Jomo kenyatta University of Agriculture and Technology

College of Engineering and Technology

School of Mechanical, Materials, and Manufacturing Engineering

Department of Marine Engineering and Maritime Operations

Advance Condition Based Predictive Maintenance of Electric Motors in Ships

Kirimi Nathan

Kibichii Kieran

Supervisors

Dr.Gerald J. Odhiambo

Mr.Elijah M. Munyao

A project submitted in Partial Fulfillment of the Requirements for the degree of Bachelor of Science in Marine Engineering of The Jomo Kenyatta University of Agriculture and Technology

Abstract

The shipping industry is vital for the global economy, promoting trade, job growth, and resource accessibility. Despite sustainability challenges, it drives economic development and global connectivity. To ensure its positive impact, efforts to enhance sustainability and efficiency are crucial.

Electric motors are pivotal in the shift to electric ship propulsion, offering environmental advantages, cost savings, and improved performance. As maritime sustainability gains priority, electric propulsion systems will shape the industry's future.

This project focuses on characterizing and predicting motor failures. It's particularly significant for autonomous ships,whose viability benefits from advancements in information technology.

We have collected extensive data, facilitating condition-based monitoring (CBM), performance analysis, and the development of a comprehensive diagnostics framework. This framework not only serves as a rapid indicator of the ship's reliability but, through the implementation of a comparative algorithm, allows us to accurately diagnose mechanical faults and identify anomalies. This integrated approach enhances our ability to proactively address potential issues and maintain optimal ship performance.

Keywords: Efficiency, Condition Monitoring, Remaining Useful Life, Autonomous Ships, Performance.

Contents

List of Figures

1 Introduction

1.1 Background

1.1.1 Maintenance

As industries globally strive for increased optimization and efficiency, the remarkable surge in computational power and technological advancements has made it possible to automate "higher-level" tasks that traditionally demanded human intellect. This progress significantly advances the development of unmanned autonomous vessels within the maritime sector, bringing them closer to mass production.

The practicality of autonomous vessels relies on maintaining continual awareness of machinery performance and operational status within the engine room. Examination of industry practices reveals that reliability in this sector has historically depended heavily on trial-and-error testing procedures. Most reliability research in the industry focuses on two distinct phases of a product's life cycle. Firstly the warranty period, during which most failures result from product malfunctions or quality-related issues. Secondly wear-out period, where failures primarily occur due to excessive wear and use [1].

By utilizing sensors and logging software, equipment conditions are continually assessed, enabling efficient data analysis and the planning of predictive maintenance directly on vessels. The electric motor, as the primary device converting electrical energy into mechanical energy, finds extensive use in electric propulsion, powering thrusters for station keeping, and various onboard equipment across hundreds of ships. Notably, around 80-90% of load installations rely on electric motors for their operation [2].

Forward-thinking organizations understand that maintenance should no longer be perceived solely as an expense; it must be seamlessly integrated into the business cycle. This integration ensures predictability, fosters growth, and enhances overall operational quality. Shifting from scheduled rule-based maintenance to on-condition maintenance and ultimately adopting a data-driven risk-based approach enables more precise and timely maintenance tasks. This evolved maintenance perspective offers numerous practical advantages, including cost reduction, improved safety, and increased availability of ship systems.

The practice of making failure predictions and determining the remaining useful life (RUL) yields substantial benefits, including work-style improvements, reduced crew workload due to autonomous monitoring, heightened safety through accident prevention, and the assurance of efficient and optimal operations. Future plans include the modular addition of monitoring devises for a holistic view of the whole engine room to continually enhance overall performance.

In the marine industry, traditional preventive and corrective maintenance approaches still prevail. Mechanical systems, including plants, machinery, and equipment components, are often replaced or overhauled at fixed intervals, even when they remain functional. This practice can lead to unnecessary replacement, repair, or maintenance expenses. Alternatively, these components may exceed their operational lifetimes before scheduled maintenance, resulting in system breakdowns and necessitating corrective maintenance. Consequently, these conventional maintenance methods are proving less effective in maintaining the reliability, safety, and maintainability of marine mechanical systems.

Predictive maintenance can be conducted even when no prior failure data for equipment is available. When such data is accessible, machine-learning models based on binary classification are employed to predict impending failures, facilitating timely repair or equipment replacement planning [2].

There are several parameters to be considered:

- Temperature

- Pressure
- Vibration
- Current

1.2 Problem statement

Ship-related incidents often result in accidental errors, with a notable 62 percent attributed to human failure, specifically 22 percent linked to shipboard operations (Bratic et al., 2019). Engine room failures, a significant contributor to these incidents, can be traced to natural mechanical failures, electrical malfunctions, and human errors stemming from negligence or inadequate competency in engine room procedures. These failures garner heightened attention when they lead to severe consequences, such as oil spills causing environmental damage and the tragic loss of crew members' lives. Beyond the human and environmental toll, vessel owners also face substantial financial losses, including maintenance expenses and shipping costs, which could be allocated more efficiently.

While it is acknowledged that complete elimination of engine room accidents is unrealistic, this seeks to address the prevalent issues by introducing strategies that enhance efficiency and reduce downtime. Specifically, the proposal focuses on minimizing human errors through the automation of engine room motors and condition monitoring, aiming to achieve a level of autonomy that aligns with Level 5 standards, i.e(Hands-off, eyes-off, mind-off = human-off). This approach not only addresses the root cause of a significant portion of accidents but also aligns with the broader industry shift towards autonomous operations, ultimately promoting safety, environmental sustainability, and cost-effectiveness in maritime activities.[3]

1.3 Objectives

1.3.1 Main Objective

To monitor the health of ship motors thus improving reliability and preventing downtime in ships.

1.3.2 Specific Objectives

1. To develop a condition monitoring framework for electric motors in ships.
2. Extraction of motor fault characteristics and signatures.
3. TO develop an algorithm that autonomously diagnoses motor faults.

1.4 Justification of the study

Predictive maintenance offers a proactive approach to managing electric motors, crucial components in any facility. These motors are prone to various issues leading to disruptions, reduced productivity, and financial impact. By transitioning from manual inspections to automated methods, productivity increases, and errors decrease. Traditional maintenance practices, often involving premature replacements, can't entirely prevent unexpected downtime. The primary objective of predictive maintenance is to provide early alerts for potential motor issues, ensuring prolonged operational life. In the maritime context, motor failures lead to extensive downtime, impacting profitability. Advanced monitoring detects issues in advance, accurately determines repair times, and mitigates unexpected shutdown risks. Predictive maintenance, while complementing reactive and preventive approaches, represents a significant leap of up to 50% forward in asset management, enhancing efficiency and minimizing disruptions for improved financial outcomes. [4].

2 Literature Review

2.1 Operation, Subsystems and Parameters

Every day, our reliance on various machines is indispensable, but inevitably, these machines will face breakdowns unless they undergo proper maintenance.

Efficient engine room performance is crucial for the control, monitoring, and maintenance of ship equipment. Sensors and actuators play a pivotal role in the operation of machinery like pumps and generators. It is imperative that these components remain in optimal working condition. To ensure this, constant monitoring takes place, and various maintenance types are implemented. Corrective maintenance is carried out when there is a critical equipment failure, leading to unplanned downtime. Scheduled maintenance is performed regularly, involving thorough checks and necessary replacements to prevent unforeseen disruptions. The industry has embraced condition-based maintenance, utilizing predictive equipment status to plan maintenance activities [5].

Condition monitoring, a form of maintenance inspection, involves continuously monitoring operational assets and analyzing the acquired data. This analysis aims to detect signs of degradation, diagnose faults, and predict the remaining safe or economical lifespan of the equipment. Five general categories of Condition Monitoring techniques include vibration monitoring and analysis, visual inspection and non-destructive testing, performance monitoring and analysis, analysis of wear particles in lubricants and contaminants in process fluids, and electrical plant testing. Effective condition monitoring relies on high-quality data, often obtained through carefully conducted tests. Valuable information can be gleaned from a plant's permanent instrumentation, particularly when repeatability is established [6]. The fundamental principle of condition monitoring is to identify deterioration by taking physical measurements at regular intervals. Condition-based monitoring is employed to inspect and replace systems based on observed deterioration levels.

Predictive maintenance involves harnessing data, machine learning techniques, and statistical algorithms to anticipate potential system failures. Monitored machinery data collected through sensors, whether wired or wireless, undergoes analysis to consistently forecast when specific components or machinery should be replaced or maintained. This predictive approach optimizes maintenance costs and minimizes downtime, positioning itself as a subset of condition-based monitoring (CBM). CBM is a maintenance practice that tolerates and detects system failures during operation through continuous monitoring [7].

Marine industry professionals emphasize that the impact of increased sensor and cable installations for monitoring marine systems is contingent on the synchronized application of various maintenance aspects. Visual inspections, vibration measurements, regular oil analysis, and supplementing the maintenance information base with performance parameters are crucial for maximizing the effectiveness of predictive maintenance strategies [7].

Remaining Useful Life (RUL) signifies the time a component has before experiencing functional failure. Predicting the RUL is crucial for minimizing catastrophic failures in both manufacturing and service sectors. In this context, real data from a normally functioning pump is acquired at different life stages using multiple sensors under various working conditions. Over time, the accumulated sensor data enhances the accuracy of predictive models, enabling near-perfect failure predictions.

Presently, the industry heavily relies on sensors for condition monitoring, enabling timely decision-making. The time between potential failure and functional failure offers a window for decision-making algorithms to recommend actions, aiming to eliminate or mitigate the anticipated failure. For instance, industrial motors' pressure and temperature conditions can be recorded and monitored, with data transmitted wirelessly to a centralized logging center. The current prototype, developed using open-source software and hardware, successfully identifies abnormal motor conditions by analyzing sensor input values exceeding predefined set-points. This integration enhances the industry's ability to proactively address and prevent potential failures [7].

The widespread integration of sensors in various industries enables data acquisition, which, when coupled with advanced analytical methods, holds the potential to significantly enhance and optimize production processes. This data not only serves to monitor the current state of processes and devices but also plays a pivotal role in predicting potential failure states. Predictive methods applied to sensory data representing the production process, including machine condition, facilitate early identification and prediction of faulty or hazardous states, as well as machine breakdowns.

This project's objective is to introduce a data-driven method for analyzing sensory data to predict machine failures or identify deteriorating conditions. The goal is to empower users to plan maintenance proactively and prevent unplanned downtime. The methodology is developed based on real-life data, with the anticipation that the implemented solution will find practical application in the industry.

Predictive maintenance relies on prognostic and health management technology, assuming that the remaining useful life of equipment can be predicted. However, the uncertainty inherent in prognostics introduces the possibility of incorrect decisions regarding the remaining time to failure.

Presently, the most promising maintenance strategy for various technical systems and production lines is predictive maintenance (PM), applicable to any system with a measurable deteriorating physical parameter such as vibration, pressure, voltage, or current [4].

Motors serve as the backbone of industry but are prone to degradation due to factors like extended operational periods, power supply variations, or harsh environmental conditions, leading to gradual permanent damage. Consequently, continuous monitoring of their condition is vital for sustaining operations and maintaining efficiency [2].

Reliability surveys in the industry, as identified by the Institute of Electrical and Electronics Engineers (IEEE) in 1997 [8], indicate that AC motor failures can be classified into five main categories:

- **Bearing:** 44%
- **Stator Winding:** 26%
- **Rotor:** 3%
- **Shaft:** 5%
- **Others:** 22%

Despite advancements in technology, these categories still provide valuable insights into the prevalent causes of AC motor failures. It is noteworthy that the replacement of defective bearings remains the most cost-effective solution among the identified failure causes. However, detecting bearing issues continues to be a challenging task, particularly in motors that operate continuously without interruption for analysis.

As of more recent developments, ongoing research and technological advancements in motor health monitoring have introduced innovative methods and tools for detecting and addressing these failure modes. These may include enhanced sensor technologies, data analytics, and condition monitoring systems, contributing to more effective and proactive motor maintenance practices in contemporary industrial settings.

2.2 Temperature Sensors

2.2.1 Temperature Sensor Types

A temperature sensor, commonly utilizing a thermocouple or a resistance temperature detector (RTD), serves the purpose of providing temperature measurements through an electrical signal. This device gauges the hotness or coolness of an object, and its types can be broadly categorized based on the mode of connection, namely contact and non-contact temperature sensors [9].

Contact sensors, exemplified by thermocouples and thermistors, establish direct contact with the object under measurement. Conversely, non-contact temperature sensors operate by measuring the thermal radiation emitted by the heat source. These sensors find applications in challenging environments such as nuclear power plants or thermal power plants.

For the specific requirements of this project, a contact-type temperature sensor was chosen due to its suitability for collecting data from the bearings.

2.2.2 Thermistors: Temperature-Sensitive Resistors

A thermistor, derived from the words "thermal" and "resistor," is a temperature-sensitive resistor commonly employed as a temperature sensor. Unlike standard resistors with minimized temperature coefficients, thermistors intentionally possess a high temperature coefficient. Most thermistors exhibit negative temperature coefficients (NTC), signifying a decrease in resistance with increasing temperature. There are also positive temperature coefficient thermistors, known as PTC thermistors [10].

The DS18B20, a compact temperature sensor with a built-in 12-bit analog-to-digital converter (ADC), is notable for its ease of integration with Arduino digital inputs. Operating on a one-wire bus, this sensor requires minimal additional components for communication [10]. The choice of the DS18B20 aligns with the project's objectives and facilitates efficient temperature data acquisition from the monitored bearings.

2.3 Current Sensors

2.3.1 Overview of Current Sensors

A current sensor serves the purpose of detecting and converting current into an easily measurable output voltage. This output voltage is directly proportional to the current flowing through the measured path.

When electric current flows through a wire or a circuit, two phenomena are harnessed for the design of current sensors: voltage drop and the generation of a magnetic field around the current-carrying conductor. As a result, two primary types of current sensing methods emerge: direct and indirect.

Direct Sensing: This method involves the measurement of the voltage drop associated with the current passing through passive electrical components. It is grounded in Ohm's law, where the relationship between current, voltage, and resistance is utilized for accurate sensing.

Indirect Sensing: In this approach, the measurement focuses on the magnetic field surrounding a conductor through which current passes. It is based on Faraday's and Ampere's laws, leveraging the principles of electromagnetic induction.

2.3.2 Pressure Transducers

Pressure transducers, available in various shapes and sizes, typically feature a cylinder-shaped center housing the diaphragm and the pressure measurement chamber. One end incorporates a pressure port, commonly threaded, bolted, barbed fitted, or open, while the other end facilitates signal transmission. These transducers play a crucial role in converting pressure variations into electrical signals for monitoring and control purposes. The diverse design options allow for flexibility in applications, catering to different pressure ranges and environmental conditions.

2.3.3 Vibration Accelerometer

A vibration accelerometer, a crucial sensor in engineering, industrial, and scientific applications, is utilized for measuring and detecting vibrations or accelerations in various objects or systems. One prevalent type of vibration accelerometer leverages the piezoelectric effect, where certain materials generate an electric charge when subjected to mechanical stress. This type of accelerometer converts mechanical vibrations into electrical signals, facilitating measurement and analysis.

Vibrational analysis, applied for condition monitoring and prognostics of electric motors, provides valuable insights into motor health and performance. Changes in the vibration signature of an electric motor can indicate various faults, including bearing wear, misalignment, or rotor imbalance. For instance, bearing wear might manifest as increased vibration amplitude, while misalignment can result in irregular frequency patterns.

In a noteworthy study conducted by Chen and Wang (2017), advanced signal processing techniques were employed for motor prognosis using vibrational analysis. Their method, based on wavelet packet decomposition and envelope analysis, effectively extracted fault-related features from motor vibration signals. This approach demonstrated promise in detecting faults at an early stage, enhancing the potential for proactive maintenance.

Furthermore, the integration of machine learning algorithms with vibrational analysis has been explored to improve prognosis accuracy. Support vector machines and principal component analysis have been successfully employed to classify vibration signals into different fault categories. This integration not only aids in precise fault identification but also contributes to the prediction of impending motor failures.

In practical applications, continuous monitoring of vibrations with accelerometers allows for the establishment of baseline patterns and the identification of deviations that may indicate developing issues. This proactive approach enhances overall motor reliability and minimizes unplanned downtime, emphasizing the importance of vibrational analysis and advanced signal processing in predictive maintenance strategies.

3 Methodology

3.1 Overview

In this project, we undertook the integration of various sensors to enhance predictive maintenance capabilities. The key components and methodologies involved in each sensor type are summarized below:

- **Temperature Sensors:**
    1. Identification of specific data requirements for bearing temperature.
    2. Selection of contact-type temperature sensor.
    3. Integration into the data acquisition system.
    4. Calibration and rigorous testing to ensure accuracy.
- **Current Sensors:**
    1. Comprehensive understanding of current sensing principles.
    2. Selection of appropriate current sensor type.
    3. Incorporation into the system architecture.
    4. Validation and testing under various conditions.
- **Vibration Accelerometers:**
    1. Deep understanding of vibration accelerometer operation.
    2. Selection of suitable accelerometer type utilizing the piezoelectric effect.
    3. Integration into the system and implementation of signal processing techniques.
    4. Exploration and implementation of machine learning algorithms.

5. Establishment of continuous monitoring for proactive maintenance.

The successful execution of these methodologies has significantly enhanced our system's capabilities for efficient data acquisition, analysis, and proactive maintenance in industrial applications.

3.2 Data Collection

The systematic collection of temperature and current data from a motor involved creating a time-ordered series of data points. Measurements of temperature and current were meticulously taken at regular intervals to establish a comprehensive dataset for analysis and monitoring.

3.2.1 Temperature Sensor

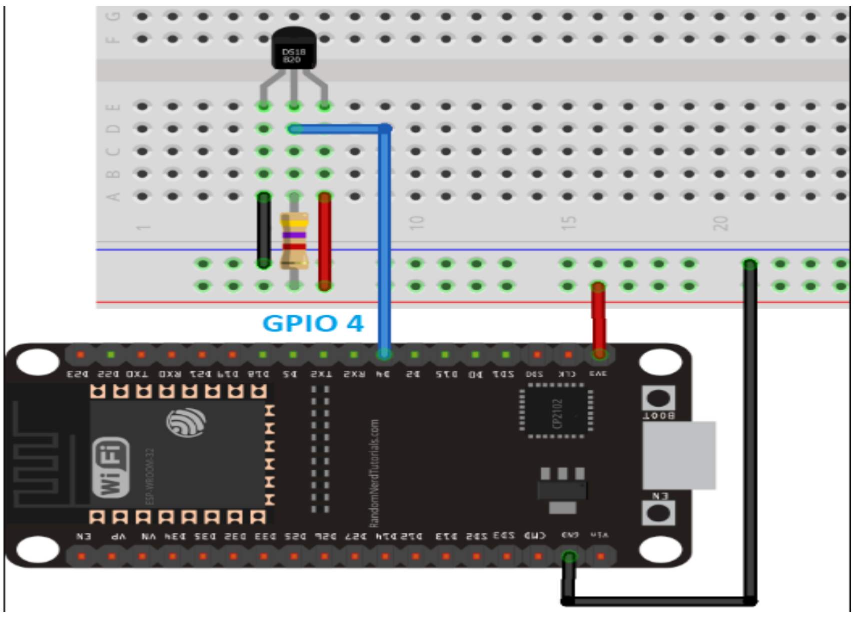


Figure 3.1: ds18b20 Pin Out

To initiate a temperature measurement and conversion, a command for reading and conversion is issued to the ds18b20 sensor. Following the conversion process, the resulting thermal data is stored in the 2-byte temperature register in the scratchpad memory. It is crucial to note that for the execution of this command, an external power supply is utilized to specify read time slots, ensuring the reliability and accuracy of the temperature readings.

3.2.2 Current Measurement

In parallel with temperature data collection, current measurements were acquired using appropriate sensors designed for the motor's electrical system. These sensors provided real-time information on the electrical load and power consumption, contributing essential data for comprehensive motor health analysis.

The combination of temperature and current data forms the foundation for subsequent analysis, enabling proactive maintenance and effective decision-making regarding the motor's operational status.

```
void loop() {
   sensors.requestTemperatures();
   float temperatureC = sensors.getTempCByIndex(0);
   float temperatureF = sensors.getTempFByIndex(0);
   Serial.print(temperatureC);
   Serial.print(temperatureF);
   Serial.println("F");
   delay(5000);
}
```

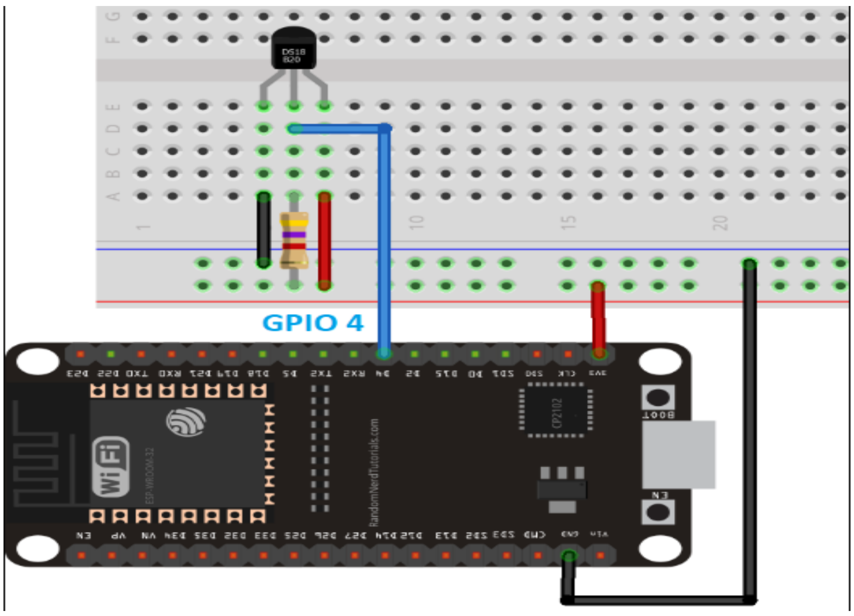


Figure 3.2: current sensor pin out

Current flows through the onboard hall sensor circuit in its IC The hall effect sensor detects the incoming current through its magnetic field generation Once detected, the hall effect sensor generates a voltage proportional to its magnetic field that's then used to measure the amount of current

3.3 Vibration Analysis

In order to ascertain the natural frequency of a system and predict potential resonance occurrences, a comprehensive understanding of the system's mass, stiffness, and damping characteristics is essential. Resonance manifests when the excitation frequency aligns with the natural frequency of the system, resulting in a significant amplification of vibration amplitudes. The following outlines the methodology for determining natural frequency and identifying resonance conditions:

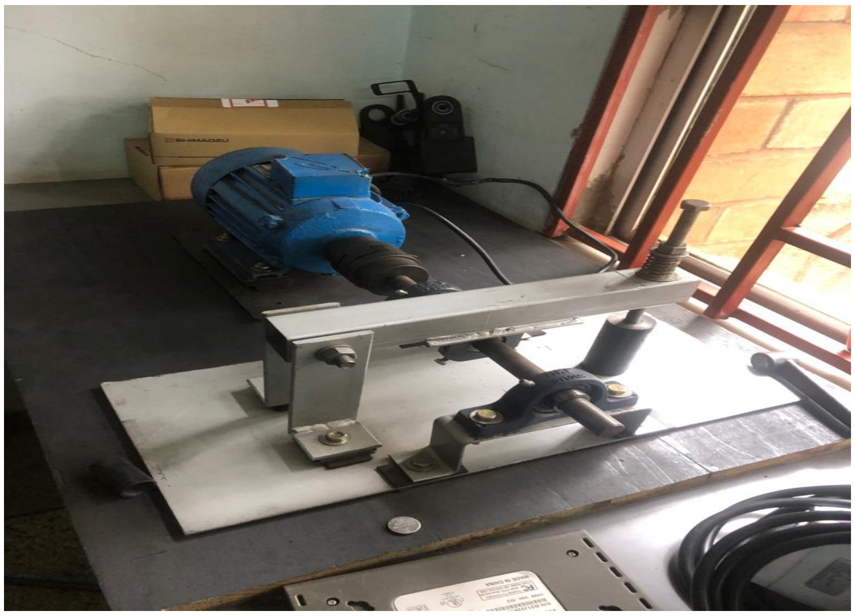


Figure 3.3: Vibration Sensor Setup

Firstly, the mechanical system under consideration is meticulously modeled, with a precise definition of the mass, stiffness, and damping properties of its components. This representation can be achieved through mathematical equations or software tools such as finite element analysis (FEA), ensuring a comprehensive understanding of the system's dynamic behavior.

Subsequently, an eigenvalue analysis or modal analysis is conducted on the system model.

This analytical process calculates the natural frequencies and mode shapes of the system, elucidating the vibration patterns associated with each natural frequency. The identification of the natural frequency linked to the mode of interest is pivotal, as it determines the specific vibration behavior under examination.

Furthermore, recent advancements in vibration analysis techniques involve the utilization of machine learning algorithms to enhance the accuracy of resonance prediction and provide insights into complex vibration patterns. This integration of data-driven approaches contributes to a more nuanced understanding of system dynamics and aids in the proactive identification of potential resonance challenges in real-world applications.

```
// The on-board ADC is 10-bits
// Different power supply will lead to different reference sources
// example: 2\^{}10 = 1024 -> 5V / 1024 \~{}= 4.88mV
//         unitValue= 5.0 / 1024.0*1000 ;
float unitValue= RefVal / 1024.0*1000 ;
float voltage = unitValue * sensorValue;

//When no load,Vref=initialValue
SERIAL.print(``initialValue: '');
SERIAL.print(voltage);
SERIAL.println(``mV'');

// Calculate the corresponding current
float current = (voltage - Vref) * sensitivity;

// Print display voltage (mV)
// This voltage is the pin voltage corresponding to the current
/*
voltage = unitValue * sensorValue-Vref;
```

```
SERIAL.print(voltage);
```

3.3.1 Omnimonitor

The device is responsible for keeping track of and managing data from different sources as well as managing read and write signals from the sensor to the remotely accessible server Both ds18b20 and DHT22 provide the output of temperature and humidity in the

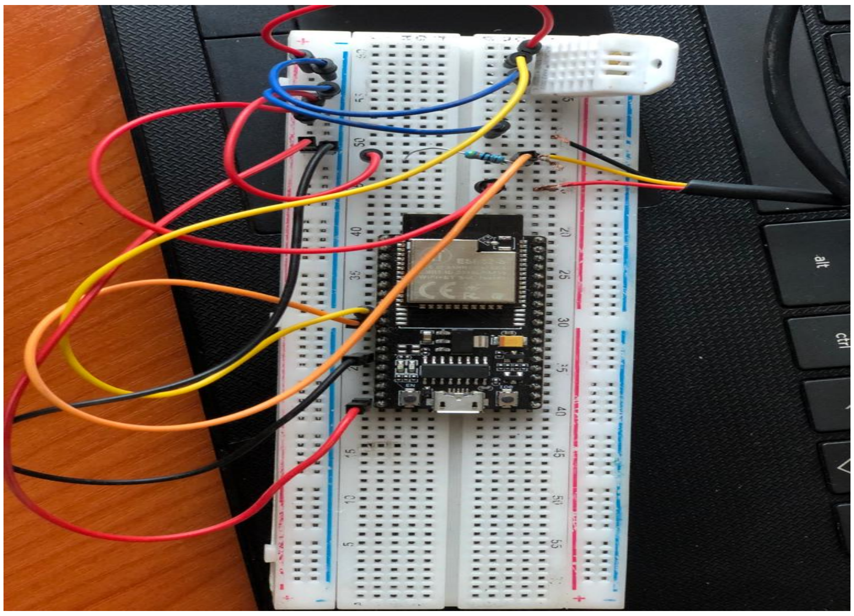


Figure 3.4: combined sensor setup

complex digital output format which can not be directly read with GPIO pins without writing any technique which can read these output signals. These sensors provide data through a single wire two-way communication protocol. A single process communication consists of 40 bits.To get values of temperature and humidity we call a number of created functions.

```
\#if defined(ESP32)
```

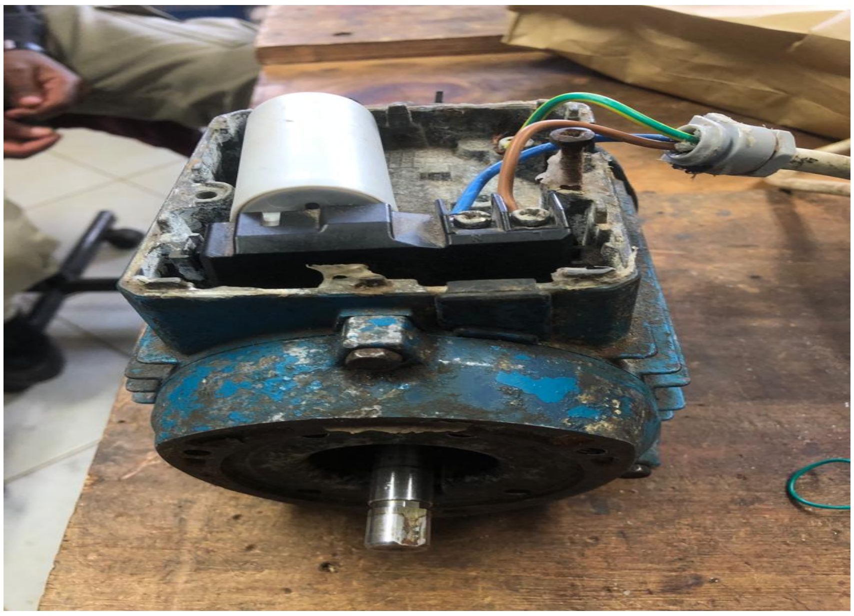


Figure 3.5: sensor setup

```
\#include <WiFiMulti.h>

WiFiMulti wifiMulti;

\#define DEVICE ‘‘ESP32’’

\#elif defined(ESP8266)

\#include <ESP8266WiFiMulti.h>

ESP8266WiFiMulti wifiMulti;

\#define DEVICE ‘‘ESP8266’’

\#endif



//\#define DHTTYPE DHT11 // DHT 11

//\#define DHTTYPE DHT21 // DHT 21 (AM2301)

\#define DHTTYPE DHT22 // DHT 22 (AM2302), AM2321


// ds18b20 sensor
```

```
\#define ONE\_WIRE\_BUS 26
OneWire oneWire(ONE\_WIRE\_BUS);
DallasTemperature sensors(\&oneWire);

uint8\_t DHTPin = 25;
DHT dht(DHTPin, DHTTYPE);

float temperature\_Celsius; // ambient
float humidity;
float temperature\_Celcius2; // contact
```

3.3.2 MQTT

Message Queuing Telemetry Transport (MQTT) has proven indispensable in my project focused on predictive maintenance, particularly when dealing with the intricate task of monitoring sensor data from a ship's motor. Operating in maritime environments, where network reliability is often challenged and resources are constrained, MQTT's lightweight messaging protocol serves as a crucial element in establishing efficient machine-to-machine (M2M) communication and seamless integration within the Internet of Things (IoT).

In the implementation phase, We connect my predictive maintenance system to an MQTT broker, which functions as the central hub for communication. This broker efficiently manages the flow of sensor data from the ship's motor, forwarding it to my monitoring and analysis tools. The PUSH/SUBSCRIBE architecture ensures a well-organized and scalable communication framework, a fundamental aspect for real-time monitoring and predictive analysis of the motor's performance.

What sets MQTT apart in this predictive maintenance project is its optimization of data transmission. The protocol's well-defined, compact message structure, featuring a fixed 2-byte header, significantly reduces the communication overhead. In the context of maritime operations, where bandwidth may be limited and latency is a concern, this streamlined approach enhances the speed and responsiveness of data exchanges. The support for Quality of Service (QoS) levels further allows me to fine-tune the balance between message delivery assurance and network performance, a critical consideration for ensuring the timely and reliable transmission of sensor data.

Having successfully implemented MQTT in this predictive maintenance scenario, We've experienced firsthand its effectiveness in navigating through the challenges of low-bandwidth conditions and intermittent connectivity. The protocol plays a pivotal role in facilitating the timely detection of anomalies and potential issues in the ship's motor, enabling proactive maintenance interventions. As the maritime industry increasingly embraces IoT technologies for predictive maintenance, MQTT remains a cornerstone in my toolkit, providing a robust solution for handling sensor data and fostering efficient communication in the dynamic and interconnected environment of predictive maintenance systems for ship systems.

3.3.3 OTA

Instead of needing the user to connect the ESP32 to a computer via USB to execute the update, OTA programming allows update of new data to the ESP32 via Wi-Fi.

When there is no physical access to the ESP module, the OTA capability comes in handy. It helps to cut down on the time spent on each ESP module during maintenance.

One of the most useful features of OTA is that it allows a single central location to deliver an update to many equipment on the same network ideally all monitored equipment in a ship engine room.

3.3.4 Declaration, definition and initialization

Define, declare variables and initialize classes.

```
#define CURRENT_VERSION "1.0.0" // The current version of the firmware
    running
#define DOWNLOAD_URL SERVER_URL // The server url where we will get the
    latest firmware version
#define DHTPIN 2 // Digital pin connected to the DHT sensor
#define DHTTYPE DHT11 // DHT 11
#define LED 4 // Digital pin connected to the inbuilt LED
WiFiClient espClient; //Initializing the WiFiClient
void setup_wifi();
void reconnect();
PubSubClient client(espClient); // Initializing the PubSubClient
WebServer server(80); // Initializing the WebServer
// functions to take care of OTA firmware update
void handleRoot();
String getDownloadUrl();
bool downloadUpdate(String);
DHT_Unified dht(DHTPIN, DHTTYPE);
int ledState = LOW; // Set the ledstate to be off at the first instance
const long interval = 1000; // This is the interval we will be using to
    check for a new version
unsigned long previousMillis = 0; // previous timings in milliseconds
unsigned long currentMillis; // current timings in milliseconds
bool success; // download success
```

3.3.5 Setup function

This is executed only once in the beginning of the program.

```
void setup() \{
Serial.begin(115200); // begin Serial at 115200 baud rate
setup\_wifi(); // function to setup up wifi
espClient.setServer(mqtt\_server, 1883); //Setup the MQTT server

// Initialize device.
dht.begin();
// Print temperature sensor details.
sensor\_t sensor;
dht.temperature().getSensor( \& sensor);
Serial.println(F(‘));
Serial.println(F(‘‘Temperature Sensor Present’’));

// Print humidity sensor details.dht.humidity().getSensor(\&sensor);
Serial.println(F(‘‘Humidity Sensor Present’’));
Serial.setDebugOutput(true); // Set debug to true in order to print more
    serial output
pinMode(LED, OUTPUT); // Initialize the inbuilt led as an output device
delay(3000); // Set a delay of 3 seconds
Serial.println(version);
// Setup Wifi Manager
WiFiManager wm;
WiFiManagerParameter versionText(version.c\_str());
wm.addParameter( \& versionText);
if (!wm.autoConnect()) \{
Serial.println(‘‘failed to connect and hit timeout’’);
ESP.restart();
delay(1000);
\}
// Check if we need to download a new version
String downloadUrl = getDownloadUrl();
if (downloadUrl.length() > 0) \{
success = downloadUpdate(downloadUrl);
if (!success) \{
Serial.println(‘‘Error updating device’’);
\}
\}
server.on(‘‘/’’, handleRoot);
server.begin();
Serial.println(‘‘HTTP server started’’);
Serial.print(‘‘IP address: ’’);
Serial.println(WiFi.localIP());
\}
```

The `setup()` function in is a special function that is called once when the microcontroller starts. It is typically used for initialization tasks. Let’s break down the provided `setup()` function into paragraphs for better explanation:

Serial Communication Setup: The function `Serial.begin(115200)` initializes serial communication at a baud rate of 115200. This is a common setup for serial communication and is often used for debugging and communication with other devices.

Wi-Fi Setup: The `setup_wifi()` function is called to set up the Wi-Fi connection. Additionally, the MQTT (Message Queuing Telemetry Transport) server is configured using the `espClient.setServer()` function with the provided `mqtt_server` variable and port 1883.

DHT Sensor Initialization and Sensor Details: The DHT (Digital Humidity and Temperature) sensor is initialized with `dht.begin()`. Sensor details for both temperature and humidity are obtained using the `getSensor()` function, and messages indicating the presence of each sensor are printed to the serial monitor.

Debug Output and LED Initialization: Debug output is enabled for the serial monitor, allowing for more detailed debugging information. The built-in LED is initialized as an output device, and there's an optional delay of 3 seconds.

Version Information and Wi-Fi Manager Setup: The version information is formatted into an HTML string and printed to the serial monitor. The WiFi Manager library is used to facilitate easy setup of the Wi-Fi connection, and a parameter for displaying the version is added.

Wi-Fi Connection Handling and Update Check: The code attempts to connect to Wi-Fi using the WiFi Manager. If the connection fails, it restarts the device after a delay. It then checks if there is a new version available for download and updates the device if necessary.

HTTP Server Setup and Serial Output: An HTTP server is set up with a root ("/") endpoint and corresponding handler function (`handleRoot`). The server is started, and messages indicating its initiation and the local IP address are printed to the serial monitor.

In summary, the `setup()` function initializes serial communication, sets up Wi-Fi, configures MQTT, initializes and prints details of DHT sensors, enables debug output, initializes an LED, provides version information, sets up the Wi-Fi Manager, handles Wi-Fi connection, checks for updates, and sets up an HTTP server for device interaction. "‘

3.3.6 Loop function

This section is repeated after a specified interval to read and write parameters specified

```
// Get temperature event and print its value.
sensors_event_t event;
dht.temperature().getEvent( & event);
if (isnan(event.temperature)) {
  Serial.println(F("Error reading temperature!"));
} else {
  Serial.print(F("Temperature: "));
  Serial.print(event.temperature);


}
// Get humidity event and print its value.
dht.humidity().getEvent( & event);
if (isnan(event.relative_humidity)) {
  Serial.println(F("Error reading humidity!"));
} else {
  Serial.print(F("Humidity: "));
  Serial.print(event.relative_humidity);
  Serial.println(F("%"));
}
currentMillis = millis();
if (currentMillis - previousMillis >= interval) {
    previousMillis = currentMillis;
    ledState = ledState == LOW ? HIGH : LOW;
    digitalWrite(4, ledState);
  }
  // Just chill
  server.handleClient();
  delay(1000);
  char msg[200];
  if (!espClient.connected()) {
    reconnect();
  }
  StaticJsonBuffer < 300 > JSONbuffer;
  JsonObject & JSONencoder = JSONbuffer.createObject();
  JSONencoder["time"] = millis();
  JSONencoder["temperature"] = event.temperature;
  JSONencoder["humidity"] = event.relative_humidity;
  char JSONmessageBuffer[100];
  JSONencoder.printTo(JSONmessageBuffer, sizeof(JSONmessageBuffer));
  Serial.println(JSONmessageBuffer);
  if (espClient.publish("dht/user_id892", JSONmessageBuffer) == true) {
    Serial.println("Success sending message");
  } else {
    Serial.println("Error sending message");
  }
  espClient.loop();
}
```

3.3.7 Handle server requests

Ensures data is read only when it can be written to the server.

```
void handleRoot() {
  server.send(200, "text/plain", "v" + String(CURRENT_VERSION));
}
```

3.3.8 Download links

```
#define SERVER_URL "http://
String getDownloadUrl() {
  HTTPClient http;
  String downloadUrl;
  String url = DOWNLOAD_URL;
  http.begin(url);
  // start connection and send HTTP header
  int httpCode = http.GET();
  // httpCode will be negative on error
  if (httpCode > 0) {
    // HTTP header has been send and Server response header has been
        handled
    if (httpCode == HTTP_CODE_OK) {
      String payload = http.getString();
      Serial.println(payload);
      downloadUrl = payload;
    } else {
      Serial.println("Device is up to date!");
    }
  } else {
  http.end();
  Serial.println(downloadUrl);
  return downloa
```

3.3.9 Download binary firmware

This is used to remotely update variables such as motor load and environmental conditions that may cause disturbances to the system.

```
/*
Download binary image and use Update library to update the device.
*/
/*
Download binary image and use Update library to update the device.
*/
bool downloadUpdate(String url) \{
HTTPClient http;

http.begin(url);

int httpCode = http.GET();
if (httpCode > 0) \{
// HTTP header has been send and Server response header has been handled
// file found at server
if (httpCode == HTTP\_CODE\_OK) \{
int contentLength = http.getSize();
Serial.println(‘‘contentLength : ’’ + String(contentLength));
if (contentLength > 0) \{
   bool canBegin = Update.begin(contentLength);
   if (canBegin) \{
   WiFiClient stream = http.getStream();
size\_t written = Update.writeStream(stream);
   if (written == contentLength) \{
   Serial.println(``Written : '' + String(written) + `` successfully'');
   \} else \{
   Serial.println(``Written only : '' + String(written) + ``/'' +
      String(contentLength) + ``. Retry?'');
   \}
   if (Update.end()) \{
   Serial.println(``OTA done!'');
   if (Update.isFinished()) \{
   Serial.println(``Update successfully completed. Rebooting.'');
   ESP.restart();
   return true;
   \} else \{
   Serial.println(``Update not finished? Something went wrong!'');
   return false;
   \}
   \} else \{
   Serial.println(``Error Occurred. Error \#: '' + String(Update.getError()));
   return false;
   \}
   \} else \{
   Serial.println(``Not enough space to begin OTA'');
   client.flush();
   return false;
   \}
```

```
\} else \{
Serial.println(``There was no content in the response'');
client.flush();
return false;
\}
\} else \{
return false;
\}
\} else \{
return false;
\}
\}
```

3.3.10 Setup wifi function

```
void setup_wifi() {
  // Connecting to a WiFi network
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}
```

3.3.11 Reconnect function

```
void reconnect() {
  // Loop until we're reconnected
  while (!espClient.connected()) {
    // Attempt to connect
    if (espClient.connect("SES_DHT", MQTT_USER, MQTT_PASS)) {
      Serial.println("connected");
    } else {
      Serial.print("failed, rc=");
      Serial.print(espClient.state());
      Serial.println(" try again in 5 seconds");
      delay(5000);
    }
  }
}
```

3.4 Data Analytics Process Steps

The motor sensors output digital and analogue electrical signals. This is parsed to literal temperature and current values i.e degrees and amperes respectively. This is automated by use of a C bash script that then data is written into a CSV file that is now the accessible database for the algorithm fetch requests.

The data is prepared for analysis by wrangling to remove unwanted and redundant values. The stage is now set to visualize the data to be able to observe meaningful patterns that can provide useful insights to the motors condition. Ultimately this will also provide the logical descriptive equations on which predictive algorithms will learn from.

To prove meaningful results tests were developed to check if the output is in line with the expected results.

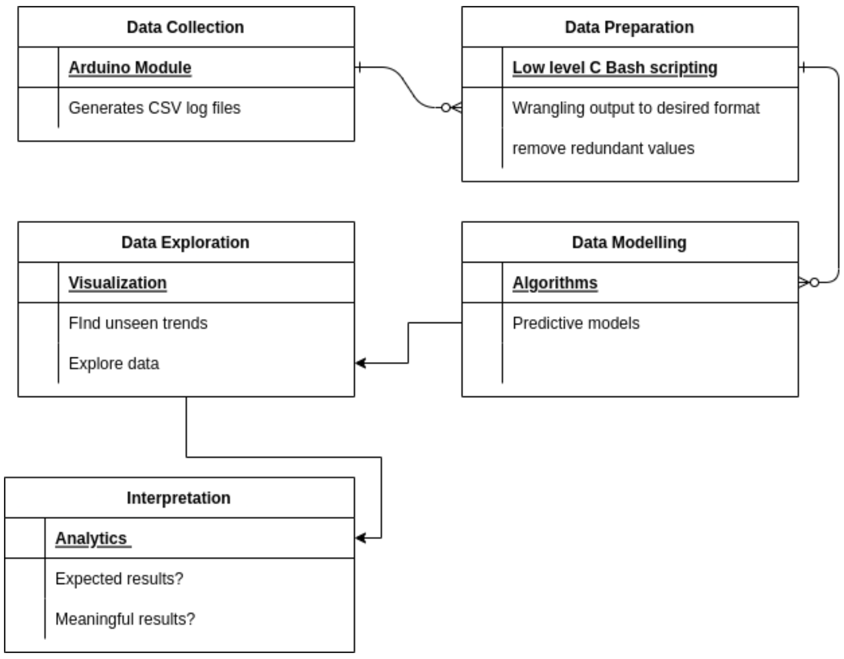


Figure 3.6: Data Analytics

3.5 Data Analysis

The information gathered from the sensors will undergo scrutiny through tailored scripts. This analytical process will involve the generation of graphs aimed at evaluating and contrasting the performance of individual system components.

3.5.1 Descriptive Analysis

Before delving into the primary analysis, an exploratory data analysis will be conducted to ascertain the historical and current operational statuses of the components.

3.5.2 Predictive Analysis

Predictive analysis will be executed through the development of models, enabling the anticipation of potential issues. This includes the ability to forecast the need for replacement parts and estimate the time required for maintenance activities.

3.5.3 Prescriptive Analysis

The prescriptive analysis phase aims to provide strategic insights for achieving specific objectives. For instance, it may recommend measures to reduce the load by 5% ultimately extending the service life of the components. This proactive approach is geared towards optimizing performance and ensuring the desired outcomes are achieved.

3.6 Sensor Placement

In the initial stages, We meticulously select suitable sensors and determine their optimal positions on the motor. The sensors are strategically placed to capture data from various motor components, including the stator, rotor, and bearings.

3.7 Data Acquisition

We collect data from the motor during its operation, utilizing accelerometers or other specialized sensors. To ensure data accuracy, We sample at a sufficiently high frequency, allowing for the comprehensive capture of the desired frequency range.

3.7.1 Preprocessing

After data collection, We apply various preprocessing techniques to the raw data to eliminate noise and artifacts. This involves filtering, normalization, and signal conditioning to enhance the overall quality and reliability of the data.

3.7.2 Feature Extraction

Next, We extract relevant features from the preprocessed data. These features encompass a spectrum of characteristics, including time-domain features like RMS value and crest factor, as well as frequency-domain features such as spectral analysis and power spectral density. Advanced techniques like wavelet analysis and empirical mode decomposition may also be employed for extracting fault-related features.

3.7.3 Fault Diagnosis

In the subsequent phase, We employ pattern recognition and machine learning algorithms to categorize the extracted features into different fault categories. To ensure the accuracy of the fault diagnosis models, the algorithms undergo training using labeled data derived from known fault conditions.

3.7.4 Prognosis and Remaining Useful Life (RUL)

Utilizing regression analysis and prognostic algorithms, We estimate the remaining useful life of the motor. This involves tracking the degradation patterns of the motor over time and predicting potential failure points based on the extracted features and historical data.

3.7.5 Prognostic Reporting

In the final steps, We generate comprehensive reports summarizing diagnosed faults, estimated remaining useful life, and recommended maintenance actions. These reports serve as valuable tools for maintenance personnel, aiding in the planning and execution of appropriate maintenance strategies, such as scheduling repairs or replacing the motor.

4 Results and Discussions

Three similar motors were chosen at different stages of their useful life:

- Motor1: New motor at max value of remaining useful life ie TIME = t
- Motor2: Motor at remaining useful life is approximately t/2
- Motor3: Motor at remaining useful life = 0

Data collected was returned as in the table below, indexed in time series data points that can often illustrate trends or patterns in a more accessible, intuitive way.

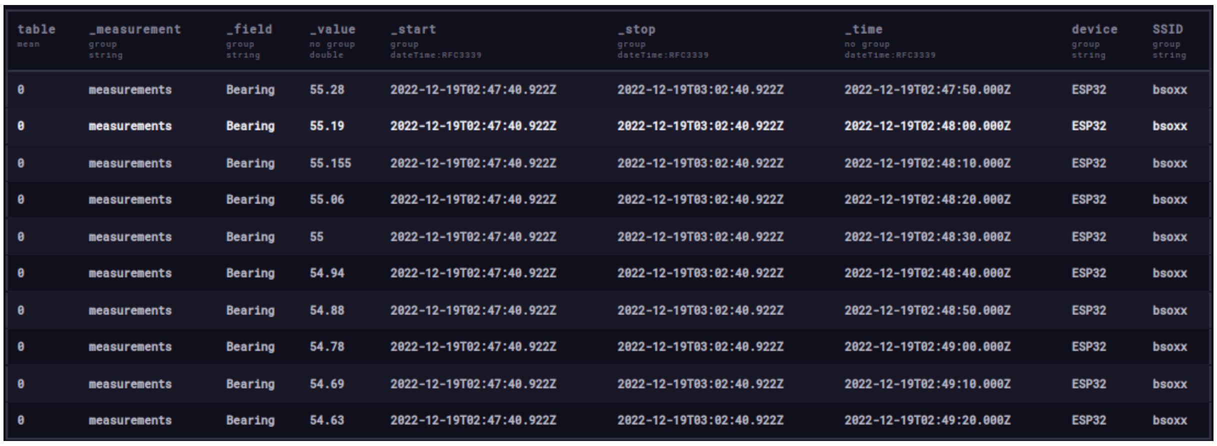


table mean	_measurement group string	_field group string	_value no group double	_start group dateTime:RFC3339	_stop group dateTime:RFC3339	_time no group dateTime:RFC3339	device group string	SSID group string
0	measurements	Bearing	55.28	2022-12-19T02:47:40.922Z	2022-12-19T03:02:40.922Z	2022-12-19T02:47:50.000Z	ESP32	bsoxx
0	measurements	Bearing	55.19	2022-12-19T02:47:40.922Z	2022-12-19T03:02:40.922Z	2022-12-19T02:48:00.000Z	ESP32	bsoxx
0	measurements	Bearing	55.155	2022-12-19T02:47:40.922Z	2022-12-19T03:02:40.922Z	2022-12-19T02:48:10.000Z	ESP32	bsoxx
0	measurements	Bearing	55.06	2022-12-19T02:47:40.922Z	2022-12-19T03:02:40.922Z	2022-12-19T02:48:20.000Z	ESP32	bsoxx
0	measurements	Bearing	55	2022-12-19T02:47:40.922Z	2022-12-19T03:02:40.922Z	2022-12-19T02:48:30.000Z	ESP32	bsoxx
0	measurements	Bearing	54.94	2022-12-19T02:47:40.922Z	2022-12-19T03:02:40.922Z	2022-12-19T02:48:40.000Z	ESP32	bsoxx
0	measurements	Bearing	54.88	2022-12-19T02:47:40.922Z	2022-12-19T03:02:40.922Z	2022-12-19T02:48:50.000Z	ESP32	bsoxx
0	measurements	Bearing	54.78	2022-12-19T02:47:40.922Z	2022-12-19T03:02:40.922Z	2022-12-19T02:49:00.000Z	ESP32	bsoxx
0	measurements	Bearing	54.69	2022-12-19T02:47:40.922Z	2022-12-19T03:02:40.922Z	2022-12-19T02:49:10.000Z	ESP32	bsoxx
0	measurements	Bearing	54.63	2022-12-19T02:47:40.922Z	2022-12-19T03:02:40.922Z	2022-12-19T02:49:20.000Z	ESP32	bsoxx

Figure 4.1: Dataset

Time series data is unique because it is ordered and often exhibits serial dependence, where the value of a data point at one time is influenced by another data point at a different time. While all events happen in a sequence, time series data specifically focuses on events whose value is affected by the passage of time. This type of data can have a very high level of granularity, such as at the microsecond or nanosecond level, and the main focus is on how the values change over time.

To determine whether a given dataset is time series data, you should consider what is needed to uniquely identify a record in the dataset. Time series data consists of a series of data points that are collected at regular intervals over a period of time. These data points typically include a timestamp, which indicates when the data was collected, and one or more variables that represent the values being measured at that time. To determine whether a dataset is time series data, you will need to determine whether it includes timestamps and whether the data points are collected at regular intervals. If the dataset meets these criteria, it is likely time series data.

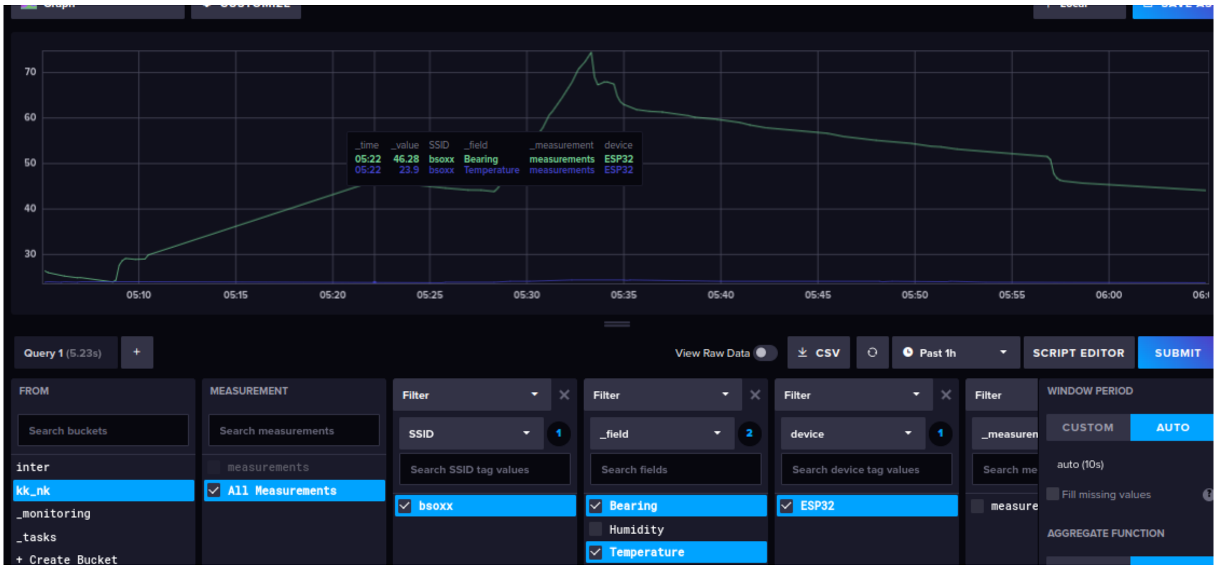
At this early point it is important that the parameters remain constant. While there is a variety of reasons that cause this fluctuations we can forecast long enough and accurate enough to respond to them and thus prevent downtime.

We then monitor the residuals and set alerts when the data deviates from the forecast as shown below

Forecasts are generated for each point of data in the algorithm, which can appear to be a copy of the raw data. However, if you examine the final data point in the top graph, you will see that the forecast predicts that the temperature will continue to drop taking a steep descent towards the bottom right of the graph. This prediction is a result of the recent decrease in temperature, which is indicated by the first anomaly.

General query format . Specify time constraints then filter network name and tags to return values required.

```
from(bucket: "kk_nk")
|> range(start: v.timeRangeStart, stop: v.timeRangeStop)
|> filter(fn: (r) => r["SSID"] == "bsoxx")
|> filter(fn: (r) => r["_field"] == "Temperature")
|> filter(fn: (r) => r["_measurement"] == "measurements")
|> aggregateWindow(every: v.windowPeriod, fn: mean, createEmpty: false)
|> yield(name: "mean
```



```
_time  _value  SSID   _field       _measurement  device
05:22  46.28   bsoxx  Bearing      measurements  ESP32
05:22  23.9    bsoxx  Temperature  measurements  ESP32
```

Figure 4.2: Data-query example

Motor performance is influenced by three factors: the voltage across the terminals, the resistance across the terminals, and the magnetic force. , we explore how temperature and current can affect these elements and modify motor conditions, using specific proven failure modes

The following graph shows a benchmark values of a motor running in pristine condition

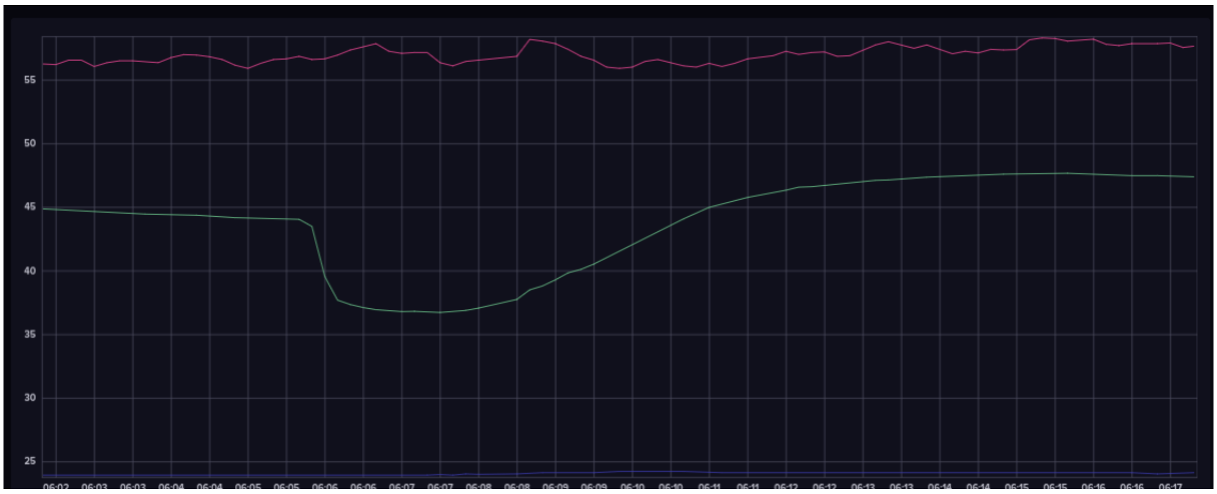


Figure 4.3: Motor1

Overheating motor: Quite conclusively Thermal effects on four performance characteristics of a motor:

No-load speed As the temperature increases, the magnetic force of the magnets becomes weaker, causing N0 to rise in inverse proportion.

No-load current The current changes in inverse proportion to the magnetic force of the magnets, but is also affected by the loss in bearing oil viscosity due to the rising temperature.

Stall current As the temperature rises, the winding resistance decreases, causing Is to fall in inverse proportion.

Stall torque As the temperature increases, the magnetic force of the magnets becomes weaker due to the decreasing current, causing Ts to fall greatly.

this is clearly shown at 70 + degrees where the motor shuts down until it cools down

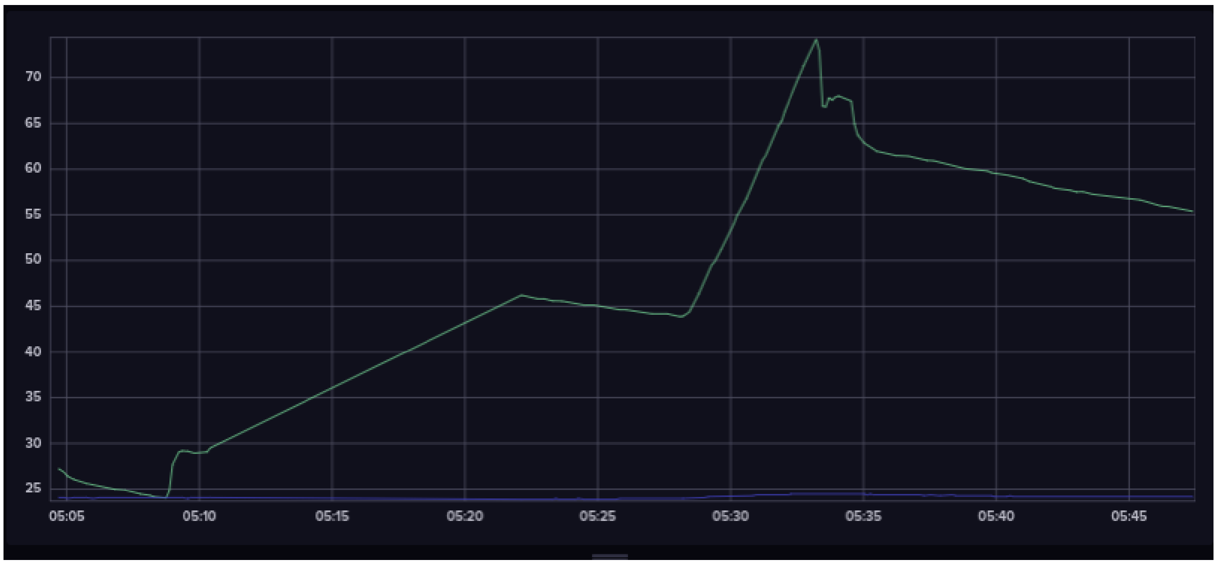
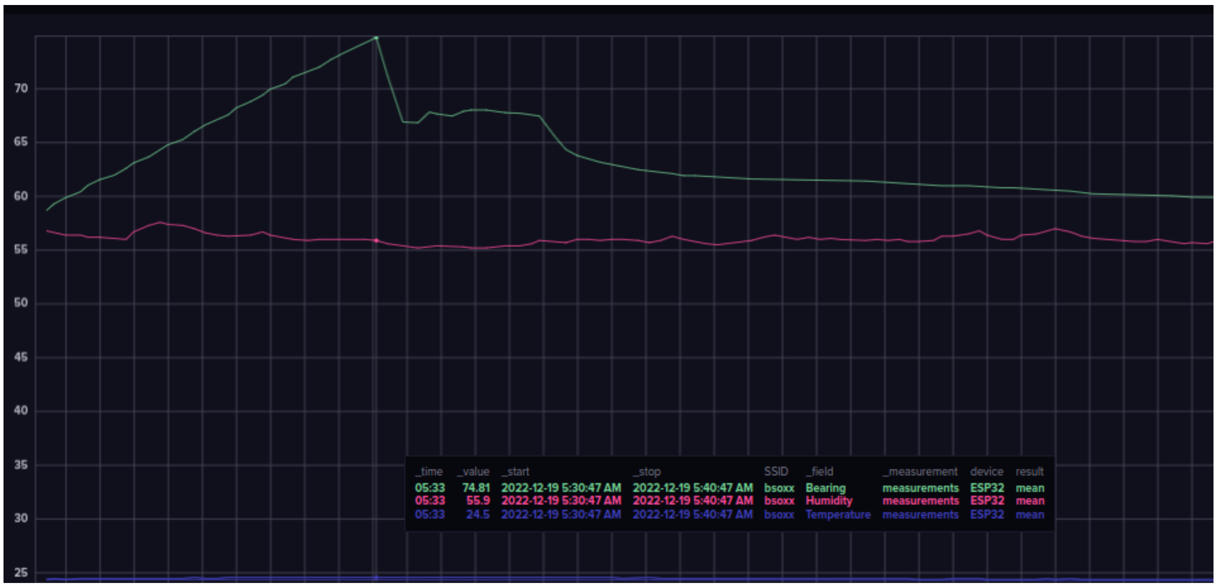


Figure 4.4: Motor3

The following graph shows the difference in operating conditions of similar motors but with different remaining useful life and how data brings out stark differences!



_time	_value	_start	_stop	SSID	_field	_measurement	device	result
05:33	74.81	2022-12-19 5:30:47 AM	2022-12-19 5:40:47 AM	bsoxx	Bearing	measurements	ESP32	mean
05:33	55.9	2022-12-19 5:30:47 AM	2022-12-19 5:40:47 AM	bsoxx	Humidity	measurements	ESP32	mean
05:33	24.5	2022-12-19 5:30:47 AM	2022-12-19 5:40:47 AM	bsoxx	Temperature	measurements	ESP32	mean

Figure 4.5: Motor comparison

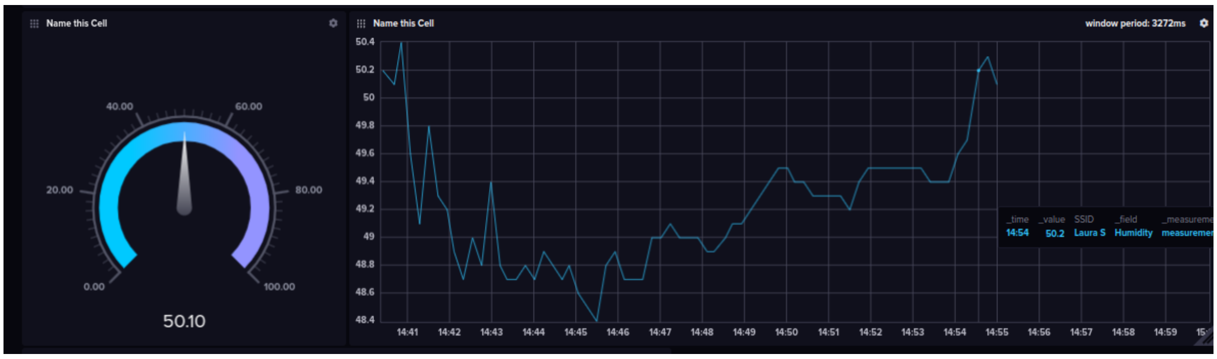


Figure 4.6: Remote user dashboard

To set alarm and checks for automated responses from the system the following script is adjusted to motor operating conditions. where r.level indicates value of the check . eg sound alarm when bearing temperature is above maximum operating temperature.

Visually engaging dashboards can then be created from the data to give indicative overviews so as to require less skill in identifying critical incidents.

```
Check: ${ r._check_name } is: ${ r._level }
```

5 Conclusion

Continuously making new predictions and carefully examining the differences between these predictions and the actual values provides us with real-time insights into how a system behaves. This iterative forecasting approach helps us respond proactively to anomalies, making timely adjustments to keep the system stable and performing well.

For instance, if the actual temperature suddenly increases, the differences between our predictions and the actual values become more noticeable, leading to higher differences called residuals. Recognizing this difference is crucial because it indicates a significant shift in the system. Comparing this event to a previous anomaly, such as a substantial temperature drop, reveals interesting patterns. Contextualizing these anomalies is important; even if the second temperature drop has smaller residuals, it gains significance when considered with the first anomaly.

Moreover, the first anomaly, marked by a temporary increase in residuals, highlights the system's ability to adapt to new input data. The quick decrease in residuals after the initial anomaly shows that the forecasting system is responsive and can rapidly adjust to changing conditions. This responsiveness is a key feature, suggesting that prolonged increases in residuals would be unusual and might need closer investigation. In summary, analyzing residuals not only helps detect anomalies but also provides insights into the system's adaptability and responsiveness to changing conditions.

6 Recommendations

Temperature and current serve as initial indicators of deterioration in electric motors, often signaling an underlying fault in most cases. However, for precise fault pinpointing, a comprehensive set of parameters must be collected. Beyond temperature and current, additional parameters play a pivotal role in a thorough fault analysis. These include:

- **Motor Phase Change:** Monitoring changes in motor phases provides crucial insights into the system's electrical health. Discrepancies or irregularities in phase characteristics can signify potential faults or imbalances.
- **Shaft Vibrations:** Vibrations in the motor shaft offer valuable information regarding mechanical integrity. Anomalies in vibrations can point to issues such as misalignments, imbalances, or mechanical wear, contributing to a holistic understanding of the motor's condition.
- **Shaft Imbalance:** Detecting imbalances in the motor shaft is critical for maintaining smooth operation. Imbalances can result in undue stress on components, leading to accelerated wear and potential system failures if left unaddressed.
- **Shaft Misalignment:** Monitoring shaft alignment is vital as misalignments can cause significant damage over time. Misalignments may lead to increased friction, heat, and wear, impacting the overall efficiency and lifespan of the electric motor.

By incorporating these additional parameters into the monitoring and analysis process, it becomes possible to enhance the diagnostic capabilities and address potential faults proactively. The multi-parameter approach ensures a more comprehensive understanding of the motor's health, enabling timely interventions to prevent or mitigate issues before they escalate. alignment. Misalignments may lead to increased friction, heat, and wear, impacting the overall efficiency and lifespan of the electric motor.

7 Project Budget

Item List			
ITEM	SPECIFICATION	QUANTITY	PRICE
Esp32	32s, usb 2.0 Cable Type A/B	1	4500
Therm Amplifier	MAX31855K	1	790
Dev Board	NodeMCU 32S ESP32/ CH340c	1	1700
Thermocouple K type	Temp range: 0-600	1	1450
Thermocouple DH22	3.3-6v input	1	500
Power Unit	12v/5W SMPS Module	1	1700
Breadboard	MB102 165*55*50mm	2	200
Jumper Wires	20cm male 20cm female	80	500
Total			7790

8 Appendices

References

[1] M. U. Thomas and S. S. Rao, “Warranty Economic Decision Models: A Summary and Some Suggested Directions for Future Research,” *Operations Research*, vol. 47, no. 6, pp. 807–820, Dec. 1999. [Online]. Available: http://pubsonline.informs.org/doi/10.1287/opre.47.6.807

[2] J.-H. Han, D.-J. Choi, S.-K. Hong, and H.-S. Kim, *Motor Fault Diagnosis Using CNN Based Deep Learning Algorithm Considering Motor Rotating Speed.* Jinju, Korea, Apr. 2019, pages: 445.

[3] “13 common causes of motor failure.” [Online]. Available: https://www.fluke.com/en-gb/learn/blog/motors-drives-pumps-compressors/13-common-causes-of-motor-failure

[4] G. Sampaio, A. Vallim Filho, L. Silva, and L. Silva, “Prediction of Motor Failure Time Using An Artificial Neural Network,” *Sensors*, vol. 19, p. 4342, Oct. 2019.

[5] “Predictive Maintenance - an overview | ScienceDirect Topics.” [Online]. Available: https://www.sciencedirect.com/topics/engineering/predictive-maintenance

[6] S. Lu, P. Zhou, X. Wang, Y. Liu, F. Liu, and J. Zhao, “Condition monitoring and fault diagnosis of motor bearings using undersampled vibration signals from a wireless sensor network,” *Journal of Sound and Vibration*, vol. 414, pp. 81–96, Feb. 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0022460X17307691

[7] D. Kimera and N. Fillemon Nduvu, “Predictive maintenance for ballast pumps on ship repair yards via machine learning,” *Transportation Engineering*, vol. 2, p. 100020, Dec. 2020.

[8] C. Cheng, G. Ma, Y. Zhang, M. Sun, F. Teng, H. Ding, and Y. Yuan, “A deep learning-based remaining useful life prediction approach for bearings,” 12 2018. [Online]. Available: https://arxiv.org/pdf/1812.03315.pdf

[9] “Temperature Sensors: Types, How It Works, & Applications,” Jul. 2019. [Online]. Available: https://www.encardio.com/blog/temperature-sensor-probe-types-how-it-works-applications/

[10] Learn | OpenEnergyMonitor. [Online]. Available: https://learn.openenergymonitor.org/electricity-monitoring/temperature/DS18B20-temperature-sensing?redirected=true