**Computer Vision Based Air Quality Analysis Using Cloud-Connected Sensors**

ABHI.C.JOJI

Computer Science Engineering

Lovely professional university

Jalandhar, Phagwara

[abhicjoji@gmail.com](mailto:abhicjoji@gmail.com)

Kotagiri Kulbhushan

Computer Science Engineering

Lovely professional university

Jalandhar, Phagwara

[kulbhushankotagiri@gmail.com](mailto:kulbhushankotagiri@gmail.com)

**1.Abstract:**

This project introduces a novel computer vision-based methodology for estimating air pollution levels by analyzing environmental images, eliminating the need for conventional hardware-based air quality sensors. Leveraging advancements in image processing and cloud integration, the proposed system utilizes various visual characteristics within photographs—such as contrast, brightness, edge density, color saturation, and blue channel intensity—to infer the presence and severity of air pollutants. The input images are stored and retrieved through Google Drive, effectively simulating a distributed, cloud-connected sensor network that allows for real-time and remote data collection.

OpenCV, a powerful open-source computer vision library, is employed for comprehensive image analysis and feature extraction. Key image features that correlate with pollution indicators are computed and analyzed to derive pollution scores. These scores are further categorized into qualitative air quality classes (e.g., Good, Moderate, Poor) to facilitate easy interpretation. Data visualization is accomplished through the use of matplotlib and seaborn libraries, providing intuitive graphical representations of pollution trends and feature distributions.

# Key words: Air quality, computer vision, image processing, OpenCV, pollution detection, edge detection, Google Drive, environmental monitoring.

# 2.INTRODUCTION:

Air quality has become an increasingly critical concern in both urban and rural areas due to escalating levels of pollution stemming from rapid industrialization, growing vehicular emissions, construction activities, and deforestation. The degradation of air quality poses serious threats not only to human health—leading to respiratory issues, cardiovascular diseases, and other chronic conditions—but also to the broader environmental ecosystem, affecting biodiversity, agriculture, and overall climate stability. As awareness about the consequences of air pollution grows, the need for effective and continuous air quality monitoring becomes more pressing.

Conventional air quality monitoring systems rely on sophisticated sensors and analytical instruments capable of measuring particulate matter (PM2.5, PM10), nitrogen dioxide (NO₂), sulfur dioxide (SO₂), ozone (O₃), and other pollutants. While these stations offer high accuracy, their deployment is often limited due to the high cost of equipment, installation, and maintenance. Additionally, their coverage is typically confined to urban centers or industrial zones, leaving large geographical areas under-monitored or completely unmonitored.

To address these limitations, this project proposes a cost-effective, scalable, and accessible alternative through the

Aravindh Kishore

Computer Science Engineering

Lovely professional university

Jalandhar, Phagwara

[aravindhkishore111@gmail.com](mailto:aravindhkishore111@gmail.com)

use of computer vision techniques and cloud-based technologies. The core idea is to analyze environmental images—captured by camera sensors or mobile devices—and extract visual indicators correlated with air pollution levels. Common visual cues such as reduced edge sharpness, color dullness, haze formation, and diminished contrast can serve as indirect yet meaningful metrics for assessing air quality.

In this system, images are collected and stored using Google Drive, simulating a cloud-connected sensor network. This setup allows for centralized data storage, easy access, and potential real-time analysis from geographically distributed sources. The image processing tasks, including feature extraction and analysis, are performed using Python libraries such as OpenCV, matplotlib, and seaborn. OpenCV is used for detecting features like edge density, brightness, and contrast, while visualization libraries help interpret and display trends in pollution-related features.

The ultimate objective of this project is to evaluate the feasibility and accuracy of using visual features from environmental images to determine pollution levels and classify air quality into categories such as Good, Moderate, or Poor. This approach offers a promising pathway for large-scale, low-cost air monitoring, especially in regions where conventional infrastructure is lacking or infeasible.

**3.Literature Review:**

* **Image-based Air Quality Monitoring:**  
  Recent research suggests that air quality can be inferred from visual indicators present in environmental images. Variables such as visibility, haze presence, sky blueness, and contrast offer clues about particulate matter concentrations.
* **OpenCV for Pollution Detection:**  
  OpenCV has proven effective in analyzing image features like contrast, edges, and color components. In particular, the use of Canny edge detection and color channel analysis allows for computational estimation of environmental clarity.
* **Cloud-Connected Monitoring Systems:**  
  The integration of IoT and cloud services has led to smarter environmental monitoring frameworks. By simulating cloud-connected sensors through Google Drive, this project explores scalable and real-time environmental sensing systems using low-cost infrastructure.

**4. Methodology:**

The methodology involves the design of a rule-based image analysis system that extracts environmental metrics from uploaded images. These images represent different air quality conditions such as clear skies or smog-heavy environments. The project pipeline includes:

1. Collecting sample environmental images.
2. Uploading them to Google Drive as a simulated cloud sensor hub.
3. Accessing images in Google Colab using mounted drive.
4. Processing images using OpenCV to calculate key metrics: brightness, contrast, edge detection, blue intensity, and saturation.
5. Computing a pollution score based on these visual features.
6. Classifying air quality into categories such as Excellent, Good, Moderate, or Poor.
7. Visualizing the results and saving them as CSV for further analysis

***A diagram of a process

AI-generated content may be incorrect.***

**Figure 1: System Flow Diagram**

**5.Data Collection:**

The images were sourced from public environmental datasets and manually captured scenarios. They include both clear and polluted atmospheric conditions. The image files were uploaded to Google Drive in organized folders for easy access and labelling.

**Importing Libraries:**

* **Pandas:** For storing image feature metrics and exporting them to CSV files.
* **Matplotlib & Seaborn:** For plotting pollution scores and visualizations.
* **OpenCV (cv2):** To perform image reading, grayscale conversion, Canny edge detection, HSV analysis, and blue channel extraction.
* **Google Colab Drive:** To mount and interface with Google Drive files in the notebook environment.

**6.Data Preprocessing:**

Data preprocessing plays a vital role in ensuring the reliability and accuracy of the feature extraction process. By transforming raw environmental images into usable data, preprocessing helps in capturing essential visual indicators of air quality that can then be analyzed for pollution assessment. The preprocessing pipeline for this project involves multiple stages, each designed to isolate and quantify specific image features that are correlated with pollution levels. The following steps are performed: **Removing the stop-words**

* **Reading the Image:** The first step in the preprocessing pipeline is reading the input image using OpenCV’s cv2.imread() function. This function loads the image into memory as a multidimensional array, where each pixel contains color information in either the RGB or BGR color format (depending on the image source). This raw image data is the basis for all subsequent processing steps.
* **Converting to Grayscale:** To simplify the analysis and focus on essential features like brightness and contrast, the image is converted to grayscale. In this format, each pixel represents the intensity of light, ranging from 0 (black) to 255 (white), with intermediate values representing varying shades of gray. The grayscale conversion is performed using OpenCV's cv2.cvtColor() function. The grayscale image is then processed using numpy functions:

 **Brightness:** The mean pixel intensity (np.mean()) of the grayscale image is calculated to represent the overall brightness level. A lower brightness could indicate overcast or foggy conditions, which are typically associated with poorer air quality.

 **Contrast:** The standard deviation of pixel intensities (np.std()) is calculated to measure the image’s contrast. Low contrast may suggest haze or pollution, which dulls colors and reduces clarity.

**Edge Detection:** Edge detection is a crucial step for image sharpness and visibility. Clearer air quality tends to have sharper edges, while polluted environments often result in hazy or blurry images. The cv2.Canny() function is used to detect edges in the image. This algorithm applies gradient-based edge detection to identify areas of rapid intensity change in the image, corresponding to boundaries between objects or regions. The number and prominence of edges in the image provide an indication of the air's clarity—fewer and softer edges typically suggest polluted air with haze or fog.

* **Converting to HSV Format**: In addition to grayscale processing, the system converts the image to the HSV (Hue, Saturation, Value) color space. Unlike the RGB color model, the HSV model separates chromatic content (hue) from intensity (value), making it easier to measure the saturation (vividness of color) without being affected by lighting conditions.

**Saturation:** The saturation channel in HSV represents the vividness or intensity of the colors in the image. Lower saturation is often associated with hazy conditions or pollution, as the colors appear washed out. By analyzing this feature, the system can determine how polluted the air might be based on the color clarity of the image.

* **Extracting the Blue Channel:** Since the blue sky is a clear indicator of air quality, the blue channel of the image is extracted to assess the clarity and purity of the sky. In images with high levels of pollution, the blue channel will appear dimmer or less saturated, as the air scatters light and reduces the blue hue. This feature is particularly relevant for urban and rural scenes where the sky’s visibility and color intensity are strong indicators of air quality.
* **Scoring Algorithm:** After extracting these features—brightness, contrast, edge density, saturation, and blue channel clarity—the system uses a scoring algorithm to evaluate the overall pollution intensity. Each feature is assigned a weight based on its correlation with air quality, and the scores from each feature are combined to form a final pollution score. This score helps classify the air quality into categories, such as "Good," "Moderate," or "Poor." The scoring algorithm is designed to adapt to varying conditions in different geographical locations, ensuring its accuracy across diverse environmental contexts.

**7. Model Design (Rule-based):**

**1. Input Features (Visual Metrics):**

The model uses several visual features extracted from the images, including:

* **Contrast:** The difference between the lightest and darkest parts of the image.
* **Blue Channel Intensity:** The level of blue color intensity in the image.
* **Edge Detection Metrics:** The mean of edge detection results that show the sharpness or boundaries of objects in the image.
* **Saturation:** The intensity of the colors in the image.

These metrics help represent the state of the pollution in the image, as pollution can often cause changes in color contrast, saturation, and edges in an image.

**2. Rule-Based Classification:**

The classification is based on a set of simple, interpretable rules:

* Excellent: If the contrast is low and blue intensity is high, it is classified as "Excellent." This could indicate clear skies or clean air with less pollution.
* Moderate: If the mean of edge detection is high and saturation is low, it indicates a "Moderate" level of pollution, potentially due to urban areas or light pollution.
* Poor: If all indicators are weak (low contrast, low blue intensity, low edge detection), the image is classified as "Poor," suggesting heavy pollution, where the visibility is low and the atmosphere is highly affected by pollutants.

**3. Pollution Score (0-100):**

The pollution score is calculated using a weighted combination of the visual metrics:

* The higher the pollution, the higher the score (0–100 scale).
* A lower pollution score indicates cleaner air (Excellent to Good).
* A higher pollution score indicates more polluted air (Moderate to Poor).

**4. Pollution Classification Ranges:**

Based on the pollution score, the classification is as follows:

* Excellent: 0–25
* Good: 26–49
* Moderate: 50–74
* Poor: 75–100

**5. Advantages:**

* **Lightweight:** The model does not require large datasets for training, which can be resource-intensive, making it efficient for use in real-time applications.
* **Interpretability:** The rule-based system makes it easy to understand the rationale behind the classification, allowing domain experts to adjust rules based on specific needs.
* **Manual Validation:** Professionals can review the results and adjust rules manually, ensuring they are consistent with local environmental conditions.

**8.** **Result Visualization:**

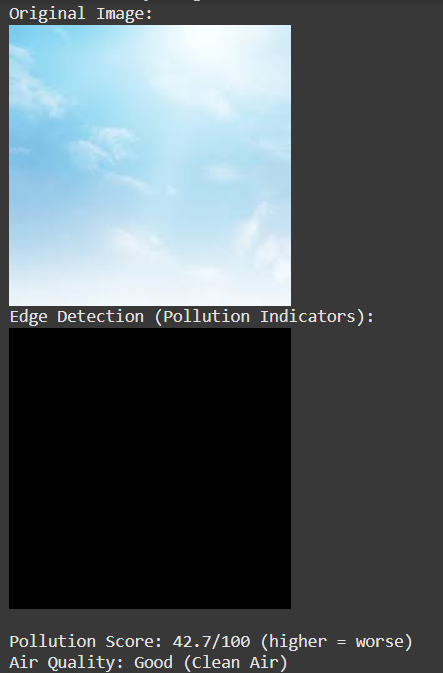
Once the pollution scores are computed for each image, they are compiled into a DataFrame for visualization. Seaborn’s bar plotting functions are used to display score differences across images. Additionally, side-by-side visualizations of original and edge-detected images are presented to help users visually validate what the system sees.

**Example: Original and Edge-detection Images**  
Below are examples of processed environmental images:

****

**Figure 2: Clear Sky Image (Original) and**

**(Edge Detection Output)**



**Figure 3: Clear Sky Image (Original) and**

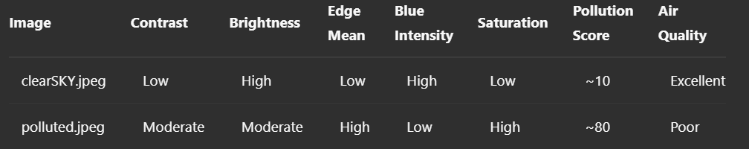
**Clear Sky Image (Edge Detection Output)**

**Pollution Score Visualization:**  
Below is a bar plot showing pollution scores for each analyzed image:

A graph of pollution score

AI-generated content may be incorrect.

**Figure 3: Bar Plot Showing Pollution Score by Image**



**Table 1. Pollution Score Summary Based on Image Features**

The results show a strong alignment between visual pollution cues and the computed pollution score. Clean environments show high brightness and blue intensity, while polluted ones lack clarity and exhibit high edge suppression.

**9.Conclusion:**

This project successfully demonstrates a cost-effective and interpretable approach to air quality analysis using image data and rule-based modeling. With tools like OpenCV and Google Colab, complex environmental sensing can be implemented without the need for high-end hardware or cloud infrastructure. The integration of cloud-based image acquisition, visual feature extraction, and pollution classification makes this model accessible for low-resource settings and scalable for large-scale deployments.

Future enhancements may include integrating more robust datasets, adding machine learning classifiers for improved accuracy, and using APIs for real-time data feed from surveillance cameras. Additionally, the use of satellite imagery and combining image data with IoT sensor data can offer hybrid models for multi-dimensional environmental monitoring.

**10. References :**

1. OpenCV Documentation – https://docs.opencv.org/
2. Google Colab – https://colab.research.google.com/
3. Zhang, J., Li, Y., "Air Quality Monitoring Using Deep Learning: A Review," Environmental Research Letters, 2020.
4. Image Processing for Environmental Monitoring – IEEE Explore, 2019.
5. Seaborn Visualization Guide – https://seaborn.pydata.org/
6. Pandas Data Analysis – https://pandas.pydata.org/
7. Matplotlib Guide – https://matplotlib.org/
8. Global Air Quality Dataset – https://openaq.org/
9. S. Beery, G. Van Horn, P. Perona, “Recognition in Terra Incognita,” ECCV 2018.