Human Factors in Design, Engineering, and Computing, Vol. 238, 2026, 1612–1622
https://doi.org/10.54941/ahfe1008389

Sharing the Edge: Multiple Concurrent Neural Workloads on a Single Microcontroller in a Smart HVAC Vent

Yousef Abdou[1], Hazem El Shorbagy[2], and Alaa Abdou[3]

[1]TatweerMEA, Abu Dhabi, UAE
[2]ZenSys Corp., Houston, TX, USA
[3]Ajman University, Ajman, UAE

ABSTRACT

Edge devices increasingly run multiple inference workloads at once, but a microcontroller offers no free lunch for multitasking: every model competes for the same memory, cycles, and physical environment, including the space the user speaks into. This paper examines that competition through the second prototype of ZenSys, a smart-vent system sharing one ESP32-S3 across a wake-word detector, a keyword spotter, and continuous thermal-occupancy inference. On 2,000 labeled thermal frames, an adaptive background-subtraction blob detector outperformed a compact CNN (F1 0.92 vs. 0.80) at a fraction of the memory and latency, a result we attribute to a single-room corpus rather than to any inherent weakness of learned inference. Small early voice trials point to a similar inversion: shorter commands held up better under noise, wake-word detection appeared weaker for female speakers, and actuator vibration shrank usable voice range to 3–10 cm whenever the fan ran. For the occupant, the binding constraint was mechanical rather than computational.

Keywords: Edge AI, Embedded machine learning, Keyword spotting, Thermal occupancy sensing, Voice user interface, Human–machine interaction, Internet of Things

1 INTRODUCTION

Internet of Things (IoT) devices are taking on increasingly complex inference workloads as privacy, latency, and on-device intelligence requirements grow. Voice interaction is now common across home devices, and HVAC systems increasingly use occupancy-driven activation to reduce energy consumption. Because occupancy sensing raises privacy concerns, thermal-based detection is being explored as an alternative to camera-based approaches. These capabilities strain the microcontrollers most IoT devices run on, raising the problem of multi-model co-residency: scheduling, memory partitioning, and synchronization across concurrent workloads.

ZenSys is a multi-node house ventilation system under development to improve household air-conditioning efficiency through occupancy sensing, with voice interfacing to streamline user interaction. This paper reports the development and evaluation of ZenSys's second prototype.

Received September 27, 2026; Revised October 30, 2026; Accepted November 9, 2026; Available online December 1, 2026

© 2026 The Authors. This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License.
For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

In this smart-vent implementation, we evaluated two occupancy-detection approaches on the MLX90640: a compact CNN classifier and a lighter-weight heuristic based on temperature differencing against an adaptive background. Comparing accuracy, latency, and compute cost, we adopted the heuristic approach. For voice interaction, a quantized TFLite model performs audio matching for a custom wake-word, passing detected audio to a keyword-spotting (KWS) engine for command recognition, with firmware-level amplifier gating. Under high mechanical vibration, this pipeline performed better on shorter command words than longer ones.

The system achieved consistent, if slow, occupancy detection, and a high wake-word and keyword detection rate under quiet conditions. Under operating conditions, however, mechanical vibration substantially degraded all audio algorithms while leaving occupancy detection largely unaffected. That split matters most from the occupant's side: the silent sensing kept working, while the one channel the user touches, their voice, failed exactly when the fan was running and a command was most likely. We therefore treat interaction range, command vocabulary, and who the wake-word works for as findings, not footnotes. This paper contributes: a direct comparison of learned and heuristic occupancy inference under shared on-device constraints; evidence that acoustic-mechanical coupling, not compute, was the limiting constraint on the user's experience; early observations on how that constraint shapes command design and speaker fairness; and a reproducible single-node design for co-resident edge inference.

Section 2 reviews related work on thermal occupancy sensing, keyword spotting, and multi-model co-residency. Section 3 outlines the ZenSys concept and scope, and Section 4 covers the system architecture, hardware, firmware, occupancy inference, and voice pipeline of Prototype 2. Section 5 describes the assembly process, Section 6 discusses findings and limitations, Section 7 outlines future work, and Section 8 concludes.

2 RELATED WORK

Occupancy-driven control is one of the more promising levers for residential HVAC efficiency, and the sensing modality decides what that control can do. Cui et al. (2025) make the energy and occupant-centric case directly, arguing for thermal over RGB on privacy grounds, though at higher resolutions accuracy can be achieved with CNN detectors like YOLOv5. Closer to our hardware, Chidurala and Li (2022) characterize the MLX90640 itself for occupancy estimation, including the blob artifacts its sub-page readout produces around moving occupants, the sensor behavior our detector lives with (Section 4.3). Klir et al. (2025) meet the same background-drift problem (HVAC airflow, sun-warmed surfaces) but resolve it with a generative model filling the low-resolution gap; we instead took the drift as reason to prefer an adaptive background estimate, and their framing shaped that choice.

The literature here splits along dataset scale rather than architecture quality. The real-time thermopile occupancy system of Shubha and Shastrimath (2022) is the closest analog to what we adopted: interpolation, adaptive

thresholding, and background suppression with no training step, on an 8×8 Grid-EYE array even coarser than ours. Thermo Presence (Aszkowski and Piechocki, 2022) supplies the other half: the dataset scale a CNN needs, which a single-room corpus does not provide. Together they frame Section 4.3 as a data-regime argument rather than an anti-CNN claim: the heuristic wins here because of what our corpus is, not because learned inference is the wrong direction.

Hello Edge (Zhang et al., 2017) remains the canonical reference for small-footprint KWS and the memory-and-compute framing we designed against; TensorFlow Lite Micro (David et al., 2021) is the runtime we deployed on, and its treatment of multi-model support on MCUs informed our own partitioning. Fucci et al. (2023) ground the demographic finding discussed in Section 6: under-representation of female speakers in training data is a known, addressable defect, and their augmentation remedy is the same lever our synthetic pipeline was meant to pull; ours added volume without prosodic coverage, exactly the failure their work warns against.

Multi-model co-residency is the thinnest literature here and the clearest gap. NestDNN (Fang et al., 2018) is the reference point for multi-tenant on-device inference but assumes a mobile SoC; Quilt (Lee et al., 2026) is more recent and still targets GPU-class edge devices like the Jetson. Neither reaches MCU scale, and neither extends past the silicon to the shared enclosure, which is the space this paper occupies: two neural workloads, wake-word and command, co-resident on a single dual-core ESP32-S3, sharing a physical housing with an actuator whose vibration bounds the system, and the person talking to it, more tightly than any compute constraint.

3 THE ZENSYS SYSTEM: CONCEPT AND SCOPE

The core concept of ZenSys is that each vent is a distributed control node, capable of communicating with other nodes and using local sensing, control, and actuation for room-level airflow management. This addresses a limitation of conventional residential HVAC: a single, centrally located thermostat represents the whole house's thermal condition with one measurement and one setpoint, while occupants experience comfort room by room, so conditioned air is commonly delivered to spaces whose thermal needs or occupancy differ. This distributed-zone-control architecture is protected by issued U.S. Patent Nos. 12,584,650 ("Distributed Zone Control System") and 12,613,053. They are cited only to establish origin and context, not to interpret claim scope.

Prototype 1 established the feasibility of local voice control and reversible airflow actuation. Prototype 2, the experimental vehicle studied here, consolidates voice interaction, thermal occupancy inference, temperature sensing, and actuator control on a single ESP32-S3 vent node. Prototype 3 is intended to incorporate P2's lessons, particularly the need to physically decouple sensing and user-interface functions from noisy mechanical actuation. This paper is limited to the design and behavior of one P2 node; it does not evaluate the multi-node coordination layer, and makes no claim regarding whole-house energy savings.

4 PROTOTYPE DESIGN AND DEVELOPMENT

The objective of the ZenSys vent subsystem was to push control to the very edge of the air-conditioning system: the node interacts with the occupant for control, reports room temperature, detects occupancy, and drives a fan to increase or decrease airflow accordingly. In the full system, it also coordinates with other vents by adjusting the direction of fan rotation.

Prototype 1 implemented only voice and actuation. Its voice stack relied on a Chinese-language vendor framework, so replies came in a Chinese AI voice with no control over customization, and it split the work across an Arduino for control and a dedicated ESP32 for voice, which made it more expensive. Its multi-chamber enclosure also, coincidentally, lessened the vibration reaching the mic.

Prototype 2 set out to fix these issues: thermal occupancy sensing, an English, customizable reply voice, one MCU, and multiple fans instead of one. Interaction follows a simple loop: on power-up the node takes a temperature measurement, then rotates the blades forward or in reverse depending on whether the setpoint is below or above it. The user says the wake-word to change the setpoint or turn the unit off, and the vent confirms every command with a spoken reply. With inter-vent communication out of scope, this is the default control mode.

4.1 Basic System Architecture

A basic architecture for this kind of system splits into input, control, and output (Figure 1). Input is an IR thermal sensor and an audio sensor, feeding a microcontroller unit (MCU) that performs all logic and data processing and drives the outputs: audio-visual feedback for user interaction, and mechanical control signals to a fan unit.

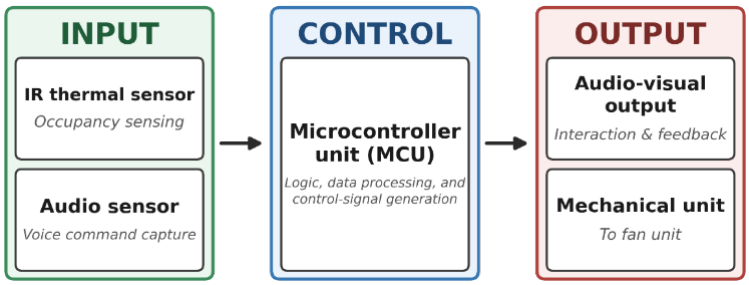


Figure 1: Basic prototype system architecture.

4.2 Detailed System Architecture and Component Selection

4.2.1 Hardware

For sensing (the audio sensor and IR thermal sensor), we needed a mic for voice and a thermal camera or thermal array for our occupancy detection approach. For the mic, we first tested an LM393-based analog sound sensor module since it is cheaper, but electrical noise from other components heavily

affected it because it relied on comparator-based sensing and a direct analog signal. We then went with the INMP441 I2S microphone, which, being digital, allowed for a much clearer signal relative to electrical noise and proved to have a greater range, confirmed by recording audio from up to 6 meters away. For occupancy detection, we went straight for an MLX90640 thermal array over I2C. The low resolution made inference over it cheaper and gave the occupant more privacy than a higher-resolution thermal camera or an RGB camera.

Given the system objective described earlier, the computational needs guided the selection of the MCU. The main compute unit was an ESP32-S3-DevKitC-1 (WROOM-1-N16R8), dual-core Xtensa LX7, 16 MB flash, 8 MB octal PSRAM. The neural workload for the wake-word model, the audio buffer ring, and the LVGL display confirmed the need for PSRAM. We further required at least two independent I2S peripherals and multiple GPIOs, all of which the ESP32-S3 satisfied. For ease of development, the DevKit was socketed onto a carrier PCB to simplify design. Figure 2 describes in detail the component selection, dataflow, and system architecture.

User interaction with the device is through a display and voice replies. The display chosen was an ST7735 128×128 SPI display, providing RGB color, paired with an LVGL interface to present the temperature setting and control information in a more aesthetic manner. For the audio interfacing, multiple strategies were tested: first, a PWM analog amp with an RC filter, which proved more susceptible to electrical noise; the option that ultimately performed well was the MAX98357A I2S Class-D amplifier, connected to a 4-ohm, 8-watt speaker.

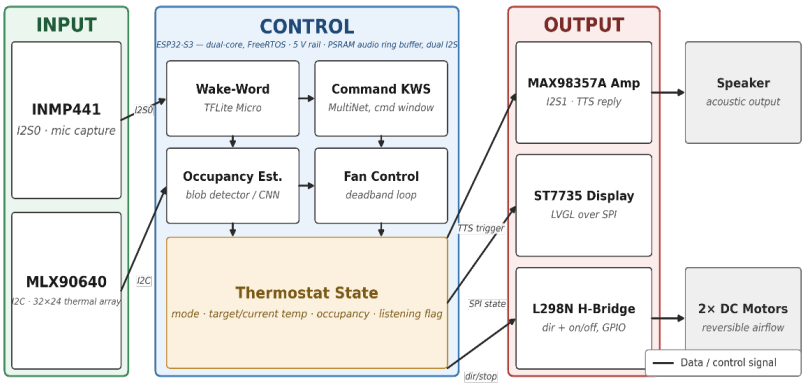


Figure 2: ZenSys P2 node detailed system architecture.

Actuation requirements were fixed from the start, considering that the vent had to move air in either direction, ruling out an ordinary single-direction axial duct fan. We compared a few ways to get that reversal, but an issue we faced regardless was the loss in airflow efficiency with direction reversal, since a fixed-pitch blade moves considerably less air in the direction it was not shaped for. We analyzed the option of an off-the-shelf bi-directional ESC to drive a BLDC fan, but that track was rejected due to financial cost relative

to other options. To decrease cost, we went with the direct option of two reversible brushed DC motors on an L298N dual H-bridge, one channel each, direction reversed by swapping GPIO polarity.

Lastly, the power circuit was designed to operate while plugged in, so no battery was added. Given the different loads from actuation, sensing, and compute, the rail structure was split into three: a 12 V input adapter to a buck converter at 5 V feeding the audio amplifier and main compute, then 5 V to 3.3 V for the sensing and display components, all tied to a common ground. Protection elements around the L298N minimize flyback noise from reversing, the buck converter is protected to limit its noise on the rail, and pull-ups and decoupling capacitors sit where needed near sensor components.

4.2.2 Firmware

The tasks were split over the two cores by timing demands: the one crucial real-time task, audio processing, runs alone on one core, while all non-time-critical tasks (thermal sampling, occupancy, fan control, and display) run on the other core.

The audio pipeline works as follows: mic data goes to the PSRAM along an I2S bus, and inference constantly pulls from it to detect the wake-word. If the wake-word is detected, the next data is forwarded from the PSRAM to the keyword-spotting (KWS) algorithm; if that detects a command, it triggers a reply task that sends TTS data along a separate I2S bus to the amplifier and speaker, and triggers the desired control effect on the thermostat state through a thread-safe locked setter.

The LVGL display refreshes at a set rate and whenever the thermostat state changes. The fan control task shares that core but runs only when the control state changes; the most demanding shared workload is thermal polling and occupancy, implemented to keep the load on the shared core minimal.

4.3 Occupancy Inference

The occupancy pipeline operated on 32×24 thermal frames from the MLX90640 sampled at approximately 2 Hz, with frames exhibiting anomalous global statistics discarded prior to inference. The labeled evaluation corpus comprises 2,000 frames (900 occupied, 1,100 vacant), collected in a single room over a single season, and both approaches were scored on the same frames.

Two candidate approaches were evaluated. The first, a compact convolutional neural network, was implemented and benchmarked but ultimately not adopted: in practice it exhibited three failure modes that proved decisive. It overfit to a single room and vent geometry; it showed sensitivity to background thermal drift from HVAC cycling and sun-warmed surfaces that a dataset of this size does not capture; and it competed with the wake-word detector for the same inference budget while targeting a signal that is, at this resolution, comparatively simple: a single warm region against a slowly drifting background.

The second approach, an adaptive background-subtraction blob detector, was adopted instead. It maintains a slowly adapting per-pixel background estimate, computes a frame-minus-background delta that is thresholded into a foreground mask, groups foreground pixels into connected components, and applies a size gate calibrated to an adult torso at the working distance (approximately 130–220 px), which sits well above the 6–34 px residual blobs produced by furniture and flooring. The adaptation rate and on/off frame-count hysteresis were tuned so that the background converges on a stationary occupant in roughly one minute. This is a deliberate trade: it stops the detector from sticking on "occupied" over a warm chair or a sun-heated wall, but it also means a person sitting still slowly fades into the background. For a vent, that second case is the one the occupant feels, so it belongs at the control layer: latch the last occupied state and release it only after a vacancy timeout, rather than acting on a single vacant reading. On the evaluation set the detector reached an F1 of 0.92 against the CNN's 0.80 (Table 1), while being cheaper, deterministic, and interpretable from a calibration log. The CNN's weakness was mostly precision (0.78): it flagged drifting background as occupancy, which for a vent means conditioning an empty room. Rather than reading this as a straightforward accuracy win, we frame the outcome as a capacity-versus-data-regime argument: the CNN's additional capacity cannot be supported by a single-room, single-season corpus, and the result would be expected to reverse under a multi-room, multi-season dataset paired with a dedicated inference budget.

Table 1: Comparison of occupancy inference approaches on the ESP32-S3 node (evaluation set: 2,000 frames, 900 occupied / 1,100 vacant).

Criterion	Approach A: Compact CNN	Approach B: Adaptive BG Subtraction (Adopted)
Precision / Recall	0.78 / 0.82	0.93 / 0.91
F1	0.80	0.92
Accuracy	81.5%	92.9%
Inference latency	~40–70 ms per frame (int8, shared core)	~3–6 ms per frame
RAM	~180–260 KB (weights + TFLite-Micro arena)	<2 KB (background buffer + scratch)
Flash	~120–200 KB	<1 KB

4.4 Voice Interaction Pipeline

Voice interaction uses a two-stage architecture: a custom wake-word model, chosen for distinct identity and tunable sensitivity, and a quantized English MultiNet engine for command recognition. The wake-word pipeline streams clips from a sampler build mode to a collection script, augments real recordings with AI-generated synthetic utterances for diversity, and trains a compact wake-word model exported to TFLite-Micro. A wake event requires several consecutive over-threshold windows plus a refractory period, trading modest latency for

fewer false triggers. Notably, the log-mel front end applies per-window peak normalization, making features gain-invariant, so the noise-floor tracker and decision threshold govern sensitivity rather than gain controls. MultiNet's grapheme-to-phoneme front end lets new phrases be registered as text, useful for an evolving thermostat vocabulary. No audio front end is yet integrated; one mitigation stands in instead: an amplifier enable line gated to clip playback only, eliminating self-acknowledgment bleed given the shared enclosure.

5 ELECTROMECHANICAL ASSEMBLY

The unit went together in three phases: electrical validation on a breadboard, motor and blade fixing, then transfer to a custom carrier PCB (one connector slot per peripheral, laid out around the fan cavity to match where each part would mount), followed by mechanical integration into a repurposed vent housing. A commercial inline booster duct fan enclosure was disassembled and reused as the vent body, retaining its original duct-mating geometry so the finished unit would still mate with standard ducting (Figure 3).



Figure 3: Carrier PCB fixing and repurposed vent housing prior to final assembly.

The motor and blades were assembled as a sub-unit before being fitted into the body: the holders were fixed to the motor seats, the motor mounted into the holders, and a shaft coupler joined the motor shaft to the fan in its rotation-axis cavity (Figure 4).



Figure 4: Motor, motor holder, and blade fixing.

The motor driver and amplifier/speaker were mounted at a separate point in the body, apart from the controller rather than stacked with it (Figure 5, left pair), a layout decision that anticipated, though did not fully resolve, the acoustic and electrical coupling problems discussed in Section 6.



Figure 5: Amplifier and fan unit placed within the vent body (left pair); sensors and display seated in the face panel (right pair).

The face panel, sensors, and enclosure closed out the build. The face panel was cut to hold the display, thermal array, and mic facing forward, since all three need an unobstructed line to the room and to the occupant; they were seated into the panel openings and fixed with silicone adhesive, which also sealed the cutouts against airflow leakage. A protective cover, drilled for the USB programming cable and the power cable, lets the unit be reprogrammed and powered without disassembly (Figure 5, right pair).

6 DISCUSSION AND LIMITATIONS

The development process ran in two stages, pre-assembly (wired, outside the housing) and post-assembly (fully enclosed), to isolate assembly regressions from functional defects. Pre-assembly: thermal readings, occupancy detection, and fan operation were verified first, the unit heard and executed a spoken command over the fan noise, the control loop stopped the fans at target temperature, and a voice-raised target reversed them. Post-assembly: voice and occupancy were verified in standby, airflow confirmed at the vent, and thermal response checked with an ice pack near the sensor. Voice performance, though, dropped noticeably with the fans running. Carried forward: the heuristic occupancy detector, a short, distinct command vocabulary, amplifier gating, the timing-sensitive core partition, and the PSRAM audio path.

The headline finding is physical, not computational. The mic and sensors share a rigid housing with the motor: vibration couples directly into the mic mount, and noise couples acoustically through the structure. Usable wake-word and command range fell to roughly 3–10 cm from the mic with the motor running, against room-scale pickup with the motor stopped (small sample). For the occupant, that is not a degraded interface but a different one: at 3–10 cm the user has to walk up to the vent and speak into it, which defeats the point of hands-free control on an outlet that usually sits on a wall or ceiling. None of the firmware-side changes we tried recovered that range: the constraint was mechanical and, worse, timing-correlated, because interaction degrades exactly when a command is most likely. Generalized: any voice-interactive device sharing an enclosure with an actuator inherits this problem. Conventionally, longer, more distinct phrases recognize more reliably; under noise we saw the reverse. Each phoneme must survive the noise, so error compounds with length. Short, distinct commands were adopted at some cost to naturalness, giving a vocabulary shaped by the machine's noise floor rather than the user's habits (again a small-trial observation). The wake-word model also appeared weaker for female speakers. We attribute this to the corpus: fewer real female recordings, plus augmentation that added volume without prosodic coverage (Fucci et al., 2023). That is a fairness defect, not merely an accuracy gap: a household device that answers some occupants more readily than others distributes comfort unevenly. The evaluation is limited to one room, geometry, and season, with few trials, no held-out multi-environment set, and no audio front end; the results therefore support functional claims, not generalizable accuracy claims.

7 FUTURE WORK

Ordered by expected impact, these set the agenda for Prototype 3. First, physically decouple the sensing and interface cluster from the actuation cluster (either separate enclosures or a damped, isolated sub-mount within one housing) and quantify the range recovery of both before choosing. Second, balance the wake-word corpus across speaker demographics in both the real and synthetic halves, checking the synthetic pipeline for coverage rather than volume, and evaluate it with a gender-balanced speaker panel at fixed distances and fan states. Third, integrate the vendor audio front end for noise suppression; echo cancellation does not apply without a reference path. For the command recognizer, train a purpose-built classifier on this vocabulary and noise profile, with DTW matching as a low-complexity fallback. For occupancy, revisit the CNN once a multi-room, multi-season corpus and a dedicated inference budget exist, and consider a hybrid scheme with the blob detector running continuously and the CNN invoked only on borderline detections.

8 CONCLUSION

This paper covers a single P2 node, and we make no claim that the thresholds or part choices reported here are the ideal choices for another build, though the general system architecture may transfer. What can be carried forward is the co-residency principle, the timing-sensitivity partitioning of tasks across cores, and the acoustic-coupling lesson: that a voice interface sharing an enclosure with an actuator is bounded by physics rather than by model capacity, and that the occupant feels that bound first.

Prototype 2 demonstrates the integration of local thermal sensing, staged on-device voice interaction, display feedback, and reversible airflow in one ESP32-S3 smart vent node. For this narrow installation, an adaptive background-subtraction detector was a practical and interpretable occupancy method, while the post-assembly, fan-on observation exposed a more important deployment risk: voice interaction can become unusable when the sensing interface shares an enclosure with active mechanical components. The contribution is therefore an evidence-bounded implementation lesson, not a claim of general recognition accuracy or concurrent neural workload performance.

REFERENCES

Aszkowski, P. and Piechocki, M. (2022). Thermo Presence: The Low-resolution Thermal Image Dataset and Occupancy Detection Using Edge Devices. Proceedings of the 3rd Polish Conference on Artificial Intelligence (PP-RAI'2022), Gdynia, Poland, pp. 49–52.

Chidurala, V. and Li, X. (2022). Detection of moving objects using thermal imaging sensors for occupancy estimation. Internet of Things, 17, 100487. doi:10.1016/j.iot.2021.100487.

Cui, X., et al. (2025). A computer vision-based model for occupancy detection using low-resolution thermal images. arXiv:2505.08336.

David, R., et al. (2021). TensorFlow Lite Micro: Embedded Machine Learning on TinyML Systems. Proceedings of MLSys. arXiv:2010.08678.

Fang, B., Zeng, X. and Zhang, M. (2018). NestDNN: Resource-Aware Multi-Tenant On-Device Deep Learning for Continuous Mobile Vision. Proceedings of MobiCom. doi:10.1145/3241539.3241559.

Fucci, D., et al. (2023). No Pitch Left Behind: Addressing Gender Unbalance in Automatic Speech Recognition through Pitch Manipulation. arXiv:2310.06590.

Klir, S., Lerch, J., Benkner, S. and Khanh, T. Q. (2025). Multi-Person Localization Based on a Thermopile Array Sensor with Machine Learning and a Generative Data Model. Sensors, 25(2), 419. doi:10.3390/s25020419.

Lee, J., et al. (2026). Quilt: Peak-memory-aware partitioning and scheduling for multi-tenant DNN model inference. Journal of Systems Architecture, 173, 103696. doi:10.1016/j.sysarc.2026.103696.

Shubha, B. and Shastrimath, V. V. D. (2022). Real-Time Occupancy Detection System Using Low-Resolution Thermopile Array Sensor for Indoor Environment. IEEE Access, 10, pp. 130981–130995. doi:10.1109/ACCESS.2022.3229895.

Zhang, Y., Suda, N., Lai, L. and Chandra, V. (2017). Hello Edge: Keyword Spotting on Microcontrollers. arXiv:1711.07128.