benefits, including improved human comfort, energy efficiency, and protection of sensitive equipment [11]–[14]. A sensor network, defined as a set of spatially distributed sensors that cooperate to monitor and record conditions such as temperature, humidity, pressure, and motion, is a key tool for addressing these challenges.

Recently, the adoption of Internet of Things (IoT) has in-

creased due to its numerous applications [15]–[18]. Specifically, IoT has facilitated the control of comfort and human well-being [19]–[21]. In [22], a project was proposed to create a sensor network to collect data from various environmental sensors and publish it to cloud services using ESP32 and Raspberry Pi. Additionally, a Supervisory Control and Data Acquisition (SCADA) system was presented in [23], consisting of analog sensors, an ESP32, Node-RED, and Message Queuing Telemetry Transport (MQTT) for local communication. Regarding IoT systems and prediction, a scheme using the same hardware components previously described was proposed to predict maximum temperature and humidity values based on measurements from the previous three days, integrating an LSTM Neural Network [24]. Other sophisticated algorithms have been used to control room temperature based on Fuzzy Logic, ESP32, and IoT concepts [25], [26]. According to several studies, Wireless Sensor Networks have been implemented to supervise temperature in different rooms using various technologies, algorithms, and strategies, thus corroborating their increasing daily use [27]–[30]. Although the variety of IoT systems offers significant potential for temperature monitoring in classrooms, their limitations include issues related to connectivity, computational complexity, power consumption, sensor accuracy, scalability, security, and adaptability. Addressing these challenges is crucial for improving the effectiveness and reliability of IoT-based temperature monitoring solutions in educational settings.

While the use of IoT for environmental monitoring and control systems is increasingly widespread, it is important to recognize that electronic systems are not without their limitations. For instance, data acquisition can sometimes be interrupted due to several situations, such as electrical blackouts, disconnections, false contacts between components, or magnetic fields from other devices [31]–[34]. Therefore, an important task is to estimate these missing measurements in diverse applications. Furthermore, several algorithms have been proposed for this purpose, such as interpolation [35]–[37], traditional estimation algorithms [38]–[40], new implementations aimed at addressing this problem [41]–[43], and strategies based on machine learning [44]–[46]. However, the selection of an appropriate technique depends on the specific requirements of the application. To avoid the complexities of researching the model, noise statistics, or applying complex algorithms, there is an interesting algorithm called Iterative Unbiased Finite Impulse Response (I-UFIR) that can effectively predict missing measurements. Currently, there are several algorithms dedicated to filtering signals. However, these well-known algorithms require knowledge of the signal to be processed, an approximation model of the signal, a noise model, or extensive tuning to achieve optimal results. In contrast, the I-UFIR does not require this prior knowledge to obtain good results. Additionally, the flexibility of I-UFIR to function as a filter, smoother, or predictor outperforms other techniques.

Several existing IoT-based systems that monitor temperature in various environments often overlook certain critical aspects specific to classrooms, such as real time adaptability to changing conditions and the ability to handle data interruptions effectively. Although these systems integrate sensors and advanced algorithms, they often focus on more generalized environmental monitoring or employ complex prediction techniques that may require extensive tuning or prior knowledge of the signal and noise characteristics. Additionally, many of these systems rely on central air conditioning or assume ideal environmental conditions, which may not be applicable in diverse educational settings, particularly in classrooms without air conditioning.

In this work is proposed an IoT system for monitoring the temperature in a classroom without air conditioning. The main contributions of this research lie in the development of a real time temperature monitoring system for classrooms using IoT devices, which enables continuous tracking of heat behavior to optimize learning environments. The system incorporates strategically placed temperature sensors near windows and doors to collect comprehensive data on environmental conditions, ensuring accurate coverage of temperature variations throughout the building. Additionally, the simplicity of the system and scalability make it a practical solution for widespread deployment in educational settings. Once the system is implemented, it could be scaled to monitor temperature dynamics in larger institutions or across multiple classrooms, enhancing energy efficiency by identifying areas with uneven heat distribution and optimizing HVAC systems accordingly. These improvements could result in significant cost savings for educational institutions while ensuring a comfortable and conducive learning environment for students.

The system is proposed to record the temperature over several hours and to implement the I-UFIR algorithm as a predictor to estimate the absence of measurements due to faults in the data recording. The proposed physical system consists of a Raspberry Pi 4 as the Data management unit, an ESP32 development board with a DS18B20 temperature sensor as the data transmitter, and a modem to create a closed Wi-Fi network. All devices are planned to be linked using the software Node-RED. So, this work is structures as follows. In Section II, the general diagram of the proposed system architecture and the details of each component are described. The I-UFIR algorithm and its tuning are presented in Section III-B. Next, the complete network configuration is specified in Section IV. Finally, the results and conclusions are detailed in Sections V and VI.

II. Architecture of the System

The proposed strategy can be summarized in the Fig. 1, which is composed of several physical components with Wireless Fidelity (WiFi) communication. The principal device is a modem, which generate a local network with restricted access by password. This modem is protected with a Wi-Fi Protected

Access 2 (WPA2), which provides stronger encryption and authentication through Advanced Encryption Standard (AES) and Counter Cipher Mode with Block Chaining Message Authentication Code Protocol to address vulnerabilities in WPA and improve Wi-Fi network security. To guarantee the security of communication between the ESP32, Raspberry Pi, and Node-RED when using MQTT, it was employed the Transport Layer Security / Secure Sockets Layer (TLS/SSL) encryption. In addition, this protocol can be combined with username/password authentication to ensure the integrity and confidentiality of the data that are exchanged between devices.

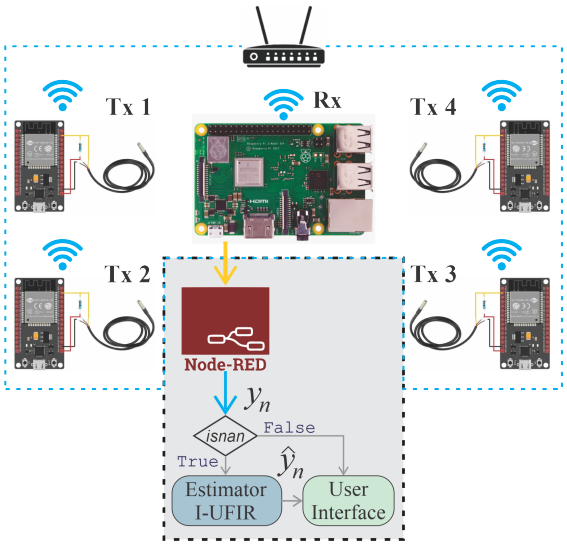

Fig. 1. General IoT Diagram of Wireless Sensor Network for Missing Data Prediction Using Raspberry Pi, Node-RED, and ESP32.

Now, it is necessary to configure the transmitters, which are numbered from Tx 1 to Tx 4 and consist of the ESP32 microcontroller and a DS18B20 temperature sensor. The ESP32 was selected over alternative microcontrollers, such as the Arduino family, due to its built-in Wi-Fi capabilities, which are essential for seamless IoT communication, as well as its low power consumption, making it highly efficient for long term deployment in networked environments. Additionally, the processing of ESP32 processing power and flexibility in handling data transmission make it an ideal choice for the proposed temperature monitoring system.

The transmitters, described above, are located at Tx 1, Tx 2, Tx 3, and Tx 4, in a corner next to another room, in a corner next to a window, in a corner next to another room, and in a corner next to a door, respectively. The transmitters were strategically placed near windows, doors, and room corners to capture temperature variations caused by environmental factors such as airflow, direct sunlight, and heat sources, which are common in classroom settings. The sensors were placed in these locations because windows often contribute to significant temperature variations due to direct sunlight exposure or drafts caused by poor insulation. Sensors were positioned near doors because doors are frequently used and can cause temperature fluctuations due to the exchange of air between the classroom and adjacent spaces. Additionally, sensors were placed in the corners of the room, as these areas can sometimes serve as zones where temperature gradients occur due to limited airflow or insulation irregularities. The hardware employed to archive the measurements is the Raspberry Pi 4. Although there are alternatives to using the Raspberry Pi, this device stands out for its affordability, strong community support, versatility, built-in connectivity, flexibility in hardware integration, low power consumption, processing power, and scalability. These advantages make it ideal for a wide range of projects, especially in IoT and educational applications. The devices Raspberry Pi 4 and ESP32 are described with detail in the sections II-A and II-B, respectively.

A. Raspberry Pi 4

The Raspberry Pi is a versatile and compact computing platform widely used in educational, research, and development projects due to its flexibility and robust features. The Raspberry Pi is a Single-Board Computer (SBC) with several pins called GPIO (General Purpose Input/Output), which are connections that allow the Raspberry Pi to communicate with the outside world [47]–[49]. This small computer is capable of controlling devices, reading information, and communicating with other devices effectively. Additionally, the Raspberry Pi can perform common tasks such as processing and storing information, browsing the internet, or creating documents.

The recommended operating system for this device is the open-source Raspberry Pi OS (formerly Raspbian), a very lightweight system based on Debian. However, any operating system compatible with the Advanced RISC Machine (ARM) architecture can be used [50]. Specifically, for this work, the Raspberry Pi's ability to establish an intuitive and fast connection with other electronic development boards was the key factor in selecting this technology. The main characteristics of the Raspberry Pi 4 are described in Table I [51].

TABLE I. CHARACTERISTICS RASPBERRY PI 4

Features	**Raspberry Pi 4**
Processor	Broadcom BCM2711, Quad-core Cortex-A72 (ARM v8) 1.5GHz
RAM	2GB, 4GB or 8GB LPDDR4
Storage	microSD (up to 1TB)
Wi-Fi Connectivity	802.11ac, dual-band (2.4GHz/5GHz)
Bluetooth	Bluetooth 5.0
USB Ports	2x USB 3.0, 2x USB 2.0
GPIO Pins	40 GPIO pins
Video Support	2x micro HDMI, up to 4Kp60
Operating System	Raspbian, Ubuntu, and others
Power Consumption	Relatively high, around 5-7W

In this table is described an overview of the key characteristics of the Raspberry Pi 4, including its processor,

memory options, connectivity features, and power consumption. In particular, it supports dual-band Wi-Fi, Bluetooth 5.0, and offers flexible storage via a microSD card, with video output capable of up to 4K resolution. These features are what give the Raspberry Pi its outstanding performance compared to other programmable boards. Among the many advantages of the Raspberry Pi, its powerful processor, support for 4K multimedia, and a great software community can be highlighted. However, some disadvantages can also be mentioned, such as higher power consumption and relatively high price compared to microcontrollers.

Regarding the software, the Raspberry Pi requires downloading and installing the operating system (OS) from the official Raspberry Pi website [52]. The OS is installed on a microSD card, and for this work, we recommend using Raspberry Pi OS (Desktop) to take advantage of its graphical interface. After installing the OS, it is necessary to update the system, which can be done using basic commands provided on the official Raspberry Pi website. Once the Raspberry Pi OS is installed and updated, the Node-RED dashboard can be deployed. The stepwise procedure for this installation is detailed on the official Node-RED website [53]. If both platforms are correctly installed, an IP address will be provided, allowing Node-RED to be accessed and executed through any internet browser.

B. ESP32

The ESP32 is a small yet powerful microcontroller, highly popular in the world of electronics and programming. This device has low power consumption and is a key tool for IoT projects [54]–[56]. Like the Raspberry Pi 4, the ESP32 features several GPIO pins that can send or receive digital signals, enabling interaction with various devices, from sensors to actuators. Basically, the ESP32 is a versatile and relatively easy-to-use microcontroller for electronics and IoT projects. Studies show that it can be easily integrated with minimal configuration [57], [58], and it is commonly used in projects involving MQTT and Node-RED [59], [60]. The key features of the ESP32 are listed in Table II [61].

TABLE II. CHARACTERISTICS OF ESP32

Features	**ESP32**
Processor	Tensilica Xtensa LX6, dual-core up to 240 MHz
RAM	520KB SRAM
Storage	External Flash memory 2MB to 16MB
Wi-Fi Connectivity	802.11 b/g/n (2.4GHz)
Bluetooth	Bluetooth 4.2 and BLE
USB Ports 2.0	Not available
GPIO Pins	Up to 34 GPIO pins
Video Support	Not available
Operating System	FreeRTOS, ESP-IDF, Arduino
Power Consumption	Very low, ideal for battery-powered IoT applications

Some advantages of the ESP32 are its very low power consumption and lower cost compared to development boards. The limited processing and memory capacity, the lack of USB ports, and the need for greater programming knowledge for complex projects are the main disadvantages of the ESP32.

C. Sensor DS18B20

The DS18B20 is a popular digital temperature sensor produced by Maxim Integrated. It is widely used in various applications due to its simplicity, accuracy, and ease of integration. This component is a 1-wire digital temperature sensor that provides temperature readings with a high degree of accuracy. It has a range of -55°C to +125°C and offers a digital output, meaning it can interface directly with digital devices like microcontrollers. Some key specifications of the DS18B20 include an accuracy of ± 0.5°C from -10°C to $+85$°C, and a configurable resolution from 9 to 12 bits, providing temperature precision from 0.5°C to 0.0625°C [62]. Regarding power supply, it can be powered either by an external supply from 3.0V to 5.5V.

The DS18B20 sensor has multiple advantages, especially in scenarios that require multiple temperature measurements with minimal wiring complexity. Its high accuracy, ease of integration, and wide operating range make it a popular choice for DIY projects, industrial systems, and IoT applications.

D. Node-RED

Node-RED is a platform designed for creating applications based on flow programming, which is a way to program as a network of black boxes, called "nodes" [63]. The main objective of Node-RED is to develop IoT systems, the configuration process is similar to block programming [64], [65]. Each node performs a specific function based on the provided input and the data flow between them [66]. This programming modality is flexible and can accommodate a wider range of users. Node-RED is well-suited for this work because it incorporates the MQTT protocol, which enables stable communication between the ESP32 and Raspberry Pi.

1) MQTT Protocol: The Message Queuing Telemetry Transport (MQTT) is the protocol of data exchange for IoT messaging [67], [68]. Recently, the MQTT was created to define as to publish and to subscribe the data over the internet between IoT devices. So, the Publisher and Subscriber share information via Topics, and the connection between them is handled by the MQTT broker.

Since the system proposed can be susceptible to errors of network disruptions, sensor failures, or other potential issues, several strategies can be implemented. For instance, the ESP32 microcontroller can be programmed to periodically verify its connection to the local Wi-Fi network and reconnect automatically in the event of a disruption. Similarly, the DS18B20 sensors can include error-checking protocols to detect faulty readings or disconnections, prompting the ESP32 to flag the issue for user attention. To address sensor failures, redundant DS18B20 sensors could be deployed at critical locations,

enabling the system to switch seamlessly to backup sensors without interrupting data collection. Additionally, the Raspberry Pi, acting as the central processing hub, could log errors and notify users via email or SMS when network disruptions or sensor anomalies are detected. However, a widely employed strategy is the approximation of missing data using estimation algorithms. To minimize the impact of missing measurements, an algorithm specifically developed for this purpose is described below.

III. Estimator UFIR

This section describes the State-Space Modeling to compensate for the absence of measurements in a defined process, such as satellite measurements, along with the detailed Iterative UFIR Algorithm. Additionally, the pseudo-algorithm is presented in detail for its implementation.

A. State-Space Modeling

The acquisition of the temperature signal can be represented under a discrete state-space model through the following equations:

$$\mathbf{x}_k k = \mathbf{A}\mathbf{x}_k k-1+\mathbf{B}w_k,, \quad (1)$$

$$\mathbf{y}_k = \mathbf{C}\mathbf{x}_k + v_k, \quad (2)$$

where $\mathbf{x}_k \in \mathbb{R}^K$ is the state vector, $\mathbf{y}_k \in \mathbb{R}^M$ is the observation vector, and $\mathbf{A} \in \mathbb{R}^{K\times K}$ is the state matrix, $\mathbf{B} \in \mathbb{R}^{K\times P}$, and $\mathbf{C} \in \mathbb{R}^{M\times K}$.

The Gaussian white noise vectors $w_k \sim \mathcal{N}(0, Q_k) \in \mathbb{R}^P$ and $v_k \sim \mathcal{N}(0, R_k) \in \mathbb{R}^M$ are considered zero-mean, $E\{w_k\} = 0$ and $E\{v_k\} = 0$, with covariances $E\{w_k w_n^T\} = Q_k\delta_{k-n}$ and $E\{v_k v_n^T\} = R_k\delta_{k-n}$, and the property $E\{w_k v_n^T\} = 0$ for all n and k.

Based on equations (1) and (2), the iterative UFIR filter algorithm can be designed as follows.

B. Iterative UFIR

The Iterative UFIR (I-UFIR) has been widely implemented to solve several problems [69]–[71]. Below, the UFIR filter algorithm is described in an iterative format, akin to the Kalman filter framework [72]. **Algorithm 1** presents the pseudo-code for this filter, where $\mathbf{y}_k$ represents the noisy input signal, N denotes the window or point horizon, and q is a step variable that can be determined using either equation (3).

$$q = \frac{N-1}{2} - \sqrt{\frac{N^2-1}{12}} \quad (3)$$

This parameter can be related as $p = -q$, which serves for filtering with $p = 0$, smoothing with $p < 0$, and prediction with $p > 0$, providing flexibility to the **Algorithm 1**.

Algorithm 1 Iterative UFIR Filter Algorithm

Input: $y_k, N, \mathbf{A}, \mathbf{C}, \mathbf{W}, q$
Begin
for $k = N-1, N$ **do**
  $m = k - N + 1, \quad s = k - N + K$
  $\mathbf{G}_s = (\mathbf{W}_{m,s}^T \mathbf{W}_{m,s})^{-1}\mathbf{Y}_{m,s}$
  $\hat{x}_s = \mathbf{G}_s(\mathbf{W}_{m,s}^T)\mathbf{Y}_{m,s}$
  for $i = s+1 : k$ **do**
    $\tilde{x}_i^- = \mathbf{A}_i\tilde{x}_{i-1}$
    $\mathbf{G}_i = \left[\mathbf{C}_i^T\mathbf{C}_i + \left(\mathbf{A}_i\mathbf{G}_{i-1}\mathbf{A}_i^T\right)^{-1}\right]^{-1}$
    $\mathbf{K}_i = \mathbf{G}_i\mathbf{C}_i^T$
    $\tilde{x}_i = \tilde{x}_i^- + \mathbf{K}_i\left(y_i - \mathbf{H}_i\tilde{x}_i^-\right)$
  end for
  $\hat{x}_k = \tilde{x}_k$
  $\hat{x}_{k-q} = \mathbf{A}^{-q}\hat{x}_k$
end for
Result: $\hat{x}_k$

IV. Network Configuration

A. Programming ESP32

The development platform ESP32 requires programming to read the sensor signal and transmit the information to the Raspberry Pi 4. Therefore, it is necessary to establish a pseudocode algorithm to summarize the code implemented on the ESP32. In **Algorithm 2**, the configuration to read and to transmit the DS18B20 sensor records is described stepwise. This pseudocode consists of one section that initializes the required libraries for the DS18B20 sensor, as well as the WiFi connection and MQTT protocol, specifically specifying the topic where the temperature is published. Next, the function *setup_wifi* establishes the WiFi connection and prints a message with the connection status. This is followed by the *callback* function, which requires two arguments the topic and the message to print the received message from the topic variable.

This Arduino-based program uses an ESP32 microcontroller to monitor temperature via a DallasTemperature sensor (DS18B20) and transmit readings to a Raspberry Pi MQTT broker. The setup function initializes serial communication, WiFi connectivity to the RED_INDUSTRIAL_IEI network, and MQTT communication with the broker at 192.168.1.70, calling the setup_wifi and callback functions for configuration. The reconnect section ensures the MQTT connection remains active, attempting reconnection every 5 seconds if it drops and subscribing to the topic upon success. The main logic resides in the loop function, which periodically reads the sensor data using the OneWire library, converts the temperature to a string, prints a "sending" message, and publishes the data to the esp/Temperature4 topic. It also listens for messages on the esp/Output4 topic and employs a non-blocking delay mechanism to ensure continuous operation. Robust error handling

ensures reconnection and uninterrupted data transmission.

Algorithm 2 ESP32 Configuration

Input: Sensor
Begin
Initialize WiFi and MQTT client
Set ssid and password for WiFi
Set mqtt_server IP address
Function setup_wifi:
  Connect to WiFi network
  Print connection status
Function callback(topic, message):
if topic matches output **then**
  Print received message
end if
Function setup:
  Initialize serial communication
  Call setup_wifi
  Set MQTT server and callback function
Function reconnect:
while not connected to MQTT: **do**
  Attempt to connect to MQTT
  if connection successful **then**
    Subscribe to output topic
  else
    5 seconds and retry
  end if
end while
Function loop:
if not connected to MQTT **then**
  call reconnect
end if
call client.loop
if sufficient time has passed or band is true **then**
  request temperature from sensor
  convert temperature to string
  print sending message
  publish temperature to topico
  set band to true
  wait 1 second to check callback
end if
Result: topico

B. Node Settings

Once each node is programmed to transmit temperature data to the Raspberry Pi board, it is necessary to configure the Node-RED platform. Specifically, the Message Queuing Telemetry Transport (MQTT) protocol is used to link the received data to the other nodes. With respect to the Quality of Service (QoS) levels, the MQTT node provides different levels of reliability. For the configuration, the maximum level, QoS=2, was chosen, which is ideal for critical applications requiring strict data accuracy and reliability, such as financial or medical data transmissions. Fig. 2 shows the complete Node-RED structure for recording measurements in a file and displaying them graphically.

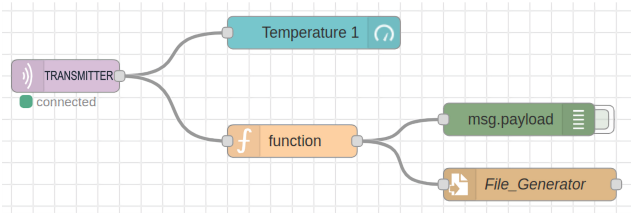

Fig. 2. Configuration proposed in Node–Red.

In Fig. 2, the first node is the “input MQTT”, which uses a broker such as Mosquitto on the same Raspberry Pi. Therefore, configuring MQTT requires only a few parameters the MQTT broker’s IP address, port number, and, importantly, the topic to which the temperature data will be sent. For this work, a closed Wi-Fi network was used, assigning the MQTT server the IP address 192.168.1.70, with port 1883, and naming the topic *esp/Temperature#*. The symbol # is replaced according to the specific sensor node from which the data is received.

Node-RED also provides a User Interface (UI) that facilitates the display of information in real-time with minimal adjustments. The integration of Node-RED and its UI played a key role in the success of the system, their eases of implementation and flexibility in system management made Node-RED an invaluable component for managing the IoT-based architecture. For this experiment, a Gauge Chart is proposed to present the temperature from each transmitter. This graph is directly connected to the output of the MQTT node, as shown in Fig. 2.

Next, the function node is arranged, this step requires a more detailed configuration because it is responsible for organizing the received information. This setup is described and outlined in Fig. 3. Here, the data is organized by establishing the year, month, day, hour, minute, and second, followed by the temperature in degrees Celsius (°C) measured with the DS18B20 sensor.

```
var d= new Date();
var fecha = d.getFullYear()+"/"+(d.getMonth()+1)+"/"...
+d.getDate()+" "+d.getHours()+":"+d.getMinutes()+":"+...
d.getSeconds();
var temp = Number(msg.payload);
var data = fecha+","+temp
msg.payload = data;
return msg;
```

Fig. 3. Function block configuration.

Finally, the function *msg.payload* receives the temperature measurement, and the *File_Generator* creates a file with a .txt extension at a specified name and path.

V. Testing and Results

A. Experimental Setup

As soon as the proposed system is configured, it will be possible to analyze the received data. The system collects temperature data through DS18B20 sensors connected to ESP32 microcontrollers, which periodically measure and prepare the data for transmission. The ESP32s then use Wi-Fi to send this data in real time to the Raspberry Pi, which acts as the central hub for data reception and processing. The Raspberry Pi ensures the consistency and reliability of the data, employing the I-UFIR algorithm to estimate and fill in any missing values. Finally, the processed data is sent to the Node-RED platform hosted on the Raspberry Pi, where it is displayed on a graphical interface with gauge charts and fast updates for each transmitter.

For this research, three probes were conducted, recording measurements from each transmitter every 5 minutes. The classroom where the tests were conducted is located on the first floor of a two-story building. Each transmitter is located in the indoor area of a classroom at a height of 1 meter, following the sequence from transmitter 1 to 4: near window 1, in the corner, near the door, and near window 2, respectively. The experiment began at 15:00 hours and ended after two days at 12:00 hours, resulting in a total recording duration of 69 hours. This experiment duration was chosen to capture a comprehensive range of classroom temperature fluctuations over different day night cycles. So, this duration ensures the system records temperature variations caused by daily environmental factors such as sunlight, ambient temperature changes, and human activity during the day, as well as the cooler, more stable conditions typical at night. By including at least two full cycles, the data provides a robust representation of typical classroom conditions, allowing for a more accurate analysis of patterns and influences on temperature dynamics. As an initial test, measurements were taken with the sensors placed in the same position at the center of the classroom to ensure they recorded the same temperature. This temperature was compared with an ambient thermometer, which showed a difference of ± 2°C.

For the graphical representation of the data from each transmitter, a user-friendly interface was designed based on gauge monitors, as shown in Fig. 4. Each gauge monitor is configured to show the temperature from Transmitter 1 to Transmitter 4. The Gauge Chart was chosen to represent the temperature graphically due to its ability to provide an intuitive and immediate understanding of temperature dynamics for end-users. The dynamic updates of gauge graphs ensure that users are always informed about the most recent data trends, making it easier to identify anomalies or significant changes. In addition to gauge graphs visualization, the system benefits from saving data in .txt files. This feature allows users to archive historical data for further analysis and reporting. Saving the data in a simple, universally readable format like *.txt* ensures compatibility with a wide range of analytical tools, enabling easy processing and sharing.

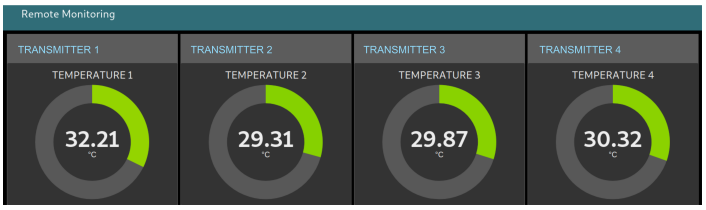

Fig. 4. User Interface UI configured in Node-RED to monitor the temperature of four transmitters.

B. Temperature Measurements

After completing the configuration of transmitters, the system is ready to obtain the temperature measurements. In Figs. 5(a), 5(b), and 5(c) show three different probes, which began on April 20, April 23, and April 26, respectively. The maximum temperatures in probe 1 were recorded by Sensor 1, reaching 33.38 °C, while the minimum value was recorded by Sensor 3, at 26.31 °C. In the second probe, the maximum measurements were also obtained by Sensor 1, but the lowest value was recorded by Sensor 2. In probe 3, Sensor 1 again recorded the maximum values, while the minimum values were obtained by Sensor 3.

The possible explanation for why node 1 recorded the highest temperatures is that the afternoon sun hits the windows directly, unlike node 2, where a tree partially shades the window. The observed temperature variation of nearly 5°C between sensors highlights significant differences in thermal dynamics within the classroom. This disparity reflects the influence of environmental factors such as direct sunlight, airflow patterns, insulation quality, and the positioning of heat sources, for example occupants or electronic devices. To guarantee the integrity and reliability of the temperature measurements, the recorded data were cross-validated with local temperature reports and digital platforms accessed via mobile devices, ensuring the measurements aligned with expected values for the time and location. Additionally, a review of the data revealed no significant outliers, indicating consistent and reliable measurements, even without applying advanced statistical methods.

It is possible to appreciate that the measurements in Fig. 5(a), Fig. 5(b), and Fig. 5(c) follow the same behavior, with high temperatures between 12:00 h and 00:00 h and low temperatures between 00:00 h and 12:00 h. It is worth highlighting that these probes were conducted during a hot season in Mexico, and the classroom was closed except when opening to change the batteries of the transmitters. So, different conditions were taken into account. If the classroom was closed, it could create a scenario where heat accumulation might occur. Additionally, the heat generated by electronic equipment and its impact on the precision of sensors were not studied in this experiment.

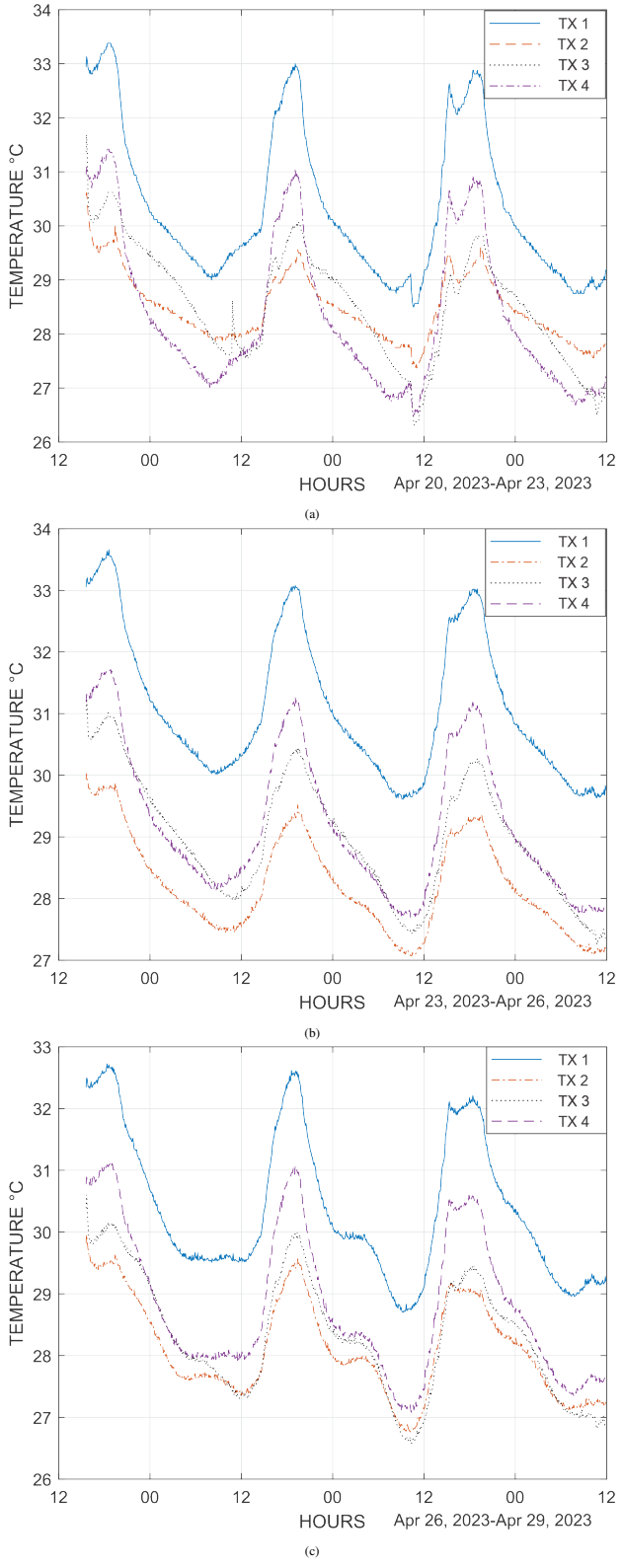

Fig. 5. Temperature measurements in classroom on several dates (a) April 20-April 23, 2023, (b)April 23-April 26, 2023, and (c) April 26-April 29, 2023

The observed temperature pattern, with higher temperatures during the afternoon and evening and lower temperatures overnight, is influenced by a combination of external environmental conditions and internal factors. Externally, this pattern aligns with the typical daily solar cycle, where sunlight and ambient heat peak in the afternoon and gradually dissipate overnight. Internally, classroom ventilation plays a crucial role. This process can amplify temperature peaks during the day due to the limited airflow or heat retention within the building structure. Additionally, the heat generated by occupants and electronic devices during active classroom hours may further contribute to elevated temperatures in the afternoon and evening.

C. Prediction Temperature

During the development and implementation of the sensor network, an error was detected when some sensors were unable to collect a temperature value. This record appears as empty in the .txt file, and when imported into the software Matlab, it was labeled as Not a Number (NaN). The origin of this fault is unknown. However,missing data can occur due to several potential factors. First, hardware issues, such as sensor malfunctions or faulty connections, can prevent the DS18B20 temperature sensors from accurately measuring and transmitting temperature data. Second, network disruptions, including Wi-Fi signal instability or temporary connectivity losses between the ESP32 microcontrollers and the Raspberry Pi, can result in data transmission failures. This could lead to gaps in the temperature data. Additionally, software bugs or errors in the data processing pipeline, such as incorrect configurations or failure in data handling logic.

The occasional missing data shows that the issue rarely occurred enough to seriously affect the reliability of the system. However, the presence of NaN values emphasizes the importance of the prediction algorithm in providing accurate and uninterrupted performance. This compensation mechanism allowed the system to handle occasional transmission issues while maintaining the integrity of the overall dataset. When this phenomenon is presented, the mean of each measurement cannot be computed, unless the data is compensated.

In this part, the UFIR algorithm is computed to predict the gap measurement. As discussed in III-B, the I-UFIR estimator has the ability to predict lost data using the available previous measurements [73]. This algorithm was implemented in Matlab R2020b, with the temperature measurements imported from *.txt* files.For this experiment, the I-UFIR algorithm was used with a window size of $N = 21$, and the q parameter is obtained using equation 3. Figs 6(a) to 6(b) illustrate the prediction of gap measurements by using different algorithms. In these Figs, the absence of temperature data for the Transmitter 1 in the Probe 1 difficult to graph the mean. When the temperature is not available from a transmitter, the mean is computed using

the data from the other probes, thus, the estimation can be compared with the mean. In Figs 6(a) and 6(b), it is most apparent how the mean (solid line) shows a displacement from the mean computed when all measurements are present (circles). In contrast, the prediction using the I-UFIR follows a stable behavior, without sharp changes. A notable disadvantage of implementing the I-UFIR algorithm is determining the optimal horizon. However, based on previous works mentioned earlier, it is possible to obtain the required results. In Fig. 6(c), the estimation and mean apparently have the same response due to variability in the existing values.

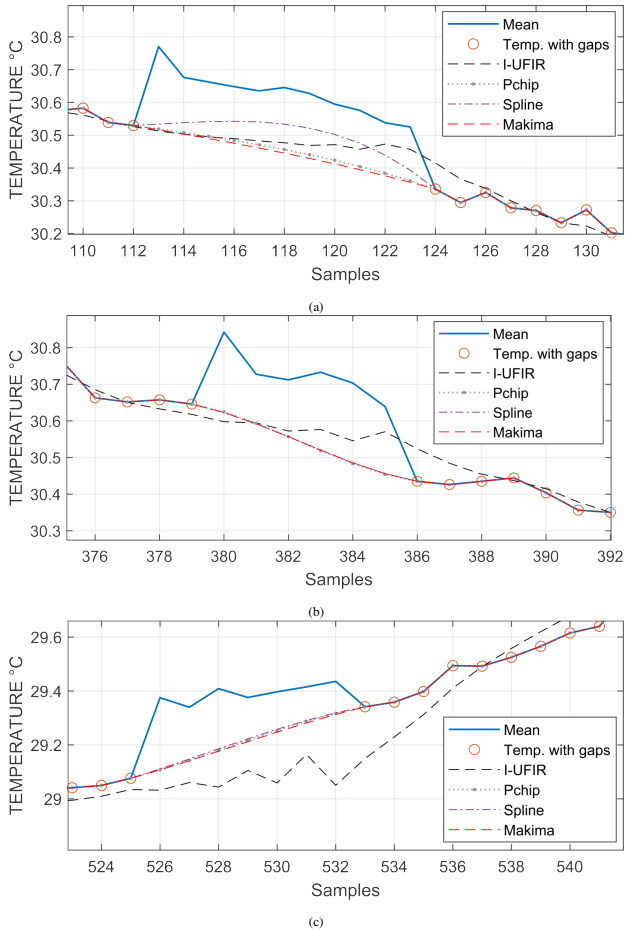

Fig. 6. Estimation of temperature data with gaps in different intervals on April 20-April 23, 2023, (a) gap 1, (b), gap 2, and (c) gap 3

Regarding to the interpolation techniques such as spline, makima, and pchip are commonly used to approximate missing data points or smooth datasets previously implemented in Matlab®. The spline function implements cubic spline interpolation, producing a smooth curve by minimizing the second derivative, which can lead to overshoots in highly oscillatory data [74]. The makima function, or Modified Akima Piecewise Cubic Hermite Interpolation, prioritizes robustness to outliers and local variations by avoiding overshoots while maintaining smooth transitions between data points [74]. The pchip function, or Piecewise Cubic Hermite Interpolating Polynomial, emphasizes monotonicity and preserves the shape of the data, making it suitable for scenarios where maintaining physical constraints is crucial [74]. However, these interpolation methods lack the robustness and adaptive capabilities of techniques like the I-UFIR filter, which can iteratively refine estimates, handle noisy datasets, and avoid assumptions about the smoothness of underlying data. Consequently, while spline, makima, and pchip are efficient for interpolating well behaved data, they may perform poorly in datasets with noise, irregularities, or measurement loss, where I-UFIR techniques excel in providing unbiased and reliable estimates.

The proposed IoT-based temperature monitoring system aligns with findings from other studies that highlight the effectiveness of IoT technologies in environmental monitoring. For instance, the use of ESP32 microcontrollers and similar sensors for temperature tracking has been reported as a cost-effective approach in educational and industrial settings [75], [76]. However, unlike previous studies that focus primarily on accuracy against commercial sensors [77], [78], this work emphasizes robustness in handling data loss through the I-UFIR algorithm, showcasing its ability to estimate missing values without abrupt changes, a key advancement over traditional interpolation techniques. Furthermore, while many IoT systems prioritize immediate data accuracy, this study broadens the scope by demonstrating the potential for long term monitoring in environments prone to interruptions, aligning with recent calls for more resilient IoT frameworks [79]. Future studies could build on this foundation by integrating additional environmental factors like humidity or CO2 levels, as suggested in [80], to enhance classroom comfort and energy efficiency, thereby expanding the system's real-world applications in educational institutions.

VI. Conclusions

The proposed architecture effectively achieves the monitoring of temperature dynamics in a classroom environment, specifically an electronic laboratory. Through the use of interconnected IoT devices with Wireless Fidelity (WiFi) communication, the system demonstrates a cost effective and efficient solution for capturing and analyzing heat distribution. The ESP32 microcontroller and DS18B20 temperature sensors, strategically placed near windows and doors, ensure comprehensive data collection. The integration with Node-RED and the UI application successfully validates the communication between the publisher and subscriber, displaying real time temperature measurements. To ensure the ethical deployment of the proposed IoT-based temperature monitoring system, data privacy and security must be prioritized. Implementing encryption protocols for data transmission and storage, as well as limiting access to

authorized personnel, will safeguard sensitive information and prevent potential misuse.

Temperature data were collected over nine days, with periodic interruptions every three days for system supervision. Each transmitter was placed in a different location within the same room, leading to temperature variations of up to 5 degrees Celsius between the sensor recording the lowest values and the one recording the highest. These differences highlight the impact of sensor placement on measurements. The proposed system focuses on extended temperature monitoring and the estimation of missing data to ensure reliable and continuous operation, prioritizing data continuity and the effectiveness of the I-UFIR algorithm in addressing data loss or sensor interruptions. While no reference system was available for direct comparison, this work emphasizes the robustness of system and reliability rather than specific temperature accuracy, showcasing its potential in environments prone to data loss.

Regarding the estimation of temperature gaps, the prediction using the I-UFIR algorithm was successfully applied, as the missing data were estimated and did not show abrupt changes. In other words, the estimations exhibited a similar behavior compared to the other temperature measurements. The ability of I-UFIR to provide smooth estimations ensures that the missing data points align closely with the natural fluctuations of the temperature measurements, avoiding abrupt changes that could skew the overall data analysis. This smooth data estimation is particularly important for maintaining the consistency and precision of the system, as any sharp discrepancies in the temperature readings could lead to incorrect conclusions or actions. In future work, we propose to conduct a comprehensive analysis of the robustness of I-UFIR algorithm by simulating larger gaps in data, such as intentionally removing measurements to evaluate its performance under varying conditions. This study will focus on understanding the impact of signal behavior, noise levels, and statistical properties on the algorithm's estimation accuracy, providing deeper insights into its reliability and limitations. Despite the aim of this system being focused on recording temperature measurements over extended periods and estimating missing data to ensure a reliable and continuous understanding of temperature dynamics, a comparison with other temperature systems or commercial sensors could be proposed to apply and evaluate various performance metrics.

The proposed IoT-based temperature monitoring system provides a practical and scalable solution for classroom environments, particularly smaller spaces. However, its current focus is limited to temperature monitoring, which restricts its ability to provide a comprehensive understanding of classroom conditions. Incorporating additional environmental factors, such as humidity, CO2 levels, or light intensity, could enhance its utility, enabling institutions to optimize energy usage, reduce costs, and improve student comfort and performance. Scaling the system to larger spaces with more complex environmental factors may also require adjustments to the architecture and transmitter placement.

Future improvements could involve enhancing the robustness of the algorithm to address diverse fault conditions and leveraging machine learning models to predict and dynamically adjust environmental factors based on historical data. These advancements would broaden the system's real world applications, offering a more comprehensive and reliable tool for environmental monitoring. Such developments could significantly benefit educational institutions by fostering better learning environments and supporting energy efficient practices.

REFERENCES

[1] G. Guevara, G. Soriano, and I. M. Rodriguez, "Thermal comfort in university classrooms: An experimental study in the tropics," *Building and Environment*, vol. 187, 2021, doi: 10.1016/j.buildenv.2020.107430.

[2] R. J. Park, J. Goodman, M. Hurwitz, and J. Smith, "Heat and learning," *American Economic Journal: Economic Policy*, vol. 12, no. 2, pp. 306–39, 2020, doi: 10.1257/pol.20180612.

[3] S. Domínguez–Amarillo, J. Fernández-Agüera, M.M. González, and T. Cuerdo–Vilches, "Overheating in schools: Factors determining children's perceptions of overall comfort indoors," *Sustainability*, vol. 12, no. 14, pp. 5772, 2020, doi: 10.3390/su12145772.

[4] A. C. Hurlimann, S. Moosavi, and G. R. Browne, "Climate change transformation: A definition and typology to guide decision making in urban environments," *Sustainable Cities and Society*, vol. 70, 2021, doi: 10.1016/j.scs.2021.102890.

[5] J. R. Short, and A. Farmer, "Cities and climate change," *Earth*, vol. 2, no. 4, pp. 1038–1045, 2021, doi: 10.3390/earth2040061

[6] H. S. Khan, R. Paolini, P. Caccetta, and M. Santamouris, " On the combined impact of local, regional, and global climatic changes on the urban energy performance and indoor thermal comfort—The energy potential of adaptation measures," *Energy and Buildings on ScienceDirect*, vol. 267, 2022, doi: 10.1016/j.enbuild.2022.112152.

[7] W. Hernandez and N. Cañas, "Sensing Classroom Temperature, Relative Humidity, Illuminance, CO2, and Noise: An Integral Solution Based on an IoT Device for Dense Deployments," *Sensors*, vol. 24, no. 16, 2024, doi: 10.3390/s24165129.

[8] M. Jowkar, H. B. Rijal, J. Brusey, A. Montazami, S. Carlucci, and T. C. Lansdown, "Comfort temperature and preferred adaptive behaviour in various classroom types in the UK higher learning environments," *Energy and Buildings*, vol. 211, 2020, doi: 10.1016/j.enbuild.2020.109814.

[9] S. Lin, E. Lipton, Y. Lu and C. Kielb, "Are classroom thermal conditions, lighting, and acoustics related to teacher health symptoms?," *Indoor Air*, vol. 30, no. 3, pp. 544–552, 2020, doi: 10.1111/ina.12640.

[10] R. Ahmed, D. Mumovic, E. Bagkeris, and M. Ucci, "Combined effects of ventilation rates and indoor temperatures on cognitive performance of female higher education students in a hot climate," *Indoor Air*, 2022, doi:10.1111/ina.13004.

[11] Y. Xiao, H. Li, C. Wang, S. Pan, J. He, A. Liu, and G. Lu, "Room temperature wearable gas sensors for fabrication and applications," *Advanced Sensor Research*, vol. 3, no. 3, 2024, doi: 10.1002/adsr.202300035.

[12] A. Estévez, J. J. F. Sokoudjou, J. I. Sancho, D. Valderas, I. Ochoa, and N. Pérez, "Chipless wireless sensor coupled with machine learning for oil temperature monitoring," *IEEE Sensors Journal*, vol. 23, no. 18, pp. 21234–21245, 2023, doi: 10.1109/JSEN.2023.3301668.

[13] P. Kumari, S. Kapur, V. Garg, and K. Kumar, "Effect of surface temperature on energy consumption in a calibrated building: A case study of Delhi," *Climate*, vol. 8, no. 6, 2020, doi: 10.3390/cli8060071.

[14] I. G. M. N. Desnanjaya, and I. N. A. Arsana, "Home security monitoring system with IoT–based Raspberry Pi," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 22, no. 3, 2021, doi: 10.11591/ijeecs.v22.i3.pp1295-1302.

[15] A. Khanna, and S. Kaur, "Internet of things (IoT), applications and challenges: a comprehensive review," *Wireless Personal Communications*, vol. 114, pp. 1687–1762, 2020, doi: 10.1007/s11277-020-07446-4.

[16] R. P. Singh, M. Javaid, A. Haleem, and R. Suman, "Internet of things (IoT) applications to fight against COVID–19 pandemic," *Diabetes & Metabolic Syndrome: Clinical Research & Reviews*, vol. 14, no. 4, pp. 521–524, 2020, doi: 10.1016/j.dsx.2020.04.041.

[17] T. Alam, "Cloud-based IoT applications and their roles in smart cities," *Smart Cities*, vol. 4, no. 3, pp. 1196–1219, 2021, doi: 10.3390/smartcities4030064.

[18] A. Abdelmaboud, A. I. A. Ahmed, M. Abaker, T. A. E. Eisa, H. Albasheer, S. A. Ghorashi, and F. K. Karim, " Blockchain for IoT applications: taxonomy, platforms, recent advances, challenges and future research directions," *Electronics*, vol. 11, no. 4, 2022, doi: 10.3390/electronics11040630.

[19] H. Sequeiros, T. Oliveira, and M. A. Thomas, "The impact of IoT smart home services on psychological well-being," *Information Systems Frontiers*, pp. 1–18, 2022, doi: 10.1007/s10796-021-10118-8.

[20] W. T. Sung, and S. J. Hsiao, "The application of thermal comfort control based on Smart House System of IoT," *Measurement*, vol. 149, 2020, doi: 10.1016/j.measurement.2019.106997.

[21] F. J. Dian, R. Vahidnia, and A. Rahmati, "Wearables and the Internet of Things (IoT), applications, opportunities, and challenges: A Survey," *IEEE access*, vol. 8, pp. 69200–69211, 2020, doi: 10.1109/ACCESS.2020.2986329.

[22] S. Thapa, and S. C. KC, *Raspberry Pi and ESP32-Based Smart Sensor Network for IoT Platform Integration and Real-Time Environmental Data Monitoring*, Bachelor's Thesis, Metropolia University of Applied Sciences, 2023.

[23] A. Zare, and M. T. Iqbal, "Low-cost ESP32, Raspberry Pi, Node-RED, and MQTT protocol based SCADA system," in *2020 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS)*, pp. 1–5, 2020, doi: 10.1109/IEMTRONICS51293.2020.9216412.

[24] K. E. Y. Laurel *et al*., "An Intelligent Humidity and Temperature Monitoring System using Long Short-Term Memory Neural Network and Raspberry Pi," *2023 IEEE 15th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM)*, pp. 1-6, 2023, doi: 10.1109/HNICEM60674.2023.10589111.

[25] Sunardi, A. Yudhana and Furizal, "Tsukamoto Fuzzy Inference System on Internet of Things-Based for Room Temperature and Humidity Control," in *IEEE Access*, vol. 11, pp. 6209-6227, 2023, doi: 10.1109/ACCESS.2023.3236183.

[26] I. Memon, R. A. Shaikh, M. K. Hasan, R. Hassan, A. U Haq, and K. A. Zainol, "Protect mobile travelers information in sensitive region based on fuzzy logic in IoT technology," *Security and Communication Networks*, vol. 2020, no. 1, 2020, doi: 10.1155/2020/8897098.

[27] A. Asif, and M. Zeeshan, "Indoor temperature, relative humidity and CO2 monitoring and air exchange rates simulation utilizing system dynamics tools for naturally ventilated classrooms." *Building and Environment*, vol. 180, 2020, doi: 10.1016/j.buildenv.2020.106980.

[28] N. Hutabarat, R. S. Siraita, J. Junaidi, and M. Pardede, "Implementation of a Wireless Sensor Network with Mesh Topology with XBee for Centralized Building Room Monitoring," *Journal of Computer Science, Information Technology and Telecommunication Engineering*, vol. 5, no. 2, pp. 581-588, 2024, doi: 10.30596/jcositte.v5i2.20002.

[29] S. Katsoulis, G. Koulouras and I. Christakis, "Energy-Efficient Data Acquisition and Control System using both LoRaWAN and Wi-Fi Communication for Smart Classrooms," *2024 13th International Conference on Modern Circuits and Systems Technologies (MOCAST)*, pp. 1-4, 2024, doi: 10.1109/MOCAST61810.2024.10615862.

[30] M. Yağanoğlu, F. Bozkurt, F.B. Günay, S. Kul, E. Şimşek, G. Öztürk, and S. Karaman, "Design and validation of IoT based smart classroom," *Multimedia Tools and Applications*, vol. 83, no. 22, pp. 62019–62043, 2024, doi: 10.1007/s11042-023-15872-2.

[31] K. Mohan, and J. Pearl, " Graphical models for processing missing data," *Journal of the American Statistical Association*, vol. 116, no. 534, pp. 1023–1037, 2021, doi: 10.1080/01621459.2021.1874961.

[32] E. M. D. van Rijn *et al*., "PELP: accounting for missing data in neural time series by Periodic Estimation of Lost Packets," *Frontiers in Human Neuroscience*, vol. 16, 2022, doi: 10.3389/fnhum.2022.934063.

[33] N. Berjab, H. H. Le, and H. Yokota, "Recovering missing data via top-k repeated patterns for fuzzy-based abnormal node detection in sensor networks," *IEEE Access*, vol. 10, pp. 61046–61064, 2022, doi: 10.1109/ACCESS.2022.3181742.

[34] R. Krishnamurthi, A. Kumar, D. Gopinathan, A. Nayyar, and B. Qureshi, "An overview of IoT sensor data processing, fusion, and analysis techniques," *Sensors*, vol. 20, no. 21, 2020, doi: 10.3390/s20216076.

[35] A. G. Rameshrao, E. Koley, and S. Ghosh, "A packet-loss resilient protection scheme for hybrid microgrids based on Markov chain model and spline interpolation," *Sustainable Energy, Grids and Networks*, vol. 35, 2023, doi: 10.1016/j.segan.2023.101121.

[36] D.M. Larson *et al*., "Reconstructing missing data by comparing interpolation techniques: Applications for long-term water quality data," *Limnology and Oceanography: Methods*, vol. 7, no. 21, pp.435–449, 2023, doi: 10.1002/lom3.10556.

[37] A. A. Attar, F. Schirle, and M. Hofmann, "Noise added on Interpolation as a Simple Novel Method for Imputing Missing Data from Household's Electricity Consumption," *Procedia Computer Science*, vol. 207, pp. 2253–2262, 2022, doi: 10.1016/j.procs.2022.09.284.

[38] M. V. Olguin *et al*., "Object tracking over distributed WSNs with consensus on estimates and missing data". *IEEE Access*, vol. 7, pp. 39448–39458, 2019, doi: 10.1109/ACCESS.2019.2905514.

[39] L. Padilla, B. L. Álvarez, J. Mateu, and E. Porcu, "Space-time autoregressive estimation and prediction with missing data based on Kalman filtering," *Environmetrics*, vol. 31, no. 7, 2020, doi: 10.1002/env.2627.

[40] J. You, X. Ma, Y. Ding, M. J. Kochenderfer, and J. Leskovec, "Handling missing data with graph representation learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 19075–19087, 2020, doi:/10.48550/arXiv.2010.16418.

[41] C. Gu, M. Zhu, Y. Wu, B. Chen, F. Zhou, and W. Chen, "Multi-output displacement health monitoring model for concrete gravity dam in severely cold region based on clustering of measured dam temperature field," *Structural Health Monitoring*, vol. 22, no. 5, pp. 3416-3436, 2023, doi: 10.1177/14759217221142006.

[42] R. J. Little, J. R. Carpenter, and K. J. Lee, "A comparison of three popular methods for handling missing data: complete-case analysis, inverse probability weighting, and multiple imputation," *Sociological Methods & Research*, vol. 53, no. 3, pp. 1105–1135, 2024, doi: 10.1177/00491241221113873.

[43] R. C. Pereira, M. S. Santos, P. P. Rodrigues, and P. H. Abreu, "Reviewing autoencoders for missing data imputation: Technical trends, applications and outcomes," *Journal of Artificial Intelligence Research*, vol. 69, pp. 1255–1285, 2020, doi: 10.1613/jair.1.12312.

[44] M. Pereira, and B. Glisic, "Detection and quantification of temperature sensor drift using probabilistic neural networks," *Expert Systems with Applications*, vol. 213, 2023, doi: 10.1016/j.eswa.2022.118884.

[45] T. Emmanuel, T. Maupong, D. Mpoeleng, T. Semong, B. Mphago, and O. Tabona, "A survey on missing data in machine learning," *Journal of Big data*, vol. 8, pp. 1–37, 2021, doi: 10.1186/s40537-021-00516-9.

[46] J. Josse, J. M. Chen, N. Prost, G. Varoquaux, and E. Scornet, "On the consistency of supervised learning with missing values," *Statistical Papers*, pp. 1–33, 2024, doi: 10.1007/s00362-024-01550-4.

[47] *Raspberry Pi Ltd, Raspberry Pi hardware documentation*, [Online]. Available: https://www. raspberrypi.com/documentation/computers/raspberry-pi.html. Accessed: 2022.

[48] H. D. Ghael, L. Solanki, and G. Sahu, "A review paper on raspberry pi and its applications," *International Journal of Advances in Engineering and Management (IJAEM)*, vol. 2, no. 12, 2020, doi: 10.35629/5252-0212225227.

[49] S. Monk, *Raspberry pi cookbook*, O'Reilly Media, Inc, 2022.

[50] K. Rzepka, P. Szary, K. Cabaj, and W. Mazurczyk, "Performance evaluation of Raspberry Pi 4 and STM32 Nucleo boards for security-related operations in IoT environments," *Computer Networks*, vol. 2024, no. 242, 2024, doi: 10.1016/j.comnet.2024.110252.

[51] *BCM2711, Quad-core Cortex-A72 (ARM v8) 64-bit SoC, BCM2711 Processor Datasheet*. [Online]. Available: https://datasheets.raspberrypi.com/bcm2711/bcm2711-peripherals.pdf. [Accessed: Dec. 8, 2024].

[52] Raspberry Pi, "Install Raspberry Pi OS using Raspberry Pi Imager", [Online]. Available: https://www.raspberrypi.com/software/. Accessed: 2022.

[53] Node-RED, "Running Node-RED locally," [Online]. Available: https://nodered.org/docs/get. Accessed: 2022.

[54] D. Hercog, T. Lerher, M. Truntič, and O. Težak, "Design and implementation of ESP32-based IoT devices," *Sensors*, vol. 23, no. 15, pp. 6739, 2023, doi: 10.3390/s23156739.

[55] R. B. Salikhov, V. K. Abdrakhmanov, and I.N. Safargalin, "Internet of things (IoT) security alarms on ESP32-CAM," In *Journal of Physics: Conference Series*, vol. 2096, no. 1, 2021, doi: 10.1088/1742-6596/2096/1/012109.

[56] G. P. Pereira, M. Z. Chaari, and F. Daroge, "IoT-enabled smart drip irrigation system using ESP32," *IoT*, vol. 4, no. 3, pp. 221–243, 2023, doi: 10.3390/iot4030012.

[57] A. Zare, and M. T. Iqbal, "Low-cost ESP32, Raspberry Pi, Node-RED, and MQTT protocol based SCADA system," In *2020 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS), IEEE*, pp. 1–5, 2020, doi: 10.1109/IEMTRONICS51293.2020.9216412.

[58] D. Witczak, and S. Szymoniak, "Review of Monitoring and Control Systems Based on Internet of Things," *Applied Sciences*, vol. 14, no. 19, 2024 doi: 10.3390/app14198943.

[59] M. J. A. Baig, M. T. Iqbal, M. Jamil, and J. Khan, "Design and implementation of an open-Source IoT and blockchain-based peer-to-peer energy trading platform using ESP32-S2, Node-RED and, MQTT protocol," *Energy reports*, vol. 7, pp. 5733–5746, 2021, doi: 10.1016/j.egyr.2021.08.190.

[60] D. A. T. Le *et al*., "Patient Remote Monitoring System Using MQTT Protocol for ECG Signals," In *2024 9th International Conference on Integrated Circuits, Design, and Verification (ICDV)*, pp. 114–118, 2024, doi: 10.1109/ICDV61346.2024.10617252.

[61] Espressif Systems, *ESP32 Series Datasheet*, versión 3.9. [Online]. Available: https://www.espressif.com/sites/default/files/documentation-/esp32_datasheet_en.pdf. [Accessed: 18-sep-2024].

[62] A. Elyounsi, and A. N. Kalashnikov, "Evaluating suitability of a DS18B20 temperature sensor for use in an accurate air temperature distribution measurement network," *Engineering Proceedings*, vol. 10, no. 1, pp. 56, 2021, doi: 10.3390/ecsa-8-11277.

[63] T. Hagino, *Practical Node-RED Programming: Learn powerful visual programming techniques and best practices for the web and IoT*, Packt Publishing Ltd, 2021.

[64] I. U. Onwuegbuzie, A. O. Olowojebutu, and K. K. Akomolede, "Node–RED and IoT Analytics: A Real-Time Data Processing and Visualization Platform," *Tech–Sphere Journal of Pure and Applied Sciences (TSJPAS)*, vol. 1, no. 1, pp. 1–12, 2020, doi: 10.5281/zenodo.13856859.

[65] I. V. Nițulescu, and A. Korodi, "Supervisory control and data acquisition approach in Node–RED: Application and discussions," *IoT*, vol. 1, no. 1, pp. 5, 2020, doi: 10.3390/iot1010005.

[66] Node-RED, *A visual tool for wiring the Internet of Things*, [Online]. Available: https://nodered.org. [Accessed: September 26 2024].

[67] MQTT, *An open standard messaging protocol for IoT*, [Online]. Available: https://mqtt.org. [Accessed: Sep. 26, 2024].

[68] A. J. Hintaw, S. Manickam, M. F. Aboalmaaly, and S. Karuppayah, "MQTT vulnerabilities, attack vectors and solutions in the internet of things (IoT)," *IETE Journal of Research*, vol. 69, no. 6, pp. 3368–3397, 2023, doi: 10.1080/03772063.2021.1912651.

[69] J. U. M. Minjares *et al*., "Navigation Path Estimation using UFIR Filter over G28U7FTTL GPS Receiver Data," In *2020 24th International Conference on Circuits, Systems, Communications and Computers (CSCC)*, pp. 169–173, 2020, doi: 10.1109/CSCC49995.2020.00038.

[70] Y. Xu, Y. S. Shmaliy, S. Bi, X. Chen, and Y. Zhuang, "Extended Kalman/UFIR filters for UWB-based indoor robot localization under time-varying colored measurement noise," *IEEE Internet of Things Journal*, vol. 10, no. 17, pp. 15632–15641, 2023, doi: 10.1109/JIOT.2023.3264980.

[71] M. V. Olguín, Y. S. Shmaliy, O. I. Manzano, and S. M. Figueroa, "Distributed UFIR filtering with applications to environmental monitoring," *International Journal of Circuits, Systems and Signal Processing*, vol. 5, pp. 349–355, 2021, doi: 10.46300/9106.2021.15.38.

[72] Y. S. Shmaliy, and S. Zhao, *Optimal and robust state estimation: Finite Impulse Response (FIR) and Kalman approaches*, John Wiley & Sons, 2022, doi: 10.1002/9781119863106.

[73] W. Xue, X. Luan, S. Zhao, and F. Liu, "A fusion Kalman filter and UFIR estimator using the influence function method," *IEEE/CAA Journal of Automatica Sinica*, vol. 9, no. 4, pp. 709-718, 2021, doi:10.1109/JAS.2021.1004389.

[74] D. Mishra, A. Kumar and V. S. Rathor, "Performance of Interpolation Techniques for Compression of Crop Image: A Comparative Study," *2023 International Conference on Computer, Electronics & Electrical Engineering & their Applications (IC2E3)*, pp. 1-7, 2023, doi: 10.1109/IC2E357697.2023.10262677.

[75] F. Furizal, S. Sunardi, and A. Yudhana, "Temperature and humidity control system with air conditioner based on fuzzy logic and internet of things," *Journal of Robotics and Control (JRC)*, vol. 4, no. 3, pp. 308–322, 2023, doi: 10.18196/jrc.v4i3.18327.

[76] R. Chataut, A. Phoummalayvane, and R. Akl, "Unleashing the power of IoT: A comprehensive review of IoT applications and future prospects in healthcare, agriculture, smart homes, smart cities, and industry 4.0," *Sensors*, vol. 23, no. 16, pp. 7194, 2023. doi:10.3390/s23167194.

[77] B. Mobaraki, S. Komarizadehasl, F. J. Castilla Pascual, and J. A. Lozano-Galant, "Application of low-cost sensors for accurate ambient temperature monitoring," *Buildings*, vol. 12, no. 9, 2022, doi: 10.3390/buildings12091411.

[78] N. U. Okafor, Y. Alghorani, and D. T. Delaney, "Improving data quality of low-cost IoT sensors in environmental monitoring networks using data fusion and machine learning approach," *ICT Express*, vol. 6, no. 3, pp. 220–228, 2020, doi: 10.1016/j.icte.2020.06.004.

[79] Y. Hajjaji, W. Boulila, I. R. Farah, I. Romdhani, and A. Hussain, "Big data and IoT-based applications in smart environments: A systematic review," *Computer Science Review*, vol. 39, 2021, doi: 10.1016/j.cosrev.2020.100318.

[80] P. Paudel, S. Kim, S. Park, and K. H. Choi, "A context-aware IoT and deep-learning-based smart classroom for controlling demand and supply of power load," *Electronics*, vol. 9, no. 6, 2020, doi: 10.3390/electronics9061039.