Predictive Maintenance using Smart Condition Monitoring

Capstone Project Proposal

Francis Chapoterera

Introduction

As more intelligent and automated systems are being developed across several industries such as mining, construction, manufacturing, and the automotive sector, the demand for systems that address maintenance challenges associated with both automated and manual machinery is evident. Organizations rely on various assets that are crucial to their operations, and these assets must be maintained in optimal working condition while ensuring maximum utilization without compromising safety or uptime.

Implementing a Smart Condition Monitoring (SCM) system can help companies remain competitive on a global scale by improving operational efficiency through increased uptime, extended equipment lifespan, and enhanced machine performance. By leveraging real-time sensor data, organizations can gain better insights into their assets, enabling proactive maintenance strategies and reducing unplanned downtime.

For this capstone project, I aim to develop an IoT-based Smart Condition Monitoring system. The system will utilize an ESP-32, a motor attached to a fan, and a nine-axis sensor consisting of a three-axis accelerometer, a three-axis gyroscope, and a three-axis magnetometer to comprehensively measure the machine's vibration and motion. Additionally, the ESP-32 camera module will serve as an image sensor, providing visual insights into the motor's condition. The nine-axis sensor will be mounted on the motor housing to capture precise vibration and motion data.

The collected sensor data will be used to develop a predictive machine learning model for classifying different operational states of the motor. This involves gathering data under various conditions, such as different speeds, obstructions, or complete stops, and interpreting these patterns for future improvements.

To enhance usability, I will develop an interactive system featuring widgets that display essential sensor data, allowing end-users to visualize machine performance in real time. The system will also generate alerts based on the severity of detected issues, enabling timely maintenance interventions. Additionally, the camera module will facilitate remote monitoring, allowing users to visually inspect the motor if an obstruction or malfunction is detected.

This solution is unique because it integrates multiple sensing modalities vibration analysis, motion tracking, and visual inspection into a single IoT-based platform. Unlike traditional condition monitoring systems that rely solely on vibration data, this approach enhances diagnostic accuracy by incorporating real-time imaging and multi-sensor fusion, providing a more comprehensive understanding of machine health. Furthermore, by leveraging edge computing on the ESP-32, the system reduces latency and bandwidth requirements, making it a cost-effective and scalable solution for industrial applications.

Project Description

Topology/Building Blocks

The Smart Condition Monitoring system is composed of several key components that work together to collect, process, and visualize data for predictive maintenance. The major building blocks of the system include:

1. **ESP-32 Microcontroller:** Serves as the core processing unit, facilitating communication between the sensors and the cloud platform.
2. **Nine-Axis Sensor (Accelerometer, Gyroscope, Magnetometer):** Captures motion and vibration data from the rotating motor's housing.
3. **ESP-32 Camera Module:** Provides visual monitoring capabilities to detect obstructions or operational anomalies.
4. **Motor with Attached Fan:** Represents the machinery under observation, simulating real-world industrial assets.
5. **Cloud Database & Server:** Stores sensor data for analysis and retrieval.
6. **Machine Learning Model:** Processes sensor data for predictive insights and anomaly detection.
7. **User Interface Dashboard:** Displays real-time sensor data, analytics, and alerts for remote monitoring and decision-making.

The ESP-32 microcontroller collects raw data from the nine-axis sensor and the ESP-32 camera module. The data is then transmitted to a cloud database for storage and processing. The machine learning model analyzes the incoming data to classify operational states and detect potential faults. The processed insights are visualized on a user interface dashboard, which also enables remote access to the live camera feed and provides alerts in case of anomalies. The system ensures real-time monitoring and predictive analytics to improve machine uptime and efficiency.

Flow of Information/Knowledge

1. The nine-axis sensor and ESP-32 camera collect vibration, motion, and image data from the motor.
2. The ESP-32 processes and transmits the raw data to the cloud server.
3. The cloud stores and forwards the data to a machine learning model for analysis.
4. The machine learning model classifies the operational state of the motor (e.g., normal operation, obstruction detected, motor failure, or irregular vibrations).
5. The processed insights and real-time data are displayed on the user interface for the end user.
6. Alerts are triggered and communicated via the interface or notifications when critical conditions are detected.
7. The camera module provides visual verification in cases where an obstruction or unexpected behavior is suspected.
8. Communication protocols:

- **MQTT (Message Queuing Telemetry Transport):** Enables lightweight, efficient communication between the ESP-32 and the cloud for real-time data transmission.
- **I2C (Inter-Integrated Circuit):** Facilitates high-speed, two-wire communication between the ESP-32 and the nine-axis sensor for motion and vibration data collection.

Unknowns

1. Effectiveness of the selected machine learning model in classifying different operational states with limited training data.
2. Variability in sensor data due to environmental factors (e.g., temperature, electromagnetic interference).
3. Optimal data transmission strategy for real-time performance without excessive latency.
4. The accuracy and usability of the user interface for non-technical end users.
5. Potential hardware limitations of the ESP-32 in handling high-frequency data collection and transmission.

Challenges

1. **Data Collection & Labeling:** Gathering sufficient labeled sensor data to train an accurate predictive model.
2. **Real-Time Processing:** Ensuring the system can process and transmit data with minimal delays.
3. **Energy Efficiency:** Managing power consumption for continuous operation of the monitoring system.
4. **Robustness & Durability:** Ensuring sensor stability and accuracy under different operational conditions.
5. **User Interface Optimization:** Designing an intuitive dashboard for effective data visualization and alerting.
6. **Integration & Deployment:** Seamlessly integrating hardware, software, and cloud components to form a cohesive system.

By overcoming these challenges, the Smart Condition Monitoring system will enable organizations to improve equipment longevity, reduce downtime, and enhance overall operational efficiency.

Tasks

1. **Research and Discovery**

    - Review literature on condition monitoring and predictive maintenance.
    - Investigate existing IoT-based condition monitoring solutions.
    - Identify suitable machine learning models for predictive maintenance.

2. **System Design & Hardware Setup**

    - Select and integrate ESP-32 with sensors and camera module.
    - Design hardware mounting setup for sensors.

3. **Software Development**

    - Develop firmware for ESP-32 to collect and transmit data.
    - Implement cloud-based data storage.
    - Develop a machine learning model for predictive analysis.
    - Create a user-friendly dashboard for real-time visualization.

4. **Testing & Iteration**

    - Collect and label training data.
    - Optimize machine learning model for accuracy.

- Test system under different operational conditions.

5. **Final Deployment & Documentation**

    - Deploy the system for real-time condition monitoring.
    - Document findings and prepare reports.

Schedule

- **Early March:** Submit initial project report (includes research findings and system design overview).
- **March 24:** Submit mid-cycle report (hardware setup, software implementation progress, and preliminary test results).
- **April 22:** Submit final report (full system functionality, test results, and project conclusions).