SMART OFFICE PEOPLE COUNTER: Wokwi-Based ESP32 IoT Simulation for Human Presence Detection

Nursena Ercan
Haliç University
Computer Engineering
mail:22092090020@ogr.halic.edu.tr

Burcunur Kılıç
Haliç University
Computer Engineering
mail:23352505910@ogr.halic.edu.tr

Mohanad Y S Alayedi
Dr. Faculty Member
Haliç University
mail:mohanadysalayedi@halic.edu.tr

***Abstract*—The primary task of this project, designed for the IoT course, is to detect human presence in a room and determine the number of people in real time. It presents the results visually and physically. Regarding the project's working steps, each frame is analyzed using a video file loaded with the YOLOv8 model. Frame analysis is performed using Python. During the analysis, efforts are made to detect humans. This work is tested using a drawn virtual line. If any person crosses this line, they are evaluated as ENTRY or EXIT, and the results are recorded. This allows the number of people to be updated instantly.**

Our next step is to share the data with IoT devices. We used the MQTT module to transmit the room occupancy data we obtained to the IoT devices. The transmitted message is sent to Node-RED via the CLOUD. Node-RED is a visual programming platform. Node-RED, which receives MQTT messages, uses the Dashboard to display the incoming occupancy information. If the number of people in the room exceeds the limit set by the user: Node-RED sends a control message.

ESP32 is a microcontroller with Wi-Fi and Bluetooth capabilities. While this entire process is running, ESP32 (Wokwi) simultaneously accesses MQTT messages coming from Node-RED. This allows it to either turn the LED on or off based on the message content. The Wokwi simulation enabled us to perform real tests on the computer screen without the need for physical devices. By combining all these technologies, we built our project step by step.

I. SYSTEM ARCHITECTURE

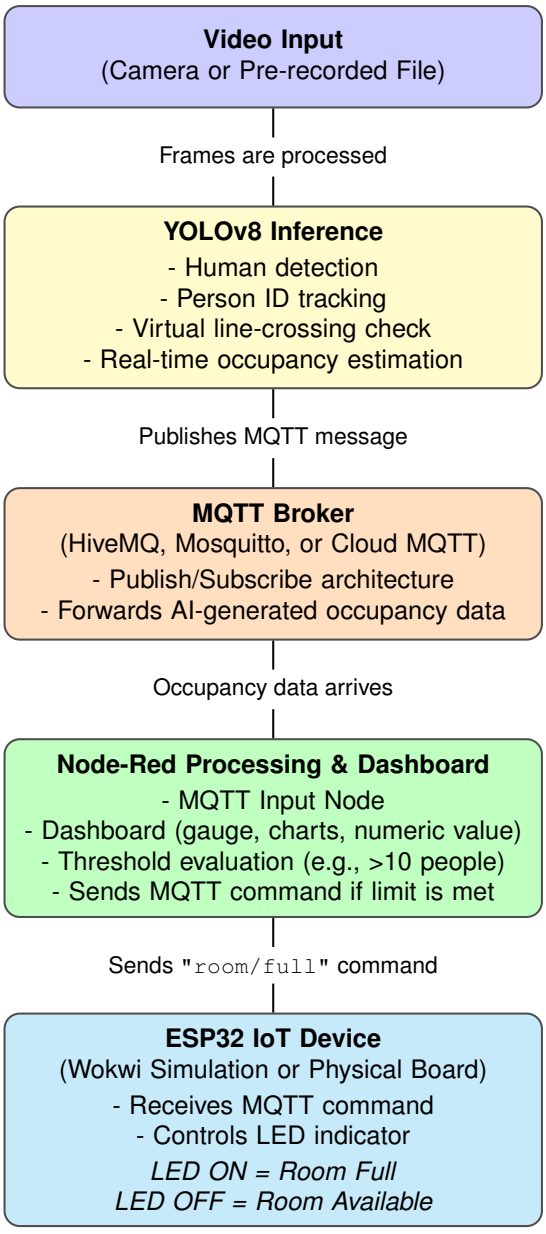

Fig. 1. System Architecture Diagram

II. INTRODUCTION

Systems for detecting human presence in an environment and determining the number of people are becoming increasingly important today. They have emerged from needs such as energy savings in smart buildings, security monitoring, automatic resource allocation, and capacity overload in enclosed spaces. To address all these needs, we researched infrared counters or turnstile systems at entrances for every environment prior to this project. However, these systems were not very successful for use in dynamic environments.

For this reason, we came up with this AI-based project that can tolerate different environments and situations to address both the needs and the negative results. In our project, we aimed to set up IoT devices in a computer environment and detect the number of people in videos we produced with AI. In this way, we developed a project suitable for the course content while also taking advantage of the latest technologies. This allowed us to discover that we could achieve successful results not only through physical means but also through simulations.

Our priority for the project was to generate data. That is, we needed to record people entering and exiting a room and note the results. Since we were building the entire project in a virtual environment, we created our own video instead of using a camera. This is where AI came into play. We produced several videos with different angles, environments, and numbers of people. This allowed us to increase our test count. We used YOLOv8 to analyze the videos we produced. YOLO (You Only Look Once): is a state-of-the-art model for object detection and image classification. Its purpose is to find all objects in an image and draw bounding boxes around them. It identifies the objects inside the boxes and extracts data related to the relevant label.

To process the video stream and utilize the visual outputs, we downloaded the OpenCV and Numpy libraries on Python to retrieve the relevant data from each analyzed frame. We then defined a virtual line on the video. We defined actions crossing the line as "ENTRY" or "EXIT." We integrated YOLO into our project to update the occupancy count after each operation. We used MQTT (Message Queuing Telemetry Transport) to send room occupancy data, updated after each check-in, to the "CLOUD" and "CONTROL SYSTEM." MQTT is a messaging-based protocol that enables two-way communication between IoT devices. It provides fast communication with IoT devices and is an ideal protocol for continuous data transfer. As a result, the updated data is transferred to Node-RED.

Node-RED is a fundamental technology for creating visual flowcharts and controlling IoT. We also took advantage of Node-RED's visualization feature to display messages received from MQTT on the Dashboard. This means that the room occupancy count is updated with each message. If the previously specified occupancy limit is exceeded, Node-RED sends a message to the ESP32 to turn on the LED and generates an alert. This is where the ESP32 simulation comes into play. The ESP32 Wokwi simulation we added to physically alert the observer turns on the LED when the occupancy limit is exceeded. This information comes from Node-Red via the MQTT protocol. The LED turns on and off based on the content of the incoming message. This way, the observer receives a physical alert.

To summarize, we created people visually with "VIDEO" and generated real data flow with simulation. We detected human presence and performed counting with "YOLOv8". We transmitted the data to the cloud with 'MQTT'. We provided visualization and IoT control on the Dashboard with "Node-RED". We created a real-world simulation that provides physical alerts using "ESP32 (Wokwi)". This section is explained in more detail in the Proposed Methods section.

III. RELATED WORKS

We worked in five key areas in our project. These are:

A. Video-based Human Detection (Deep Learning):

- YOLOv3 is an important technology that creates bounding boxes for each object, makes predictions about them, and performs real-time object detection. It was described by Redmon Farhadi in 2018. Source: J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," arXiv:1804.02767, 2018.
- YOLOv4 is a real-time object detection model developed by Bochkovskiy et al. in the year 2020. Unlike others, it can be utilized for the management of independent processes outside the recommendation systems. Source: A. Bochkovskiy, C.-Y. Wang and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," arXiv:2004.10934, 2020.
- YOLOv8 (Jocher et al., 2023) is an open-source object detection technology. This particular version generates faster and more accurate data compared with other versions of the software. Source: G. Jocher, YOLOv8 Documentation, Ultralytics, 2023.

  We utilized YOLOv8 for this project as it was very fast and accurate.

B. Counting With a Virtual Line:

- Khan et al. (2017) identified the direction in which moving objects on video crossed a predetermined line and performed people counting. Source: S. Khan et al., "People Counting Using Video Based Line Crossing Method," IEEE ICIEV, 2017.

- Zhang et al. (2019) demonstrated that virtual line-based entry/exit counting in crowded areas is effective for density estimation. Source: Y. Zhang et al., "Cross-Line People Counting in Crowded Scenes," Pattern Recognition Letters, 2019.

  We also utilize the same logic in this project. The counter is updated based on the crossing direction.

C. Real-time tracking with IoT and MQTT:

- Hunkeler et al. (2008) proved that MQTT was created for low-powe devices and is the best protocol to suit IoT systems. Source: H. Hunkeler, H. L. Truong, A. Stanford-Clark, "MQTT-S—A publish/subscribe protocol for Wireless Sensor Networks," IEEE COMSWARE, 2008.
- In his paper, Naik (2017) also showed the performance evaluation of MQTT with other protocols like CoAP and AMQP. From that, he was able to establish that MQTT can give better latency and reliability. Source: N. Naik, "Choice of Effective Messaging Protocols for IoT Systems: MQTT, CoAP, AMQP," IEEE IoT, 2017.

D. Cloud-based visualization with Node-RED:

- Azzam et al. (2020) have shown that Node-RED is ideal for rapid prototyping in managing real-time data flow. Source: A. Azzam et al., "Rapid IoT Application Development Using Node-RED," International Journal of Online Engineering, 2020.
- López et al. (2021), Node-RED having a low latency and easy integration for processing MQTT version. Source: J. López et al., "Using Node-RED for IoT Real-Time Dashboards," Sensors, 2021.

E. Controlling IoT devices with ESP32:

- Hunkeler et al. (2008) demonstrated that MQTT was developed for low-power devices and is the most suitable protocol for IoT systems. Source: H. Hunkeler, H. L. Truong, A. Stanford-Clark, "MQTT-S—A publish/subscribe protocol for wireless sensor networks," IEEE COMSWARE, 2008.
- In 2017, Naik demonstrated that MQTT provides lower latency and higher reliability compared to other protocols such as CoAP and AMPQ. Source: N. Naik, "Selecting Effective Messaging Protocols for IoT Systems: MQTT, CoAP, AMQP," IEEE IoT, 2017.

IV. PROBLEM STATEMENT

The basic demand of the modern intelligent building applications is to measure the room filling level in real time in indoor areas. Determination of how many people are inside is a critical need considering aspects of security, energy management, and efficiency, especially in environments characterized by heavy human traffic, such as universities, offices, conference rooms, and laboratories. Turnstiles, infrared sensors, or manual counting methods, which are used in existing systems, are both costly and insufficient for environments that require high accuracy.

Meanwhile, traditional camera-based counting methods may show low performance in mobile variable light or crowded environments. Sensor-based solutions do not carry direction information; hence, they cannot differentiate inlets from outlets, which complicates the correct measurement of fullness. Besides, the overwhelming majority of the existing solutions do not provide IoT-based cloud integration; thus, they are inadequate for instant data sharing and remote control.
Therefore, there is a need for a unified system that can detect people with high accuracy, determine the direction of entry and exit, and send real-time data to the cloud via IoT. The aim of this study is to develop a real-time, low-cost, and high-accuracy room occupancy measurement system by integrating artificial intelligence-based image processing, the MQTT communication protocol, and IoT device control. The system can control an ESP32-based device via the cloud according to the detected occupancy level, thus successfully providing device management based on human presence, one of the fundamental components of smart environment automation.

V. PROPOSED METHODS

1) Video source: Our project receives a video stream.
2) Frame analysis: YOLOv8 processes each model and detects human presence.
3) Bounding Box: A rectangle is created for all frames where humans are detected.
4) Virtual Line: This is the line where ENTRY and EXIT counts are recorded when a transition occurs.
5) Occupancy Count Update: The person count is updated to +1 for right transitions and -1 for left transitions.
6) MQTT Protocol: Data sent by the AI system is routed to the MQTT Broker.
7) Node-RED: Node-RED processes incoming MQTT data and provides a decision mechanism.
8) Dashboard: Node-RED creates visuals via the Dashboard. Updates are made in real-time.
9) ESP32: The IoT device that connects the project to the physical world.
10) Wokwi: Wokwi runs on the simulator and turns the LED on or off based on messages received from the cloud.

VI. SIMULATION RESULTS

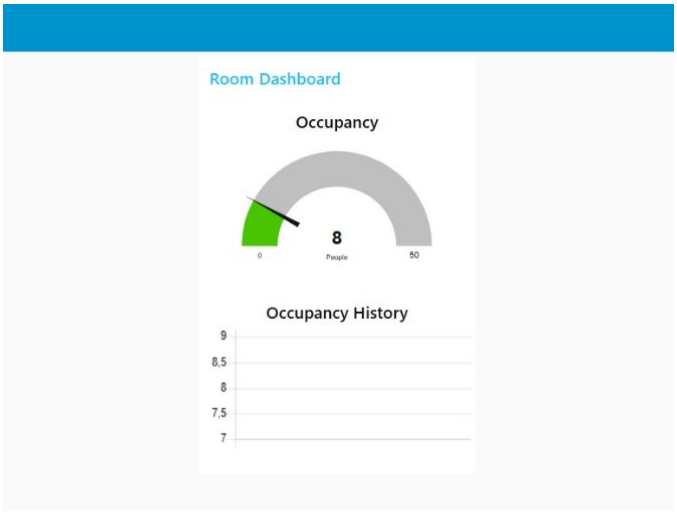

Fig. 2. Simulation Output

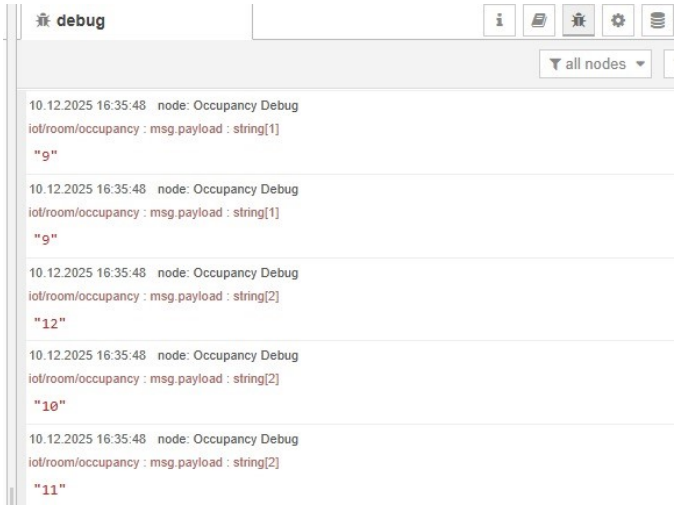

Fig. 3. MQTT Outputs1

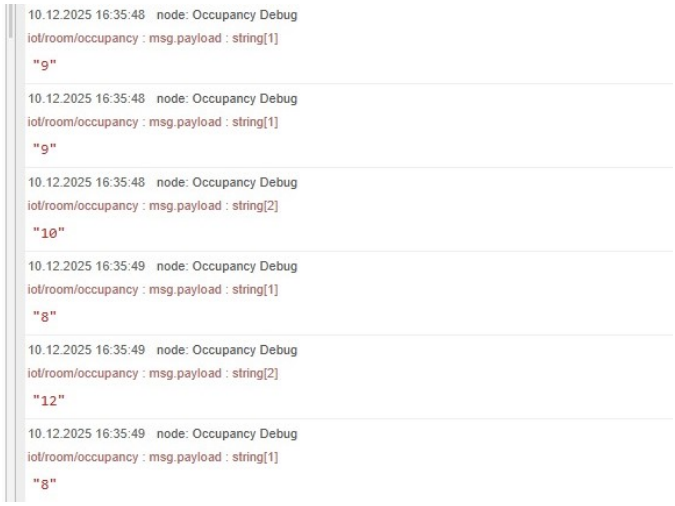

Fig. 4. MQTT Outputs2

VII. COMPARİNG OUR PROPOSAL WİTH EXİSTİNG WORKS

Work	**Proposed Methodology**	**Contributions**	**Limitations**
Physical PIR-based human detection (Li et al., 2019)	Real PIR sensor to detect motion in an office setting	Low-cost, simple implementation	Requires physical hardware; limited scalability
Ultrasonic sensor human detection (Zhang & Wang, 2020)	Real ultrasonic sensor measures distance for presence detection	Accurate distance measurement, enhances detection	Hardware required; angular blind spots exist
Wi-Fi CSI-based human activity detection (Zhang et al., 2018)	Uses Wi-Fi Channel State Information to infer human presence	Innovative approach, non-intrusive	Complex data processing; requires specialized hardware
Simulation-based IoT human detection (Kumar et al., 2021)	Software simulation using virtual sensors	Hardware-free testing, suitable for classroom projects	Simulated sensors may not perfectly reflect real-world behavior
Our proposal	Wokwi + ESP32, PIR/Ultrasonic sensor simulation, LED/Buzzer output	Fully simulation-based, IoT logic applied, easy to implement in class	No real hardware testing; simulated sensor behavior may differ from actual sensors

VIII. CONCLUSION

This paper presents the development of an integrated system that is able to perform real-time tracking of room occupancy by integrating artificial intelligence-based image processing with IoT technologies. The YOLOv8 model detected people in the video quite accurately, while reliable occupancy counts were obtained by separating entry and exit movements using virtual line logic. The occupancy data was sent to Node-RED using MQTT, a lightweight and fast communication. Here, a real-time dashboard was created, and a simple decision mechanism was implemented to send an alert when the room is full. The change in the LED status through the ESP32 Wokwi simulation effectively demonstrated the response of the system towards the physical world.

The results obtained demonstrate that camera-based people counting combined with IoT infrastructure can provide low-cost, scalable, and automatic occupancy management in areas like smart classrooms, laboratories, libraries, or offices. Further improvements may involve testing the system with real cameras, enhancing the performance in more crowded environments, and integrating additional sensors. The provided study is proof that in harmony, AI and IoT technologies form a very strong backbone for applications involving smart environments.

REFERENCES

[1] Glenn Jocher et al., *YOLOv8 Documentation*, Ultralytics, 2023. https://docs.ultralytics.com

[2] Bradski, G., *The OpenCV Library*, Dr. Dobb's Journal of Software Tools, 2000.

[3] Python Software Foundation, *Python Language Reference*, version 3.11, 2023. https://www.python.org

[4] Banks, A., Gupta, R., *MQTT Version 3.1.1*, OASIS Standard, 2014.

[5] Eclipse Paho, *Eclipse Paho MQTT Clients*, 2023. https://www.eclipse.org/paho/

[6] Node-RED, *Node-RED Documentation*, 2023. https://nodered.org/docs/

[7] Espressif Systems, *ESP32 Technical Reference Manual*, 2023. https://www.espressif.com/en/products/socs/esp32/resources

[8] Wokwi, *Online ESP32 Simulator*, 2023. https://wokwi.com/

[9] Stojmenovic, I., *Handbook of Sensor Networks: Algorithms and Architectures*, Wiley, 2005.

[10] Szeliski, R., *Computer Vision: Algorithms and Applications*, Springer, 2010.

[11] Redmon, J., Farhadi, A., *YOLOv3: An Incremental Improvement*, arXiv, 2018. https://arxiv.org/abs/1804.02767

[12] Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., Ayyash, M., *Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications*, IEEE Communications Surveys Tutorials, 2015.

[13] Bewley, A., Ge, Z., Ott, L., Ramos, F., Upcroft, B., *Simple Online and Realtime Tracking*, ICIP, 2016.

[14] Hunkeler, U., Truong, H. L., Stanford-Clark, A., *MQTT-S — A Publish/Subscribe Protocol for Wireless Sensor Networks*, 2008.

[15] Antonio, D., *Getting Started with ESP32 Development*, Packt Publishing, 2020.

[16] Few, S., *Information Dashboard Design: The Effective Visual Communication of Data*, O'Reilly Media, 2006.

[17] Shi, W., Cao, J., Zhang, Q., Li, Y., Xu, L., *Edge Computing: Vision and Challenges*, IEEE Internet of Things Journal, 2016.

[18] Lane, N. D., Bhattacharya, S., Georgiev, P., Forlivesi, C., Jiao, J., Qendro, L., Kawsar, F., *DeepX: A Software Accelerator for Low-Power Deep Learning Inference on Mobile Devices*, IPSN, 2016.

[19] Chen, L., Liang, X., *People Counting Based on Deep Learning in Video Surveillance*, Journal of Visual Communication and Image Representation, 2019.

[20] Sicari, S., Rizzardi, A., Grieco, L. A., Coen-Porisini, A., *Security, Privacy and Trust in Internet of Things: The Road Ahead*, Computer Networks, 2015.

[21] Redmon, J., Divvala, S., Girshick, R., Farhadi, A., *You Only Look Once: Unified, Real-Time Object Detection*, CVPR, 2016.

[22] Marco Schwartz, *ESP32 and MQTT Tutorial*, 2021. https://randomnerdtutorials.com/esp32-mqtt-publish-subscribe-arduino/

[23] Marquez, J., *IoT Data Visualization Using Node-RED Dashboard*, 2020.

[24] Ultralytics, *YOLOv8: State-of-the-Art Real-Time Object Detection*, 2023. https://github.com/ultralytics/ultralytics

[25] Goodfellow, I., Bengio, Y., Courville, A., *Deep Learning*, MIT Press, 2016.

[26] Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., Cayirci, E., *Wireless Sensor Networks: A Survey*, Computer Networks, 2002.

[27] Ultralytics, "Ultralytics YOLOv8 Documentation," 2023. [Online]. Available: https://docs.ultralytics.com/

[28] OpenCV Team, "OpenCV Library Documentation," OpenCV.org, 2024. [Online]. Available: https://docs.opencv.org/

[29] HiveMQ, "Public MQTT Broker and MQTT Essentials," HiveMQ, 2024. [Online]. Available: https://www.hivemq.com/mqtt/public-broker/

[30] Node-RED Foundation, "Node-RED Documentation," 2024. [Online]. Available: https://nodered.org/

[31] Wokwi, "Wokwi ESP32 Simulator Documentation," 2024. [Online]. Available: https://docs.wokwi.com/

[32] Eclipse Foundation, "Paho MQTT Python Client Library," 2023. [Online]. Available: https://www.eclipse.org/paho/

[33] EMQX, "EMQX MQTT Broker Documentation," 2024. [Online]. Available: https://www.emqx.io/docs/