PRAANBOT-Cam: A Mobile IoT Robot for Hyperlocal Air-Quality Monitoring in Major Urban Centers

---

1. Executive Summary

Many of the world's major urban centers struggle with severe air pollution, a well-documented public health crisis. Existing monitoring systems, while crucial, provide city-level data that often misses hyperlocal variations where pollution can be significantly higher. This gap in real-time, granular data hinders effective policy-making, community awareness, and personal protective measures. To address this challenge, we present **PRAANBOT-Cam**, a low-cost, scalable, and autonomous mobile IoT robot designed for hyperlocal air-quality monitoring in any major city, using Delhi as a prime example.

PRAANBOT-Cam is an integrated system that combines robotics, IoT, and artificial intelligence. The robot autonomously navigates designated urban areas, capturing real-time air quality data (CO, smoke, LPG, NH3, etc.) and a live video feed of its surroundings. This data is streamed to a cloud database (Baserow) via Wi-Fi. In the backend, a suite of AI models analyzes the data stream to predict pollution trends, detect anomalous spikes, and identify pollution hotspots.

The core innovation lies in closing the loop from data acquisition to actionable intelligence. When a pollution event is detected or predicted, the system automatically dispatches alerts to local communities and authorities via a Telegram bot. A public-facing dashboard provides a live view of the robot's data, including an Air Quality Index (AQI) graph, the camera feed, and trend forecasts.

This white paper details the complete design and architecture of the PRAANBOT-Cam system, from its hardware and software components to the AI-driven analytics and alert mechanisms. We argue that this affordable and replicable solution offers a new paradigm for environmental monitoring, empowering communities globally with the data needed to drive change and enhance climate resilience in polluted urban environments.

---

2. Background – The Urban Air Pollution Challenge

Major cities across the globe, from Delhi to Beijing, and from Mexico City to Los Angeles, face a recurring and severe air pollution crisis. Using Delhi as a case study, the problem becomes starkly evident. The National Capital

Territory (NCT) of Delhi, for instance, faces a severe crisis, particularly during the winter months. A combination of factors, including vehicular emissions, industrial discharge, construction dust, and seasonal agricultural practices, contributes to a toxic atmospheric cocktail. Particulate matter (PM2.5 and PM10), nitrogen oxides (NOx), sulfur dioxide (SO2), carbon monoxide (CO), and volatile organic compounds (VOCs) frequently exceed safe limits set by the World Health Organization (WHO) by several orders of magnitude.

This chronic exposure to polluted air has dire consequences for public health, leading to a high incidence of respiratory illnesses, cardiovascular diseases, and reduced life expectancy. While many cities have deployed networks of Continuous Ambient Air Quality Monitoring Stations (CAAQMS), these stations are often sparsely located. Their data, while accurate for their immediate vicinity, fails to capture the significant spatial and temporal variability of air pollution at the street or neighborhood level.

A resident in an industrial zone or near a high-traffic corridor may experience far worse air quality than what is reported by the nearest official monitor several kilometers away. This lack of hyperlocal data creates a critical information gap in many cities, preventing: * **Targeted Interventions:** Authorities cannot pinpoint specific sources of pollution for enforcement. * **Public Awareness:** Citizens lack the information to make informed decisions, such as avoiding certain routes or wearing masks. * **Scientific Research:** Researchers are limited in their ability to study the micro-dynamics of urban pollution.

PRAANBOT-Cam is conceived as a direct response to this global challenge, aiming to democratize air quality data and make it a real-time, hyperlocal, and actionable utility for any city that needs it.

---

3. System Architecture

The PRAANBOT-Cam system is designed as a modular and scalable end-to-end solution. The architecture can be broken down into four primary layers: the Edge Layer (the robot), the Communication Layer, the Cloud & Analytics Layer, and the User Engagement Layer.

Textual Diagram: System Flow

1. **Edge Layer (PRAANBOT-Cam Robot):**
    - **Input:** Onboard sensors (MQ135 for air quality, HC-SR04 for obstacles, IR for line-following/edge detection) and an ESP32-CAM for video.
    - **Processing:** An ESP32 microcontroller reads sensor data, controls the L298N motor driver for navigation, and packages the data into a JSON object.
    - **Output:** The JSON payload and video stream are transmitted over Wi-Fi.

2. **Communication Layer:**
    - **Protocol:** The robot connects to a local Wi-Fi network (or a mobile hotspot for extended range).
    - **Data Transmission:** An HTTP POST request sends the JSON data to a Baserow database API endpoint. The ESP32-CAM streams video over a separate HTTP stream.
3. **Cloud & Analytics Layer:**
    - **Database:** Baserow, an open-source no-code database, receives and stores the time-series data from the robot. Its API-first design simplifies integration.
    - **AI/ML Platform:** Kaggle Notebooks are used to periodically fetch data from Baserow via its API.
        - **Prediction:** An LSTM (Long Short-Term Memory) model is trained on historical data to forecast near-term AQI trends.
        - **Anomaly Detection:** An Isolation Forest algorithm runs on the real-time data stream to identify sudden, unexpected pollution spikes.
        - **Hotspot Analysis:** A K-Means clustering algorithm groups the geo-tagged (future work) or location-approximated data to identify persistent pollution hotspots, which are then visualized on a Folium map.
    - **Data Flow:** The results of the AI analysis (e.g., an anomaly alert) are written back to a separate table in Baserow or trigger a webhook.
4. **User Engagement Layer:**
    - **Alerts:** A webhook from the analytics platform triggers a Python script that formats a message and sends it to a designated Telegram channel or group via the Telegram Bot API.
    - **Dashboard:** A web-based dashboard (e.g., built with Dash or a simple HTML/JavaScript frontend) continuously queries the Baserow API to display:
        - A real-time line graph of AQI values.
        - An embedded live video feed from the ESP32-CAM.
        - The LSTM-generated trend forecast for the next few hours.
        - The Folium map showing identified pollution hotspots.

---

4. Hardware Overview

The PRAANBOT-Cam is built using off-the-shelf, low-cost, and widely available electronic components, ensuring its replicability in any part of the world.

- **Chassis:** A simple 2WD or 4WD robot chassis with DC motors and wheels.
- **Microcontroller (MCU): ESP32-WROOM-32**. This MCU is the brain of the robot. It was chosen for its dual-core processor, built-in Wi-Fi and Bluetooth capabilities, and ample GPIO pins, making it ideal for handling sensor inputs, motor control, and network communication simultaneously.
- **Air Quality Sensor: MQ135 Gas Sensor**. This is a versatile and affordable sensor capable of detecting a range of gases, including

ammonia (NH3), nitrogen oxides (NOx), alcohol, benzene, smoke, and carbon dioxide (CO2). While not as precise as professional-grade sensors, it is highly effective for detecting relative changes in air quality and identifying the presence of pollutants. The analog output is read by one of the ESP32's ADC pins.

- **Obstacle Avoidance Sensor: HC-SR04 Ultrasonic Sensor**. This sensor is used for basic navigation and obstacle avoidance. It emits an ultrasonic pulse and measures the time it takes for the echo to return, allowing the robot to calculate the distance to objects in its path. When an obstacle is detected within a predefined threshold, the robot executes a pre-programmed maneuver (e.g., stop, reverse, turn).
- **Infrared (IR) Sensors:** Two IR proximity sensors are mounted at the front, facing downwards. These can be used for simple line-following applications or, more critically, as edge detectors to prevent the robot from falling off ledges or stairs.
- **Camera Module: ESP32-CAM**. This is a separate, compact board featuring an ESP32-S microcontroller, an OV2640 camera, and a microSD card slot. It is dedicated to capturing and streaming video. It runs its own web server to provide a live MJPEG stream, which is accessed independently by the dashboard. This offloads the video processing from the main ESP32, ensuring that sensor reading and motor control are not interrupted.
- **Motor Driver: L298N Dual H-Bridge Motor Driver**. This module allows the ESP32, which operates at 3.3V and low current, to control the high-current DC motors. It can control the speed (via PWM) and direction of two motors independently.
- **Power Supply:** A portable power bank or a Li-Po battery pack provides power to the entire system, regulated to 5V for the motors and 3.3V for the microcontrollers and sensors.

---

5. Software Design

The software is designed for simplicity, robustness, and easy integration with web services. The primary firmware is written in C/C++ using the Arduino framework for the ESP32.

JSON Data Schema

To ensure data consistency and interoperability, all sensor readings are packaged into a standardized JSON format before transmission. This structured format is easily parsed by the Baserow database and other web services.

Example JSON Payload:

```
{
  "robot_id": "PRAANBOT-001",
  "timestamp": "2025-11-07T14:30:00Z",
  "city": "Delhi",
  "location_approx": "Urban Residential Area, Zone 4",
  "sensors": {
```

```
    "mq135_raw": 750,
    "aqi_estimated": 185,
    "pollutants": {
      "co": 15.2,
      "nh3": 2.5,
      "smoke": 45.1
    }
  },
  "status": "patrolling"
}
```

Wi-Fi Networking and Database Integration

- **Connectivity:** On startup, the main ESP32 connects to a pre-configured Wi-Fi network. The credentials can be hardcoded or set up via a simple web portal (WiFiManager library).
- **Baserow Integration:** The robot sends data to Baserow by making an HTTP POST request to a specific table's API endpoint. An API key is included in the request header for authentication. The `Arduino_JSON` and `HTTPClient` libraries are used for this purpose.
    - **Workflow:**
        1. Read data from the MQ135 and other sensors.
        2. Convert raw analog readings to estimated PPM or an AQI value using calibration curves.
        3. Construct the JSON string.
        4. Initialize an HTTP client.
        5. Set the request header (`Content-Type: application/json`, `Authorization: Token YOUR_API_KEY`).
        6. Send the POST request with the JSON payload.
        7. Check the HTTP response code to confirm successful data submission.
        8. Enter a delay period (e.g., 60 seconds) before the next reading to manage data volume and power consumption.

---

6. Alerts & Automation – Telegram Bot Workflow

Real-time alerts are critical for making the data actionable. We use a Telegram bot for its simplicity and widespread adoption.

Workflow:

1. **Trigger:** The AI/ML script running on Kaggle (or a dedicated server) detects an anomaly or predicts a pollution spike. For example, the Isolation Forest model flags a data point as an outlier, or the LSTM model forecasts the AQI to cross a "Severe" threshold within the next hour.
2. **Webhook/API Call:** The detection script makes an API call to a simple Python web service (e.g., a Flask or FastAPI app) or a serverless function.

3. **Message Formatting:** This service receives the alert data (e.g., robot ID, location, AQI value, type of pollutant). It formats a human-readable message.
    - Example Message: “ **Air Quality Alert**

Location: Urban Residential Area, Zone 4 **AQI:** 185 (Unhealthy) **Details:** Sudden spike in Carbon Monoxide (CO) detected by PRAANBOT-001.

Recommendation: Avoid outdoor activity if possible. Consider using an air purifier.” 4. **Dispatch:** The Python script uses the `python-telegram-bot` library to send this formatted message to a pre-configured Telegram channel subscribed to by residents, local administrators, or health officials.

This automated workflow transforms a raw data point into a timely, contextual, and potentially life-saving notification with zero human intervention.

---

7. AI & Analytics

The AI and analytics component is the core of the system’s intelligence, turning raw data into predictive insights. All models are developed and run in Kaggle Notebooks, which provide a free and powerful environment for data science.

- **Data Acquisition:** The notebook uses the `requests` library in Python to fetch data from the Baserow database API.
- **Time-Series Prediction with LSTM:**
    - **Objective:** To forecast future AQI values based on historical trends.
    - **Model:** A Long Short-Term Memory (LSTM) network, a type of Recurrent Neural Network (RNN), is used. LSTMs are exceptionally well-suited for learning from time-series data, as they can remember patterns over long sequences.
    - **Process:**
        1. The historical AQI data is preprocessed and scaled.
        2. It is then structured into sequences (e.g., use the last 60 minutes of data to predict the next 10 minutes).
        3. The LSTM model is trained on this sequential data.
        4. Once trained, the model takes the most recent sequence of data and outputs a prediction for the next time steps. This forecast is then displayed on the dashboard.
- **Anomaly Detection with Isolation Forest:**
    - **Objective:** To detect sudden, abnormal spikes in pollution that deviate from the established pattern.
    - **Model:** The Isolation Forest algorithm is an unsupervised learning model that is highly efficient at detecting outliers. It works by randomly partitioning the data and “isolating” observations.

Anomalies are easier to isolate and thus have shorter path lengths in the decision trees.
    - **Process:**
        1. The model is trained on a baseline of “normal” air quality data.
        2. It then evaluates each new data point from the robot in real-time.
        3. If a data point is flagged as an anomaly (i.e., it has a low anomaly score), it triggers the Telegram alert workflow. This is effective for identifying events like illegal waste burning or a sudden increase in traffic.

- **Hotspot Mapping with K-Means and Folium:**
    - **Objective:** To identify geographical areas with persistently high pollution levels.
    - **Model:** K-Means clustering is an unsupervised algorithm used to partition the data into ‘K’ distinct clusters.
    - **Process:**
        1. The dataset, including location data (initially approximated, later from GPS), is collected over time.
        2. The K-Means algorithm is applied to group the data points based on their pollution levels and coordinates. This will result in clusters representing low, medium, and high pollution zones.
        3. The **Folium** library in Python is used to visualize these results. A map of the target city is generated, and the data points are plotted on it, colored according to their cluster. This creates a powerful visual tool for identifying and communicating the locations of pollution hotspots.

---

8. Dashboard

The public-facing dashboard serves as the primary user interface for accessing the system’s data. It is designed to be intuitive and informative.

Key Components:

1. **Real-Time AQI Graph:**
    - A dynamic line chart, built with a library like Chart.js or Plotly.js.
    - It fetches data from the Baserow API every minute and updates the graph to show the latest AQI trend.
    - The background of the chart is color-coded according to standard AQI categories (Good, Satisfactory, Moderate, Poor, Very Poor, Severe) to provide at-a-glance understanding.
2. **Live Camera Feed:**
    - An `<img>` tag or video player element in the HTML whose `src` attribute points directly to the IP address and port of the ESP32-CAM’s video stream (e.g., `http://192.168.1.105:81/stream`).

    - This provides crucial visual context. For example, if a pollution spike is detected, the video feed might reveal the source, such as a garbage fire or a vehicle emitting black smoke.
3. **Trend Forecast:**
    - A small widget that displays the LSTM model's prediction for the next 1-3 hours.
    - It might show a simple arrow (up, down, or stable) and the predicted AQI range.
4. **Hotspot Map:**
    - An embedded Folium map showing the results of the K-Means clustering analysis. Users can zoom and pan to explore pollution patterns across the monitored areas.

---

9. Results & Discussion

The PRAANBOT-Cam system successfully demonstrates a closed-loop architecture for environmental monitoring: **IoT → AI → Alert**.

- **IoT (Data Collection):** The robot proved to be a reliable platform for collecting hyperlocal air quality data that is otherwise unavailable. Its mobility allows it to cover a wider area than a stationary sensor, providing a richer dataset.
- **AI (Data Analysis):** The application of LSTM, Isolation Forest, and K-Means algorithms transforms the raw data stream into high-value intelligence. The predictive capability of the LSTM model allows for proactive, rather than reactive, responses. The Isolation Forest is highly effective at flagging transient but significant pollution events that would be lost in daily averages.
- **Alert (Actionable Intelligence):** The automated Telegram alerts bridge the final gap, delivering this intelligence directly to the people who need it. This empowers residents to take protective measures and provides authorities with evidence for targeted enforcement.

The integration of a live camera feed is a significant value-add. It provides qualitative, visual evidence to support the quantitative sensor data. This can be invaluable for source apportionment and for creating compelling narratives to drive public and political action.

The primary achievement of this project is its demonstration of how affordable, open-source technologies can be integrated to build a sophisticated and impactful environmental monitoring system. It challenges the notion that such systems must be prohibitively expensive, paving the way for community-driven, citizen-science initiatives worldwide.

---

10. Future Work

PRAANBOT-Cam is a foundational platform with significant potential for expansion and improvement.

- **GPS Integration:** Adding a GPS module (e.g., NEO-6M) to the robot would provide precise geo-tagging for each data point. This would dramatically improve the accuracy of the K-Means hotspot mapping and allow for the creation of high-resolution pollution maps.
- **Autonomous Navigation:** The current obstacle-avoidance system is basic. Future iterations could incorporate SLAM (Simultaneous Localization and Mapping) using a LiDAR sensor or advanced computer vision algorithms to enable true autonomous navigation and mapping of complex indoor and outdoor environments.
- **Drone-Based Monitoring:** The entire sensor and communication package could be adapted for use on an aerial drone. This would allow for 3D mapping of pollution, monitoring of difficult-to-reach areas (e.g., industrial smokestacks), and rapid deployment to emergency sites.
- **Solar Power:** To increase operational endurance and sustainability, the robot could be equipped with solar panels and a charge controller, allowing it to recharge its batteries during the day.
- **Multi-Robot Systems:** Deploying a fleet of PRAANBOTs would enable the creation of a distributed sensor network. A central server could coordinate their movements to ensure optimal coverage of a large area, creating a real-time, city-wide hyperlocal monitoring grid.
- **Sensor Fusion and Calibration:** Incorporating more specialized sensors (for PM2.5, O3, SO2) and co-locating the robots with official monitoring stations periodically would allow for better calibration and more accurate, multi-pollutant AQI calculations.

---

11. References

1. World Health Organization. (2021). WHO global air quality guidelines: particulate matter (PM2.5 and PM10), ozone, nitrogen dioxide, sulfur dioxide and carbon monoxide.
2. Central Pollution Control Board (CPCB), India. National Air Quality Index.
3. Liu, F. T., Ting, K. M., & Zhou, Z. H. (2008). Isolation forest. In Proceedings of the 8th IEEE International Conference on Data Mining.
4. Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780.
5. Espressif Systems. ESP32-WROOM-32 Datasheet.
6. Baserow. Baserow API Documentation. https://baserow.io/api-docs
7. Arduino. Arduino Core for the ESP32. https://github.com/espressif/arduino-esp32