



Research

Seamless real-time thermal imaging system with ESP8266: wireless data transfer and display using UDP



Md. Abdul Muttalib Moon¹ · Md. Kaimujjaman² · Md. Mahabub Hossain¹ · Md. Mehedi Islam¹ · Md. Selim Hossain¹

Received: 1 August 2023 / Accepted: 16 October 2023

Published online: 23 October 2023

© The Author(s) 2023 [OPEN](#)

Abstract

Thermal imaging technology has become increasingly popular for various applications, including industrial monitoring, building automation, and medical diagnostics. However, existing thermal imaging systems often come with high costs and limited connectivity options. In this paper, we propose a method to address these challenges by utilizing the ESP8266 microcontroller to create a thermal imaging system that can measure thermal pixel values, transfer the data wirelessly using the ESP8266's networking capabilities and display the pixel data in real-time on a Thin-film-transistor liquid-crystal (TFT) display. The objective is to establish a seamless and real-time transfer of thermal images within a local network environment. User datagram protocol (UDP) supports transmission via broadcast and multicast, making it highly efficient for delivering data to multiple clients or devices on a network. It allows a single UDP packet to be simultaneously sent to multiple destinations, enhancing its effectiveness. This feature simplifies the implementation of network protocols and applications, reducing their overall complexity. UDP is particularly well-suited for devices with limited resources, such as microcontrollers or embedded systems, where memory and computing power are constrained. Experimental results demonstrate the successful transmission and display of thermal pixel data between the ESP8266 microcontrollers using the UDP protocol. The project utilizes the Arduino framework along with ESP8266WiFi and UDP libraries to enable network connectivity and UDP communication. The sender and receiver devices are connected to the same local network, guaranteeing efficient and low-latency transmission of thermal pixel data. The system achieves real-time communication within a radius of approximately 15–18 m, ensuring immediate visualization of thermal images on connected displays. By minimizing latency, the system enables a seamless and instantaneous viewing experience offering seamless and instantaneous image visualization for the users.

Article highlights

- A proposed method that overcomes the challenges of high cost and limited connectivity issues of exiting thermal imaging system.
- Arduino based framework along with ESP8266WiFi and UDP libraries to enable network connectivity and UDP communication.
- Minimize latency, and seamless and instantaneous image visualization.

Keywords UDP protocol · Thermal camera · ESP8266 · TFT display · Thermal imaging

✉ Md. Mahabub Hossain, im.mahabub@gmail.com | ¹Department of Electronics and Communication Engineering, Hajee Mohammad Danesh Science and Technology University, Dinajpur 5200, Bangladesh. ²Department of Software Engineering, Daffodil International University, Dhaka, Bangladesh.



SN Applied Sciences

(2023) 5:296

| <https://doi.org/10.1007/s42452-023-05529-y>

SN Applied Sciences
A **SPRINGER NATURE** journal

1 Introduction

Wireless communication is the transmission of information or data between devices using electromagnetic waves, rather than physical wires. Wi-Fi operates at the physical and link layers, serving as an interface for wireless communication. Within the wireless communication framework, UDP/Internet Protocol (IP) plays a crucial role as one of the layers associated with the physical and data link layers. UDP is a transport layer protocol that enables the exchange of datagrams between devices over an IP network. UDP is often utilized in scenarios where real-time data transmission is critical, such as video streaming, voice over IP (VoIP), and online gaming. In response to the growing demand for grid technology and high-performance computing, novel UDP-based protocols have emerged to cater to the requirements of high bandwidth and large delay networks in recent years [1]. These UDP-based protocols use a hybrid approach that combines the reliability of Transmission Control Protocol (TCP) with the speed of UDP. This approach has several advantages, including improved reliability, reduced overhead, and increased bandwidth utilization [2]. This protocol operates in a connectionless manner, allowing data to be transmitted without the need to establish a prior connection between the transmitter and receiver. Data packets, known as datagrams, are instead transferred from the sender to the recipient without any assurance of delivery or order according to the “fire-and-forget” approach. UDP operates without establishing an end-to-end connection with the remote UDP module. Instead, it transmits datagrams directly into the network and accepts incoming datagrams from the network without requiring any prior connection setup [3]. It reduces transmission delay by avoiding away with the requirement for connection establishment, recognition, and repetition procedures [4]. In contrast, a specialized imaging technique called thermography, commonly referred to as infrared imaging, captures and identifies the electromagnetic radiation that a tangible thing emits. The thermal radiation released by a body is proportional to the disparity of the fourth power of the body and the fourth power of its surroundings, according to Boltzmann’s law, which is applied in this method. [5]. A unique lens is used in thermography for capturing the object’s infrared radiation. A thermogram, or detailed temperature pattern, is produced by scanning the concentrated light through an alternating sequence of infrared-detector components. Temperature data is acquired in just one out of thirteen of a second. The impulses, which form an amalgamation, carry information about the intensity of the infrared emission and are

transmitted to a signal-processing unit. This unit then presents the data on the display, representing it using different colors [6]. Recently, the use of thermography in biological research has been researched. Numerous applications have been identified for it, such as determining the thermal conditions necessary for burns to be therapeutically cooled [7] and identifying possible weak spots where plaque has accumulated. This identification is possible because vulnerable plaques generate heat through their inflammatory response [8].

In this research, a novel method for transmitting and receiving thermal pixel data over UDP between two ESP8266 microcontrollers and displaying the data on a TFT Display is demonstrated. Data transfer is instantaneous and low-latency since both the transmitter and receiver ESP8266 devices are linked to the same local network. The system enables real-time visualization of the captured thermal images, empowering users to gain insights and make informed decisions based on the displayed data. The applications of this study include locating people with elevated body temperatures, identifying birds and animals on a poultry farm that are experiencing heat stress, sensing of excessive hot devices and chips inside data centers or server rooms without getting too close to them.

2 Early research

The utilization of ESP8266 microcontrollers and the UDP protocol in wireless communication applications has been extensively investigated in research. However, there is a scarcity of specific literature that focuses on the transfer of thermal pixel data between ESP8266 devices using UDP and integrating it seamlessly with display systems. Despite this gap, there exists a significant body of research dedicated to UDP-based communication, thermal imaging, and display integration. These areas offer valuable insights, methodologies, and techniques that can be leveraged to enhance the development of ESP8266-based systems for real-time transfer and visualization of thermal images. Further exploration and experimentation are necessary to bridge the existing knowledge gap and enable efficient integration of thermal pixel data with UDP communication and display technologies. Nowadays The majority of the world’s developed regions are fiber-connected through national or international networks with high speeds [9, 10]. Composable UDP accommodates an extensive range of control algorithms, encompassing TCP algorithms (such as New Reno (RFC 2582), Vegas [11], FAST [12], Westwood [13], High-speed [14], BiC [15], and Scalable [16]), bulk data transfer algorithms (like SABUL [17], RBUDP [18], Lambda Stream [19], CHEETAH [20], and Hurricane [21]), as well as group transport control algorithms (e.g., CM and GTP [22]).

Yao [23] developed a method that uses the size of data packets to transmit hidden information. In this scheme, the sender and receiver share a secret matrix, where each cell represents a unique size. The sender randomly selects a cell from the matrix and uses the size of that cell to determine the size of the message to be sent. The receiver then uses the same algorithm to find the random number from the table and obtain the hidden information. In a recent development, Li Ji et al. [24] presented an enhanced algorithm that employs message lengths for secure transmission of secrets. In the proposed method, both the sender and the receiver cooperate to generate a reference by collecting specific lengths from regular network traffic. These reference values are then used to transmit secret messages. However, a drawback of this approach is that stego traffic deviates from normal network behavior. In Liping's second proposal, a novel technique is introduced, wherein "buckets" are established, each containing specific values. The number of buckets created is determined by the payload of secret data per packet. Based on the secret data, a value is chosen from one of these buckets, and the corresponding packet is transmitted across the network. Parween and Hussain [25] discussed the benefits of using UDP in IoT applications, highlighting its low overhead, low latency, and suitability for real-time data transmission. Their work emphasized the advantages of UDP in scenarios where occasional data loss is acceptable, making it a suitable choice for applications with time-sensitive data such as temperature monitoring and sensor data transmission. Bourdeau et al. [26] presented a wireless sensor network-based thermal imaging system for building energy management. Their study focused on utilizing thermal sensors and wireless communication to monitor and analyze thermal patterns in buildings. Adu-Kankam et al. [27] proposed a framework for integrating displays into IoT environments. Their research highlighted the importance of visualizing data in real-time to facilitate decision-making. Tahir et al. [28] explored the applications of ESP8266 microcontrollers in IoT-based systems. Their work discussed the capabilities and advantages of ESP8266 in terms of low power consumption, Wi-Fi connectivity, and compatibility with various sensors. While their study did not specifically focus on thermal imaging or UDP protocol, it provided valuable insights into the use of ESP8266 in IoT contexts. Gayathri [29] proposed a real-time thermal imaging system using ESP8266 for industrial applications. Their work demonstrated the integration of ESP8266 microcontrollers with thermal sensors and the UDP protocol. Tran et al. [30] presented a wireless thermal imaging system for smart agriculture using ESP8266 and the UDP protocol. The research aimed to monitor and analyze temperature distribution in agricultural fields. This work emphasized the application of ESP8266-to-ESP8266 thermal data

transfer and display integration in the context of smart agriculture. The research literature demonstrated innovations in thermal data transfer between ESP8266 devices using the UDP protocol. The studies illustrated that many sectors, such as industrial monitoring, indoor temperature monitoring, smart agriculture, and remote temperature monitoring, achieved successful implementation. The results of this research aid in the comprehension and real-world application of ESP8266-based temperature monitoring systems and offer insightful information for additional research and expansion in this area.

3 Materials and methods with models

A method for implementing a thermal imaging system using an ESP8266 microcontroller system is designed to measure thermal pixel values and transmit the pixel data to another ESP8266 module. The receiving ESP8266 module processes the received pixel data and displays it on a TFT display in real time.

3.1 Materials

The following are the primary materials we used to do the research and create the system:

- Two ESP8266 development boards as central units of transmitting and receiving devices.
- AMG8833 thermal camera for image capturing of diverse objects.
- TFT display for image visualization.
- 5 V DC power supply units to provide power to the transmitting and receiving modules.
- Connecting wires for interconnection among the devices of transmitting and receiving modules.
- Arduino Integrated Development Environment (IDE) for embedded programming.
- Arduino serial monitor for data visualization.

3.2 Block diagram

The block diagram of the system is presented in Fig. 1. The proposed system consists of two main components: the thermal camera and the display module but the Microcontrollers are the heart of the proposed system. The thermal camera module incorporates a thermal sensor, such as the AMG8833, to capture the temperature distribution and convert it into thermal pixel data, and two power supply units are used to activate these devices.

The thermal camera, which has a 64-pixel (8×8 matrix) resolution, is connected to the Microcontroller-1 (ESP8266) using the I2C bus. The I2C bus is a synchronous serial

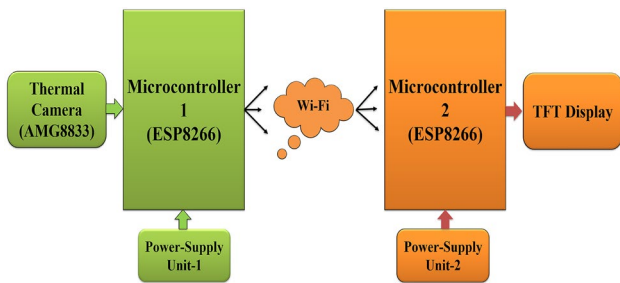


Fig. 1 Block diagram of the system

communication protocol that is commonly used to connect peripherals to microcontrollers. In this case, the I2C bus will be used to transfer the thermal pixel data from the camera to the ESP8266. The ESP8266 is responsible for reading the pixel data from the thermal sensor and establishing a wireless connection with another ESP8266 module. Once the thermal pixel data is acquired by the ESP8266, it is encapsulated into UDP packets for efficient transmission. The Microcontroller-1(ESP8266) module acts as a UDP client that connects to the Microcontroller-2(ESP8266) module, which acts as a UDP server. The client module sends the pixel data packets over the wireless network using the UDP protocol. The server module receives the packets and reconstructs the pixel data for further processing. The server module processes the received pixel data and uses the SPI protocol to interface with a TFT display that has a resolution of 240 × 320 dots (RGB) to visualize the thermal image.

The interpolation method implemented in the code was bicubic interpolation, which was used to upscale data from an 8 × 8 pixel grid obtained from the AMG8833 thermal camera to a higher-resolution 24 × 32 pixel grid for display on the Adafruit_ILI9341 TFT screen (240 × 320 pixels). Bicubic interpolation involved several steps: first, it calculated scaling factors (μ_x and μ_y) based on the source and destination grid dimensions. Then, it iterated through the destination grid, determined the corresponding coordinates in the source grid, and retrieved a 4 × 4 matrix of adjacent data points using `get_adjacents_2d`. Next, it calculated fractional coordinates within this 4 × 4 matrix ($frac_x$ and $frac_y$) to estimate the value at the desired position. This was done by applying cubic interpolation both horizontally and vertically using `cubic-Interpolate` and `bicubic-Interpolate`, resulting in the final interpolated value. This process was repeated for each pixel in the destination grid, effectively enhancing the thermal image's resolution for a smoother and more detailed display on the TFT screen.

After the pixel data has been successfully interpolated, it is further translated into corresponding color intensities or temperature values. These values are then mapped onto

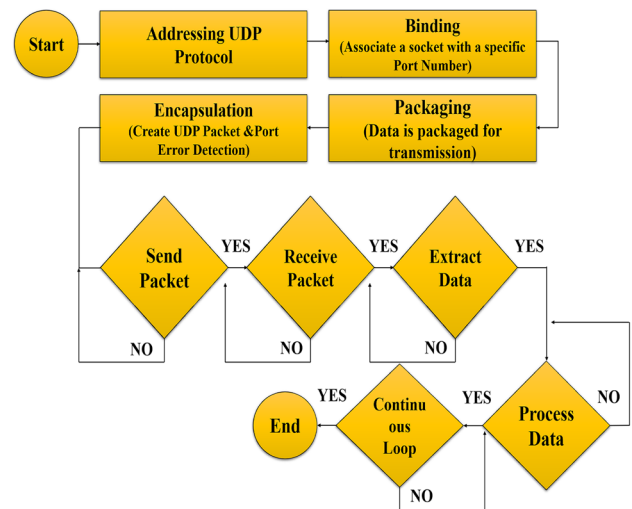


Fig. 2 Flowchart of UDP protocol

the individual pixels of the TFT display. The display module, which is connected to the ESP8266 server, performs real-time updates to the TFT display. This ensures that the thermal image is continuously refreshed, providing immediate and dynamic visual feedback to the viewer.

3.3 System flowchart

The flowchart illustrates the process of capturing thermal images using thermal camera module, transmitting the pixel data to the ESP8266 microcontroller, and visualizing it on a TFT display.

The UDP Protocol's main attributes are discussed by the flowchart in Fig. 2. The protocol begins by addressing the data to be sent. Then, the sender binds the socket to a specific port. After that, the data is packaged and prepared for transmission. The sender encapsulates the data into a UDP datagram, including source and destination port numbers, and sends it to the intended receiver's IP address. UDP is a connectionless protocol, which means that there is no guarantee that the data will be delivered successfully. Consequently, when a UDP packet is re-sent, the data is not re-packed. If the data does not get delivered, the sender will simply keep trying to send it until it succeeds or until it gives up. The receiver continuously extracts and processes the data from the sender UDP datagrams that has received. To avoid endless loops, the sender implements a timeout mechanism, which stops sending after a certain period without acknowledgment. The timeout value is carefully chosen to balance delivery chances and endless loop prevention. The sender can also implement an exponential back off algorithm to reduce the chances of getting stuck in an endless loop. This means that the sender will not wait for a longer and longer period between attempts to

send the packet if the previous attempt failed. This helps to reduce the chances of the sender getting stuck in an endless loop.

The flowchart in Fig. 3 explains the main functionality of the client program in the ESP8266 microcontroller. The program first connects to a Wi-Fi network. The application reads the object temperature from a thermal camera as pixel data after the connection has been established. The pixel data is then converted into packet data, which is a series of bytes that can be easily transmitted over a network. The packet data is then sent to another microcontroller using UDP protocol over Wi-Fi.

The flowchart depicted in Fig. 4 provides a comprehensive overview of the primary operations executed by the Server program on another ESP8266 microcontroller. The program initiates by creating a Wi-Fi network and establishing a connection with the client microcontroller. Whenever the connection is established effectively, the program proceeds to receive packet data transmitted by the client microcontroller. The received packet data is then processed and converted back into pixel data, which corresponds to temperature information. Subsequently, the pixel data undergoes further processing and manipulation to generate a visual representation known as Thermal Imaging.

3.4 Software design

The ESP8266 board is programmed in embedded C++ to interface sensors and modules. The Arduino IDE is free and open-source platform that can be used to write and debug the program. After being uploaded the program, the microcontroller automatically collects thermal values from the thermal camera at regular intervals. The microcontroller then sends the thermal values to another microcontroller and displays the thermal values as numerical values in the form of thermal imaging on the display.

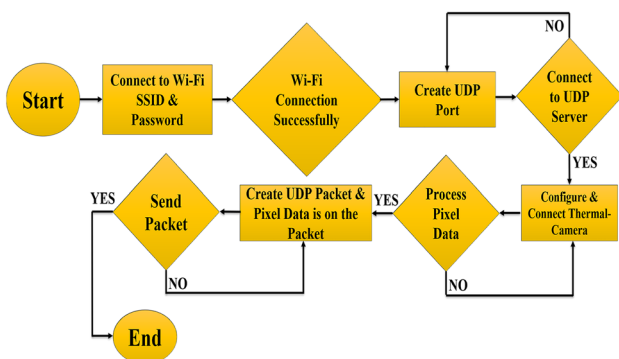


Fig. 3 Client flowchart of the system

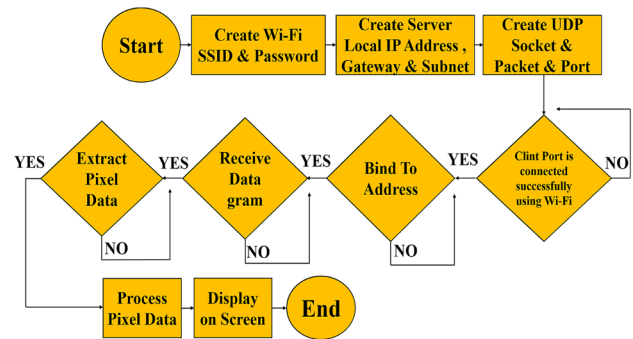


Fig. 4 Server flowchart of the system

3.5 Hardware implementation

Figure 5 indicates the interconnection of all the devices in the project. The block diagram shows that all of the

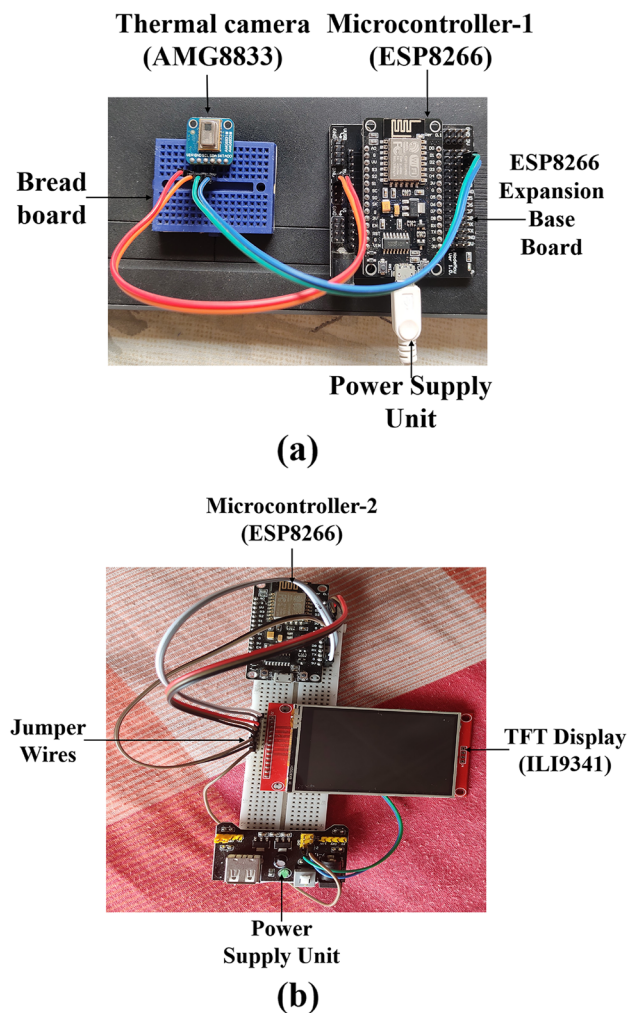


Fig. 5 Hardware implementation setup: **a** transmitting device, **b** receiving device

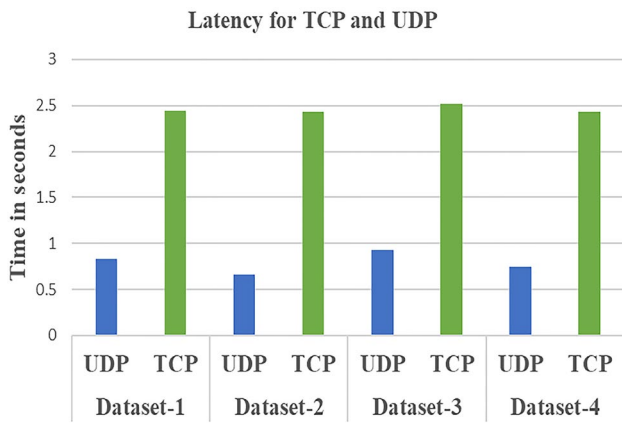


Fig. 6 Latency for TCP and UDP

system's devices are connected and that they are all powered by a 12 V-2 A battery. Figure 5a shows a transmitting device as a client using a thermal camera to capture thermal images of an object. The camera converts the infrared radiation emitted by the scene into an electrical signal, which is then digitized and sent to the server. Conversely, Fig. 5b represents a receiving device as a server that receives and decodes the signals and also creates a thermal image of the scene.

4 Result and discussion

In this section, we will first provide a latency bar chart for TCP and UDP. Then we show how to establish a connection between a server and a client, as well as how to send and receive data between them. Finally, we demonstrate how to identify cold and hot objects in front of a thermal camera.

Figure 6 illustrates the latency results for TCP and UDP. The TCP protocol exhibits a pixel data transmission time from the client to the server within the range of 2.358 to 2.513 s. In contrast, the UDP protocol achieves a significantly faster pixel data show time, taking only 0.598 to 0.922 s.

Figure 7, 8 and 9 demonstrate the serial monitor of the client, whereas Fig. 7 shows the communication link between a server and a client, showcasing the vital connection that enables data exchange between the two entities. The establishment of this connection is achieved within a range of 15 to 18 m, and the entire process typically takes around 30 s. However, if the distance between the server and client is more than 18.28 m, distortion occurs, that can lead to data transfer issues. The aforementioned UDP ports are specifically utilized for facilitating the transfer of the packets. Figure 8 illustrates the collection

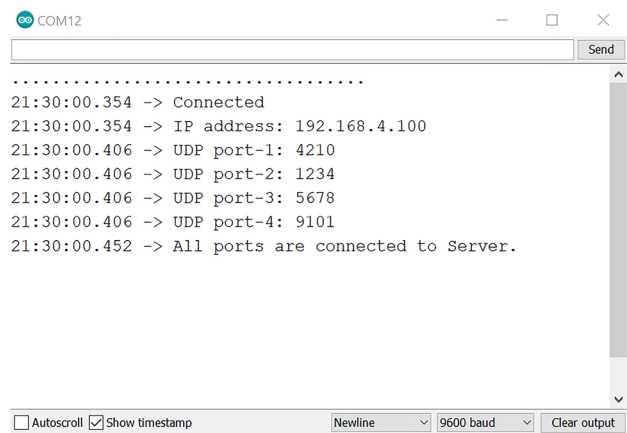


Fig. 7 Client and server connection

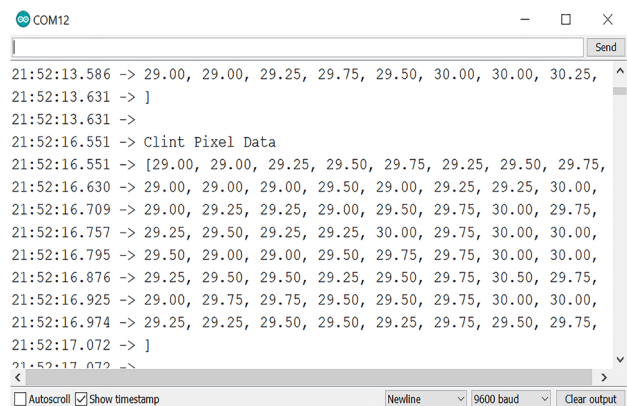


Fig. 8 Client pixel data

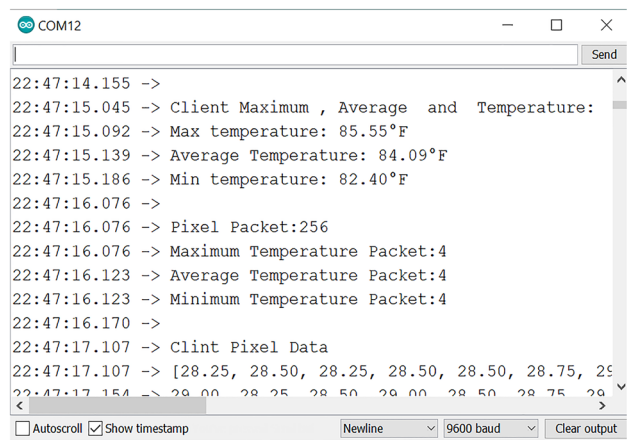


Fig. 9 Client thermal data and packets

of thermal pixel data presented in degrees Celsius. The acquired data is then processed to get three distinct value sets: maximum, average and minimum and these

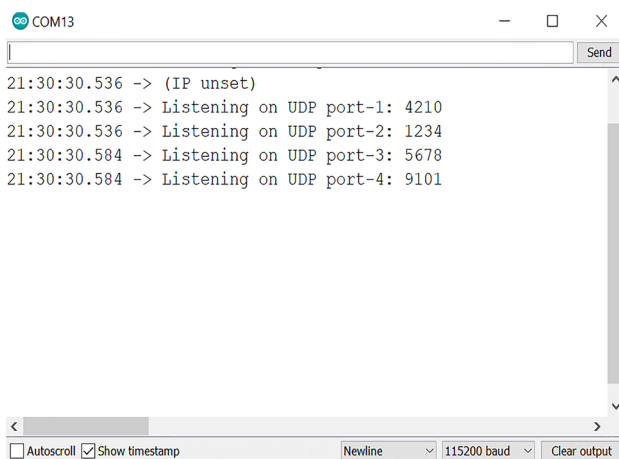
processed values are expressed using Fahrenheit scale and four UDP packets are generated to send those pixel value, maximum value, average value and minimum value to the server through Wi-Fi which are conveyed in Fig. 9. Each packet is sent to a different port, which allows for the efficient transfer of data between the client and the server.

The server's serial monitors are pictured in Figs. 10, 11 and 12 whereas Fig. 10 displays that using UDP in conjunction with four different ports offers a dependable and effective method of transferring data between the client and the server. According to Fig. 11, the client can transmit the related data (pixel value, maximum value, average value, and minimum value) to the server in 2.129–2.427 s. Figure 12 shows that data interpolation from the server to the display adds 114–116 milliseconds to the data conversion time from digital to visual format. The interpolation technique smooths the data and makes it appear more continuous.

The measurement of a cold object's temperature is appeared in Fig. 13. The cold object is depicted in Fig. 13a, whereas in Fig. 13b, it appears as a blue area within the image. This is because the object is emitting less infrared radiation than the surrounding objects. The temperature of the object is displayed as 37.85 °F, which is a relatively cold temperature.

Figure 14 expresses the measuring of a hot object's temperature. Figure 14a reveals the hot object, and Fig. 14b reflects the hot object as a red region within the image. The object emits a higher level of infrared radiation compared to the surrounding objects, resulting in its distinctive coloration. The temperature of the object is displayed as 183.20 °F, indicating a relatively high temperature.

Figure 15 represents the measurement of fingers temperature. In Fig. 15a, the fingers are visible, while Fig. 15b exposes a distinct blue region within the finger image.

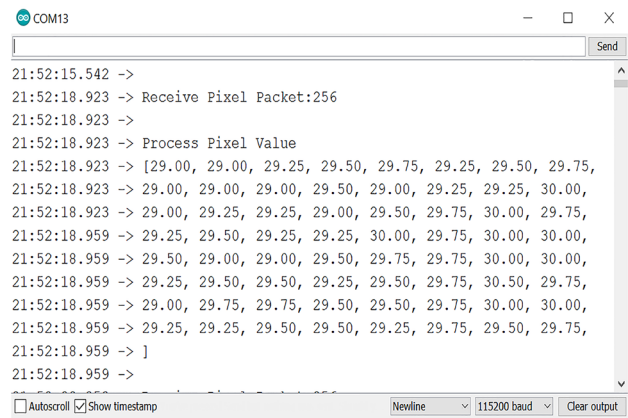


```

COM13
21:30:30.536 -> (IP unset)
21:30:30.536 -> Listening on UDP port-1: 4210
21:30:30.536 -> Listening on UDP port-2: 1234
21:30:30.584 -> Listening on UDP port-3: 5678
21:30:30.584 -> Listening on UDP port-4: 9101
Autoscroll Show timestamp Newline 115200 baud Clear output

```

Fig. 10 Listening UDP port on server



```

COM13
21:52:15.542 ->
21:52:18.923 -> Receive Pixel Packet:256
21:52:18.923 ->
21:52:18.923 -> Process Pixel Value
21:52:18.923 -> [29.00, 29.00, 29.25, 29.50, 29.75, 29.25, 29.50, 29.75,
21:52:18.923 -> 29.00, 29.00, 29.00, 29.50, 29.00, 29.25, 29.25, 30.00,
21:52:18.923 -> 29.00, 29.25, 29.25, 29.00, 29.50, 29.75, 30.00, 29.75,
21:52:18.959 -> 29.25, 29.50, 29.25, 29.25, 30.00, 29.75, 30.00, 30.00,
21:52:18.959 -> 29.50, 29.00, 29.00, 29.50, 29.75, 29.75, 30.00, 30.00,
21:52:18.959 -> 29.25, 29.50, 29.50, 29.25, 29.50, 29.75, 30.50, 29.75,
21:52:18.959 -> 29.00, 29.75, 29.75, 29.50, 29.50, 29.75, 30.00, 30.00,
21:52:18.959 -> 29.25, 29.25, 29.50, 29.50, 29.25, 29.75, 29.50, 29.75,
21:52:18.959 -> ]
21:52:18.959 ->
Autoscroll Show timestamp Newline 115200 baud Clear output

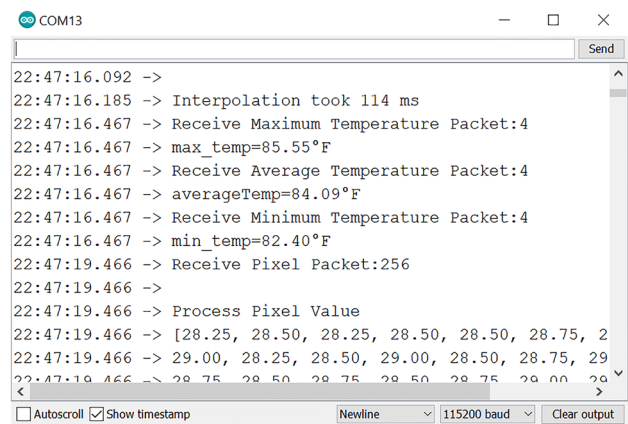
```

Fig. 11 Server pixel data

This blue hue signifies that the fingers were emitted less infrared radiation than their surrounding objects. The displayed temperature of the fingers was recorded at 24.00 °C, indicating a relatively low temperature.

5 Conclusion

In this study we developed a wireless client-server architecture to facilitate exchange of thermal pixel data. The server and client devices use UDP to transmit thermal pixel data with lower latency than TCP, which takes 0.598 to 0.922 s. The time discrepancy between the clients' and servers' all data (pixel value, maximum value, average value, and minimum value) is 2.129–2.427 s and a communication range of between 15 and 18 m. The data were processed to derive the maximum, average, and minimum temperatures, which were then converted to Fahrenheit and transmitted using a UDP packet over Wi-Fi connectivity. The server decoded and displayed the received values,

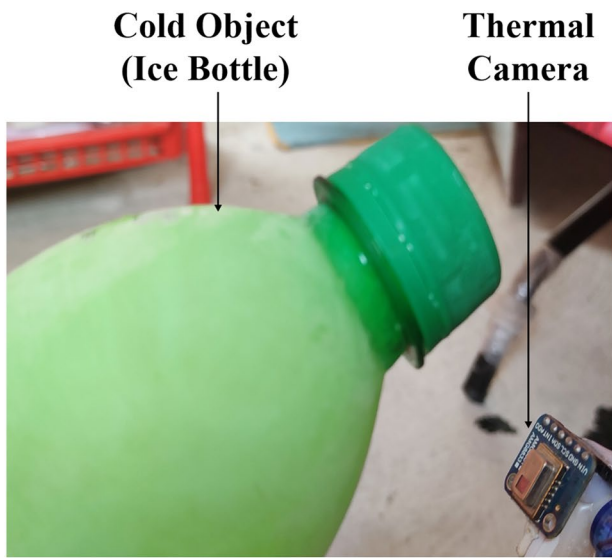


```

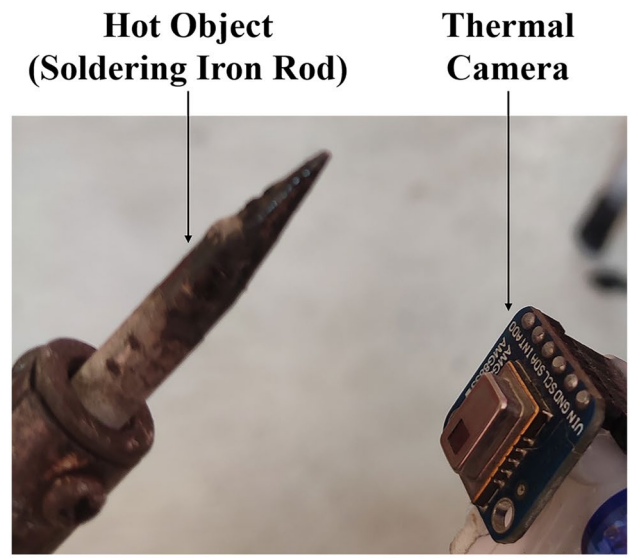
COM13
22:47:16.092 ->
22:47:16.185 -> Interpolation took 114 ms
22:47:16.467 -> Receive Maximum Temperature Packet:4
22:47:16.467 -> max_temp=85.55°F
22:47:16.467 -> Receive Average Temperature Packet:4
22:47:16.467 -> averageTemp=84.09°F
22:47:16.467 -> Receive Minimum Temperature Packet:4
22:47:16.467 -> min_temp=82.40°F
22:47:19.466 -> Receive Pixel Packet:256
22:47:19.466 ->
22:47:19.466 -> Process Pixel Value
22:47:19.466 -> [28.25, 28.50, 28.25, 28.50, 28.50, 28.75, 2
22:47:19.466 -> 29.00, 28.25, 28.50, 29.00, 28.50, 28.75, 29
22:47:19.466 -> 28.75, 28.50, 28.75, 28.50, 28.75, 29.00, 29
Autoscroll Show timestamp Newline 115200 baud Clear output

```

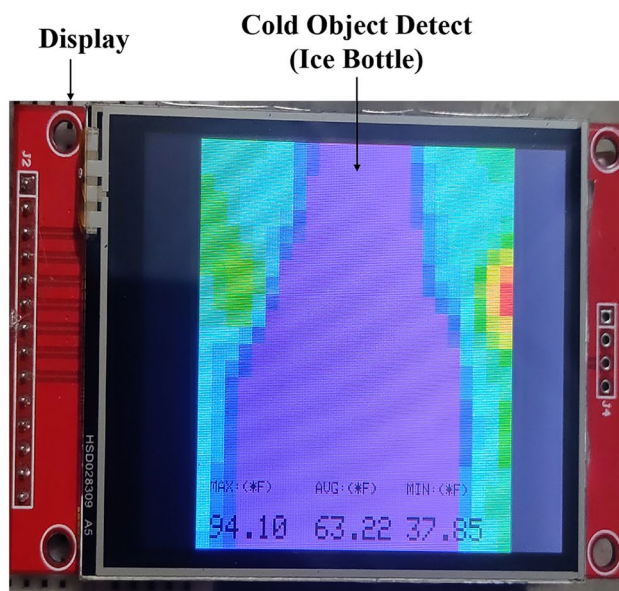
Fig. 12 Server thermal data



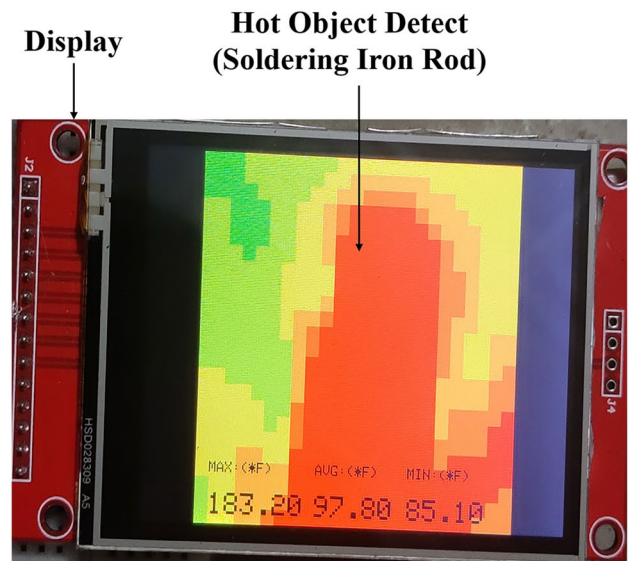
(a)



(a)



(b)



(b)

Fig. 13 Measuring temperature: **a** cold object in front of thermal camera, **b** Detected cold object and displayed

Fig. 14 Measuring temperature **a** hot object in front of thermal camera, **b** detected hot object and displayed

enabling further analysis and processing. The system was successfully used to detect cold (37.85 °F) and hot (183.20 °F) objects in thermal images and also showed a thermal image of fingers with a recorded temperature of 24.00 °C. The color-coded representation accurately identified regions emitting lower or higher levels of infrared radiation compared to their surroundings. The corresponding temperature values provided valuable insights into the relative temperatures of the detected objects. The results of this study demonstrate the potential of the proposed

system for real-time analysis and identification of temperature variations in thermal images. For instance, the system can detect hot spots in electrical equipment, monitor patient body temperature, and map heat distribution in buildings. Future research could focus on enhancing the system using machine learning algorithms for better temperature detection accuracy and exploring different data transmission protocols to improve performance in various environments. Overall, this study demonstrates the system's potential for real-time analysis and temperature

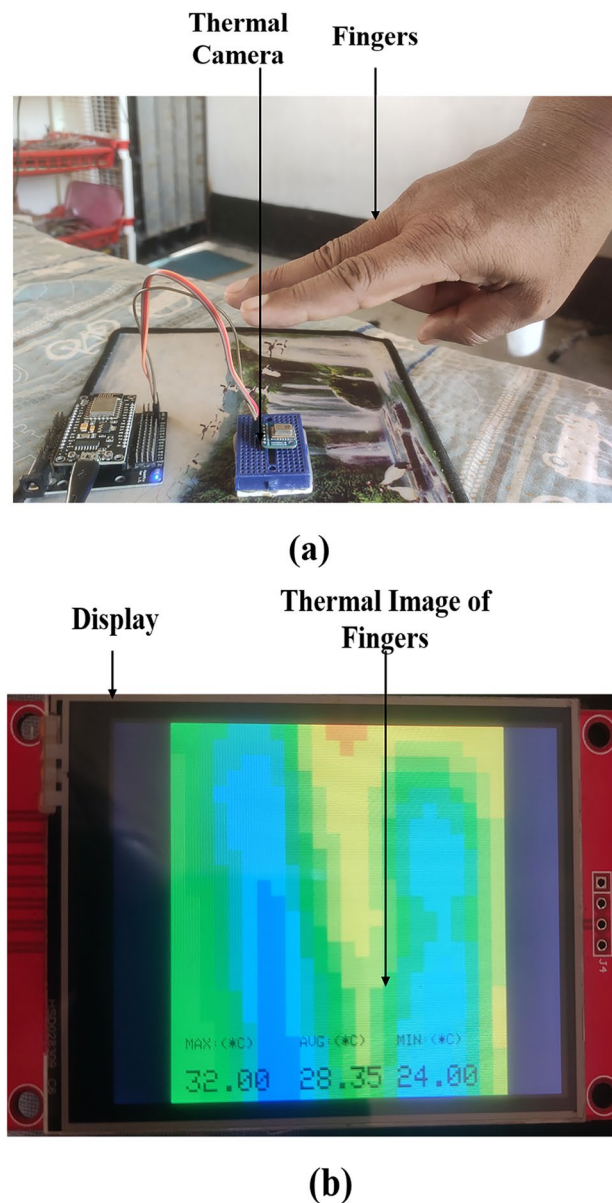


Fig. 15 Measuring temperature **a** fingers in front of thermal camera, **b** thermal imaging of fingers

variation identification in thermal images, contributing significantly to the field of thermal imaging technology.

Acknowledgements This study was supported by the Information and Communication Technology Division, Ministry of Posts, Telecommunications and Information Technology, Bangladesh.

Author contributions MMH, MAMM and MK contributed to the study conception, design and experiment discussion. Materials and methods preparation, conducted the experiment, result data collection and analysis were performed by MAMM, MK, MMI and MSH. The first draft of the manuscript was written by MAMM and MK, and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Funding The authors have not disclosed any funding.

Data availability All data generated or analyzed during this research are included in the presented Tables and Figures in this manuscript.

Declarations

Conflict of interest The authors declared that there is no conflict of interest in this research. Moreover, no funds have been received for this study.

Ethical approval It should be noted that no experiments were done involving human issues in this research.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

1. ; Zhaojuan Yue, Yongmao R, Li J (2011) Performance evaluation of UDP-based high-speed transport protocols. IEEE 2nd International Conference on Software Engineering and Service Science (ICSESS)-Beijing, China, pp. 69–73, <https://doi.org/10.1109/ICSESS.2011.5982257>
2. Wu RX, Chien AA (2004) GTP: Group transport protocol for lambda-Grids, in Proceedings of IEEE International Symposium on Cluster Computing and the Grid, pp. 228–238, <https://doi.org/10.1109/CCGrid.2004.1336572>
3. Postel J (1980) RFC 768: User Datagram protocol. <https://www.rfc-editor.org/rfc/rfc768>. Accessed Aug 28 1980
4. Liu PX, Meng M, Xiufen Y, Gu J (2002) An UDP-based protocol for Internet robots. Proceedings 4th World Congress on Intelligent Control and Automation, pp.59–65, <https://doi.org/10.1109/WCICA.2002.1022068>
5. Saxena A, Willital G (2008) Infrared thermography: experience from a decade of pediatric imaging. Eur J Pediatr 167:757–764. <https://doi.org/10.1007/s00431-007-0583-z>
6. Tyson J (2001) How Night Vision Works. How Stuff Works. <https://electronics.howstuffworks.com/gadgets/high-tech-gadgets/nightvision.htm>. Accessed 2 Oct 2023
7. Pushcar N, Sandorminsky B (1982) Cold treatment of Burns. Burns 9:101–110. [https://doi.org/10.1016/0305-4179\(82\)90056-0](https://doi.org/10.1016/0305-4179(82)90056-0)
8. Larsen PJ, Waxman S (2009) Intracoronary Thermography: utility to detect vulnerable and culprit plaques in patients with coronary artery Disease. Curr Cardiovasc Imaging Rep 2:300–306. <https://doi.org/10.1007/s12410-009-0035-4>
9. Chien A, Faber T, Falk A, Bannister J, Grossman R, Leigh J (2003) Transport protocols for high performance: Whither TCP? Commun ACM 46(11):42–49. <https://doi.org/10.1145/948383.948408>

10. Tom DeFanti C, de Laat J, Mambretti K, Neggers BS, Arnaud (2003) TransLight: a global-scale Lambda Grid for e-science. *Commun ACM* 46(11):34–41. <https://doi.org/10.1145/948383.948407>
11. Brakmo L, Peterson L, Vegas "TCP (1995) End-to-end congestion avoidance on a global internet. *IEEE J Sel Areas Communication* 13(8):1465–1480. <https://doi.org/10.1109/49.464716>
12. Jin C, Wei DX, Low SH (2004) FAST TCP: motivation, architecture, algorithms, performance. *IEEE Infocom*. <https://doi.org/10.1109/INFCOM.2004.1354670>
13. Gerla M, Sanadidi MY, Wang R, Zanella A, Casetti C, Mascolo S (2001) TCP Westwood: congestion window control using bandwidth estimation. *IEEE Globecom*. <https://doi.org/10.1109/GLOCOM.2001.965869>
14. Floyd S (2003) High Speed TCP for large congestion windows. *IETF RFC 3649 Exp Stand*. <https://doi.org/10.17487/RFC3649>
15. Xu L, Harfoush K, Rhee I (2004) Binary increase congestion control for fast long-distance networks. *IEEE Infocom*. <https://doi.org/10.1109/INFCOM.2004.1354672>
16. Kelly T (2003) Scalable TCP: improving performance in high-speed wide area networks. *ACM Comput Communication Rev* (April). <https://doi.org/10.1145/956981.956989>
17. Gu Y, Grossman R (2003) SABUL: a transport protocol for grid computing. *J Grid Comput* 1(4):377–386. <https://doi.org/10.1023/B:GRID.0000037553.18581.3b>
18. He E, Leigh J, Yu O, De Fanti TA (2002) Reliable blast UDP: predictable high-performance bulk data transfer. *IEEE Cluster Computing*. <https://doi.org/10.1109/CLUSTER.2002.1137760>
19. Xiong C, Leigh J, He E, Vishwanath V, Murata T, Renambot L, DeFanti T (2005) LambdaStream- a data transport protocol for streaming network-intensive applications over photonic networks. *Third International Workshop on Protocols for Long-Distance Networks (PFLDnet 2005)*, Lyon, France
20. Veeraraghavan M, Zheng X, Lee H, Gardner M, Feng W (2003) CHEETAH: circuit-switched high-speed end-to-end transport architecture. *Proc Opticomm 2003*. <https://doi.org/10.1117/12.533340>
21. Qishi Wu, Nageswara SV, Rao (2005) Protocol for high-speed data transport over dedicated channels. *Third International Workshop on Protocols for Long-Distance Networks (PFLDnet 2005)*, Lyon, France,
22. Wu R, Chien A (2004) GTP: group transport protocol for lambda-grids. *Proceedings of the 4th IEEE/ACM International Symposium on Cluster Computing and the Grid*, Chicago, IL, April <https://doi.org/10.1109/CCGrid.2004.1336572>
23. Quan-zhu Y, Peng Z (2008) Converting channel based on packet length. *Comput Eng* 34(3):183–185
24. Ji L, Jiang W, Dai B, Niu X (2009) A novel covert channel based on length of messages. *Information engineering and electronic commerce, international symposium on information engineering and electronic commerce* pp. 551–554. <https://doi.org/10.1109/IEEC.2009.122>
25. Parween S, Hussain SZ (2022) A Comparative analysis of CoAP based Congestion Control in IoT. *2021 4th International Conference on Recent Trends in Computer Science and Technology (ICRTCST)*, Jamshedpur, India, 2022, pp. 321–324. <https://doi.org/10.1109/ICRTCST54752.2022.9781821>
26. Bourdeau M, Waeytens J, Aouani N, Basset P, Nefzaoui Elyes (2023) A wireless sensor network for residential building energy and indoor environmental quality monitoring: design, instrumentation, data analysis and feedback. *Sensors* 23(12):5580. <https://doi.org/10.3390/s23125580>
27. Adu-Kankam KO, Camarinha-Matos LM (2022) A framework for the integration of IoT components into the household digital twins for energy communities, internet of things. *IoT through a multi-disciplinary perspective 2022*. https://doi.org/10.1007/978-3-031-18872-5_12.
28. Tahir M et al (2022) Implementation of a smart energy meter using blockchain and internet of things: a step toward energy conservation. *Front Energy Res* 10:1029113. <https://doi.org/10.3389/fenrg.2022.1029113>
29. Gayathri K (2019) Implementation of environment parameters monitoring in a manufacturing industry using IoT. *2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS)*. IEEE, India, <https://doi.org/10.1109/ICACCS.2019.8728365>
30. Tran MA, Tuan TN, Le, Tan Phuong Vo (2018) Smart-config wifi technology using ESP8266 for low-cost wireless sensor networks. *2018 International Conference on Advanced Computing and Applications (ACOMP)*. IEEE. <https://doi.org/10.1109/ACOMP.2018.00012>

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.