iJRASET
International Journal For Research in
Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 Issue: IV Month of publication: April 2026

DOI: https://doi.org/10.22214/ijraset.2026.80803

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue IV Apr 2026- Available at www.ijraset.com

Autonomous Olfactory Canine Robot

Rohith Natteti[1], Nagarjuna Reddy Golamaru[2]
Electronics and Communication Engineering, NBKR Institute of Science and Technology (Autonomous), Affiliated to JNTUA, Ananthapuramu, Vidyanagar-524413, Tirupati Dist., Andhra Pradesh, India

Abstract: The Autonomous Olfactory Canine Robot (AOCR) is an advanced electronic nose (e-nose) engineered for trace gas monitoring and environmental odor classification. Its primary purpose is to overcome the severe electrical noise and baseline drift inherent in Metal-Oxide Semiconductor (MOS) gas sensors by utilizing a highly isolated dual-controller architecture. An Arduino Mega 2560 safely manages 5V high-current sensor heaters and precise 16-bit analog-to-digital conversions, while an ESP32 handles 3.3V logic, WiFi synchronization, and environmental compensation. The AOCR successfully mitigates hardware interference, calculates normalized chemical "Signatures" from its eight-sensor array, and securely uploads a comprehensive 66-column dataset to a Google Sheets cloud database to facilitate real-time monitoring and future machine learning analysis.
Keywords: Autonomous Olfactory Canine Robot (AOCR), Dual-Controller Architecture, Metal-Oxide Semiconductor (MOS) Gas Sensors, Electronic Nose (E-Nose), Environmental Compensation

I. INTRODUCTION

The Autonomous Olfactory Canine Robot (AOCR) is an advanced electronic nose (e-nose) engineered for trace gas monitoring and environmental odor classification. Developing an autonomous electronic nose presents severe hardware challenges, particularly regarding analog signal integrity. This project addresses two fundamental limitations that prevent commodity microcontrollers from achieving reliable gas sensing.

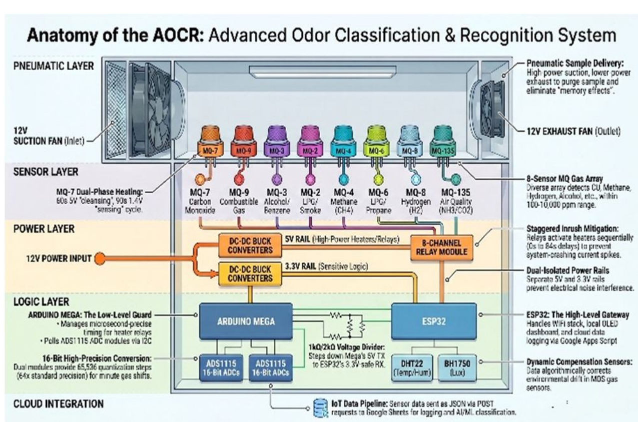
First, standard microcontrollers rely on native 10-bit Analog-to-Digital Converters (ADCs), offering only 1,024 discrete resolution steps. This is insufficient for capturing the subtle chemical transients of trace gases in low concentrations where the signal-to-noise ratio is generally poor. The AOCR solves this by integrating 16-bit ADS1115 modules, providing 65,536 quantization steps — a 64-fold improvement in measurement precision.

Second, Metal-Oxide Semiconductor (MOS) gas sensors require high-current heating coils that generate massive transient voltage spikes and electromagnetic interference (EMI). This noise can easily corrupt delicate sensor readings and crash sensitive logic controllers. The AOCR resolves this by implementing a highly isolated dual-controller architecture, independent DC-DC power distributions, and hardware-level signal filtering to completely mitigate electrical interference.

The result is a system that not only collects high-fidelity gas data but also uploads a structured 66-column dataset per sample to a Google Sheets cloud database, enabling real-time monitoring and creating a rich dataset for future machine learning classification.

II. HARDWARE AND SYSTEM DESIGN

The AOCR system architecture is designed to isolate high-current heater noise from sensitive logic processing units. The complete hardware connection list follows the block architecture shown in Fig. 1.



A. *Dual-Controller Architecture*

The system divides computational tasks between two controllers to prevent network latency from disrupting sensor sampling.

1) *Low-Level Controller:* An Arduino Mega 2560 operates at 5V logic and is dedicated to timing-critical tasks, such as managing the 8-channel relay module for sensor heaters and polling the high-resolution ADS1115 ADCs via the I²C bus. The Mega features 54 digital I/O pins, 16 analog inputs, and four hardware UARTs, providing sufficient I/O for simultaneous relay and sensor management.
2) *High-Level Controller:* An ESP32-D operates at 3.3V logic and serves as the communication gateway, offloading resource-intensive tasks like WiFi stack management, cloud synchronization via Google Apps Script, and rendering real-time data to the SSD1306 OLED display. This separation ensures that WiFi operations do not introduce timing jitter into the sensor sampling cycle.
3) *UART Communication and Logic Level Shifting:* To prevent the Mega's 5V transmit (TX) signal from permanently damaging the ESP32's 3.3V receive (RX) pin, the UART serial connection utilizes a resistor voltage divider circuit using 1 kΩ and 2 kΩ resistors, stepping the signal down to the safe 3.3V range. Data is transmitted at 9600 baud from Mega TX1 (Pin 18) to ESP32 GPIO 32.

B. *Power Distribution and Noise Isolation*

1) *Independent Buck Converters:* The system steps down a main 12V supply using two completely independent DC-DC buck converter modules. The first creates a 5V high-power rail for the Arduino Mega, relays, ADS1115 ADCs, and all MQ sensor heaters. The second creates an isolated 3.3V low-power rail for the ESP32, SSD1306 OLED, DHT22, and BH1750. Using two independent converters rather than a sequential chain physically protects the sensitive 3.3V logic from the severe electrical noise generated by high-current 5V operations.
2) *Inrush Current Mitigation:* Each MQ sensor draws approximately 150 mA, resulting in a collective power demand exceeding 6 W. When relay coils switch, they introduce transient voltage sags. To counteract this, 10 µF bulk capacitors are placed across the 5V power rails near the relay board and sensor array, acting as localized energy reservoirs that instantly supply current during heavy switching events.

C. *Sensor Array and Signal Conditioning*

1) *MOS Gas Sensor Array:* The array consists of eight MQ-series chemiresistive sensors: MQ-2 (smoke/LPG), MQ-3 (alcohol), MQ-4 (methane), MQ-6 (LPG/propane), MQ-7 (carbon monoxide), MQ-8 (hydrogen), MQ-9 (CO/combustible gases), and MQ-135 (air quality/ammonia). Each sensor uses a tin dioxide (SnO_2) sensing layer whose electrical resistance changes in proportion to the target gas concentration.
2) *Low-Pass RC Filters:* A hardware-level low-pass filter consisting of a 100 Ω resistor and a 100 nF capacitor is placed on the analog output of every single MQ sensor. This suppresses high-frequency EMI before the signal reaches the ADS1115 ADCs, ensuring the subtle chemical transients are not corrupted by electrical interference.
3) *High-Resolution ADC Modules:* Two ADS1115 16-bit ADC modules are used, addressed at 0x48 and 0x49 on the I²C bus. The first module (U1, 0x49) reads MQ-6, MQ-4, MQ-2, and MQ-3. The second module (U2, 0x48) reads MQ-7, MQ-8, MQ-9, and MQ-135. I²C pull-up resistors of 4.7 kΩ are placed on both the SDA and SCL lines.

D. *Environmental Compensation Sensors*

Because MOS sensors are highly susceptible to baseline drift from ambient temperature and humidity changes, the ESP32 actively polls a DHT22 sensor (GPIO 13) for temperature and humidity, and a BH1750 sensor (GPIO 21/22) for ambient light levels via I²C. These environmental parameters are included in every cloud upload, enabling future compensation algorithms to normalize readings against environmental artifacts.

TABLE I
HARDWARE COMPONENT SUMMARY

COMPONENT	VOLTAGE	FUNCTION
Arduino Mega 2560	5V Logic	Sensor heater control, ADC polling, UART
ESP32-D	3.3V Logic	WiFi, cloud sync, OLED, environmental sensors

ADS1115 (x2)	5V Power	16-bit ADC for 8 MQ sensors via I²C
8-Ch Relay Module	5V Coils	Staggered heater activation and MQ-7 dual-phase cycle
MQ-Series (x8)	5V Heater	Chemiresistive gas sensing (MQ-2,3,4,6,7,8,9,135)
DHT22	3.3V	Temperature and humidity compensation
BH1750	3.3V	Ambient light context logging
SSD1306 OLED	3.3V	Local real-time display via SPI

III. METHODOLOGY

A. Active Pneumatic Sample Delivery

The AOCR does not rely on passive gas diffusion; instead, it uses an Active Pneumatic System. A 12V suction fan (activated at 85 seconds into the warm-up cycle by the Arduino Mega on Pin 48) draws environmental air through a protective mesh filter into a sealed sensing chamber.

This forces a controlled directional airflow over the sensor array, maximizing chemical interaction while minimizing turbulence. Computational Fluid Dynamic (CFD) research on similar architectures identifies an optimal airflow velocity of approximately 2 cm/s for maximizing odor-molecule residence time on the sensor surfaces.

A 12V exhaust fan (Pin 52) activates at 150 seconds to rapidly purge residual gases from the chamber after sampling is complete. This resets the sensors to clean-air conditions and eliminates "memory effects" between measurement cycles.

B. Staggered Warm-Up and MQ-7 Dual-Phase Cycle

MOS sensor heaters have a much lower electrical resistance when cold, creating a large inrush current during startup. To prevent this from crashing the power supply, the Arduino Mega activates the 8 sensor heaters sequentially via the relay module, with one sensor activating every 12 seconds (MQ-7 at 0s, MQ-9 at 12s, MQ-3 at 24s, MQ-2 at 36s, MQ-4 at 48s, MQ-6 at 60s, MQ-8 at 72s, MQ-135 at 84s).

The MQ-7 (Carbon Monoxide) sensor requires a unique dual-phase heating cycle for accurate CO measurement. Unlike other MQ sensors that operate at continuous 5V, the MQ-7 uses a relay to apply 5V for 60 seconds (a high-temperature cleansing phase that burns off interfering oxidized compounds) followed by 1.4V for 90 seconds (the actual sensing phase where CO absorption is measured). This 150-second periodicity is managed entirely by the Arduino Mega's timer logic.

C. Odor Classification and Cloud Data Processing

Once sensors are stabilized (120 seconds after startup), the Mega transmits CSV-formatted 16-bit ADC readings to the ESP32 over UART. The ESP32 packages these values with environmental metadata (temperature, humidity, lux) into a GET request sent to a Google Apps Script deployment URL.

The cloud script performs the following processing pipeline for each sample:

- Normalization: Each sensor reading is divided by its clean-air baseline value (Rs/R0) to produce a Normalized Value (NORM_MQx).
- Delta Calculation: The difference between the current reading and baseline (MQx - BASE_MQx) produces a Delta Value (DELTA_MQx).
- Virtual Sensor Ratios: Twelve inter-sensor ratios are computed (e.g., MQ2/MQ3, MQ4/MQ6, MQ7/MQ8) to create dimensionless chemical fingerprints that are robust to absolute baseline shifts.
- Dominant Sensor Classification: The sensor with the highest delta value is identified as the Dominant Sensor, and the corresponding gas group is assigned (e.g., Dominant = MQ4 → GasGroup = "Methane", SpectrumColor = "Green").
- Signature Generation: A string of all 8 delta values joined by hyphens forms the unique chemical "Signature" for that sample.

The complete 66-column dataset (ID, Timestamp, Label, Matched_Label, Confidence, DominantSensor, GasGroup, SpectrumColor, Signature, 3 environmental + 8 raw + 8 baseline + 8 normalized + 8 delta + 12 virtual sensor + 8 rise rate columns) is appended to Google Sheets for persistent storage and future ML analysis.

IV. FIRMWARE ARCHITECTURE

A. Arduino Mega Firmware

The Mega firmware is written in C++ using the Arduino framework. On power-up, it initializes all relay pins HIGH (heaters OFF), configures the ADS1115 modules via I²C, and enters a listening state awaiting commands from the ESP32 via UART (Serial1 at 9600 baud).

Two operational modes are supported via a timing multiplier variable. The INIT_CYCLE command sets a 1.0x multiplier for normal full-speed warm-up (120 seconds to SENSORS_READY), while the CALIB_CYCLE command sets a 0.33x multiplier for a fast recalibration cycle (~40 seconds), allowing rapid session resets without full thermal stabilization. Upon receiving the START_CYCLE command after sensor readiness, the Mega reads all eight ADS1115 channels and transmits the CSV packet over UART.

B. ESP32 Firmware

The ESP32 firmware implements a finite state machine with the following states: SPLASH, HANDSHAKE, INITIALIZING, WAITING_SENSORS, DASHBOARD, CALIBRATING, and SUCCESS_SPLASH. The HANDSHAKE state performs a two-way serial verification (AOCR_MASTER_REQ / AOCR_SLAVE_ACK) before proceeding to ensure communication integrity.

During INITIALIZING, a 99-second countdown is displayed on the OLED with a dynamic progress bar. Upon entering DASHBOARD mode, the system enters a continuous monitoring loop: parse CSV from Mega → capture baseline on first reading → calculate normalized values → sync to Google Sheets → parse label response → update OLED → request next sample.

V. RESULTS AND DISCUSSION

The AOCR successfully demonstrated reliable 16-bit sensor acquisition and cloud synchronization in continuous operation. The dual-controller architecture effectively isolated the ESP32 from the EMI generated by relay switching events, eliminating the system crashes observed in single-controller prototype configurations.

The staggered warm-up routine proved critical for power stability. Initial tests without staggering showed voltage sags of sufficient magnitude to reset the Arduino Mega. With the 12-second stagger intervals, the power rails remained stable throughout the warm-up sequence.

The 66-column dataset structure captures sufficient feature diversity for supervised machine learning. The normalized values, delta values, and 12 virtual sensor ratios together form a high-dimensional chemical fingerprint that is robust against absolute baseline drift. The Dominant Sensor classification provides an interpretable first-pass label while the full signature enables more nuanced ML classification.

The OLED dashboard provided real-time feedback during testing, displaying all 8 sensor values, environmental parameters (T, H, L), the last received cloud label, and confidence score. The Boot Button recalibration feature (triggering CALIB_CYCLE at 0.33x speed) allowed rapid iterative testing without full 120-second restarts.

VI. CONCLUSION

The Autonomous Olfactory Canine Robot (AOCR) successfully demonstrates that a high-fidelity electronic nose can be built from commodity hardware by addressing the two fundamental problems of resolution and electrical noise through architectural design rather than expensive specialized components. The dual-controller architecture, independent power isolation, hardware-level RC filtering, and staggered warm-up routine collectively eliminate the noise and interference that limit single-controller MOS sensor systems.

The structured 66-column Google Sheets dataset, built on normalized values, delta values, and virtual sensor ratios, provides a robust foundation for training supervised machine learning classifiers for gas identification. Future work will focus on implementing a Support Vector Machine or lightweight neural network model for real-time on-device gas classification, moving toward a fully autonomous odor identification platform suitable for environmental monitoring and safety applications.

VII. ACKNOWLEDGMENT

The author thanks the faculty of the Electronics and Communication Engineering Department at NBKR Institute of Science and Technology for their guidance throughout this project. Special acknowledgment is given to the open-source Arduino and ESP32 communities whose libraries and documentation were instrumental in the development of this system.

REFERENCES

[1] N. Bhattacharya, S. Bhattacharyya, R. Chowdhury, and A. Gupta, "Electronic nose: A review," *Indian J. Phys.*, vol. 82, no. 7, pp. 859–862, 2008.

[2] A. Hulanicki, S. Glab, and F. Ingman, "Chemical sensors: definitions and classification," *Pure Appl. Chem.*, vol. 63, no. 9, pp. 1247–1250, 1991.

[3] "ADS1115 Ultra-Small, Low-Power, 16-Bit Analog-to-Digital Converter with Internal Reference, Oscillator, and Programmable Comparator," Texas Instruments, Dallas, TX, Datasheet SBAS444D, 2009.

[4] "MQ-Series Semiconductor Gas Sensor Datasheet," Hanwei Electronics Co., Ltd., Zhengzhou, China.

[5] Espressif Systems, "ESP32 Series Datasheet," Version 3.6, Shanghai, China, 2022. [Online]. Available: https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf

[6] Arduino, "Mega 2560 Rev3 Technical Specifications," 2023. [Online]. Available: https://docs.arduino.cc/hardware/mega-2560

[7] R. Gutierrez-Osuna, "Pattern analysis for machine olfaction: A review," *IEEE Sensors J.*, vol. 2, no. 3, pp. 189–202, Jun. 2002.

[8] Google LLC, "Google Apps Script — Reference Documentation," 2024. [Online]. Available: https://developers.google.com/apps-script/reference