

Desenvolvimento de aplicativo para análise de padrões de vibração de compressores

 <https://doi.org/10.47236/2594-7036.2026.v10.1887>

João Vitor Gouveia de Lima¹
Meuse Nogueira de Oliveira Júnior²

Data de submissão concluída: 19/10/2025. Data de aprovação: 29/1/2026. Data de publicação: 12/2/2026.




Resumo – Este trabalho apresenta o desenvolvimento de um aplicativo para análise de padrões de vibração em compressores utilizados em sistemas de climatização. O objetivo principal foi validar uma arquitetura de software capaz de processar, armazenar e visualizar sinais provenientes de sensores MEMS ruidosos, comuns em cenários de recursos limitados. A metodologia baseou-se no desenvolvimento de uma aplicação *desktop* em Python, integrando captura via protocolo MQTT, persistência em banco de dados PostgreSQL e processamento de sinais via algoritmos de Transformada Rápida de Fourier (FFT) e Transformada de Fourier de Curto Prazo (STFT). Para a validação da aplicação, utilizou-se um nó de aquisição baseado no ESP8266 e sensor MPU-6050 em um motor de indução. Os resultados demonstraram que o sistema processou *buffers* de 256 amostras com uma latência de transmissão e renderização de 3,4 segundos. A análise espectral identificou a frequência fundamental de vibração em torno de 120 Hz a 125 Hz, apresentando um desvio aceitável em relação ao esperado teórico (120 Hz) dada a resolução espectral imposta pela janela de amostragem. Conclui-se que o modelo desenvolvido é eficaz para triagem e monitoramento remoto, compensando as limitações de hardware através de técnicas robustas de processamento e visualização, oferecendo uma alternativa de favorável custo-benefício que atua como uma central de monitoramento para auxiliar equipes de manutenção.

Palavras-chave: Análise de vibrações. Desenvolvimento de software. IoT. Manutenção preditiva. Processamento de sinais.

Application development for vibration pattern analysis of compressors

Abstract – This work presents an application for analyzing vibration patterns in compressors used in HVAC systems. The main objective was to validate a software architecture capable of processing, storing, and visualizing signals from noisy MEMS sensors, common in resource-constrained scenarios. The methodology was based on developing a desktop application in Python, integrating capture via MQTT protocol, persistence in a PostgreSQL database, and signal processing via Fast Fourier Transform (FFT) and Short-Time Fourier Transform (STFT) algorithms. For the application validation, an acquisition node based on ESP8266 and MPU-6050 sensor was used on an induction motor. The results showed that the system processed 256-sample buffers with a transmission and rendering latency of 3.4 seconds. Spectral

¹ Estudante do curso de Tecnologia em Análise e Desenvolvimento de Sistemas do *Campus* Recife, do Instituto Federal de Pernambuco. Recife, Pernambuco, Brasil.  jvgl@discente.ifpe.edu.br  <https://orcid.org/0009-0005-2501-0621>  <http://lattes.cnpq.br/3176336326143248>.

² Doutor em Ciência da Computação pela Universidade Federal de Pernambuco. Professor das áreas de Engenharia Biomédica e Ciência da Computação do Ensino Básico, Técnico e Tecnológico no *Campus* Recife, do Instituto Federal de Pernambuco. Recife, Pernambuco, Brasil.  meusejunior@recife.ifpe.edu.br  <https://orcid.org/0000-0001-7670-0765>  <http://lattes.cnpq.br/8250068675147894>.

analysis identified the fundamental vibration frequency around 120 Hz to 125 Hz, presenting an acceptable deviation from the theoretical expectation (120 Hz) given the spectral resolution imposed by the sampling window. It is concluded that the developed model is effective for screening and remote monitoring, compensating for hardware limitations through robust processing and visualization techniques, offering a cost-effective alternative that acts as a monitoring hub to assist maintenance teams.

Keywords: IoT. Predictive maintenance. Signal processing. Software development. Vibration analysis.

Desarrollo de una aplicación para el análisis de patrones de vibración en compresores

Resumen – Este trabajo presenta el desarrollo de una aplicación para el análisis de patrones de vibración en compresores utilizados en sistemas de climatización. El objetivo principal fue validar una arquitectura de software capaz de procesar, almacenar y visualizar señales provenientes de sensores MEMS ruidosos, comunes en escenarios con recursos limitados. La metodología se basó en el desarrollo de una aplicación de escritorio en Python, integrando la captura mediante protocolo MQTT, persistencia en una base de datos PostgreSQL y procesamiento de señales vía algoritmos de Transformada Rápida de Fourier (FFT) y Transformada de Fourier de Corto Tiempo (STFT). Para la validación de la aplicación, se utilizó un nodo de adquisición basado en ESP8266 y sensor MPU-6050 en un motor de inducción. Los resultados demostraron que el sistema procesó búferes de 256 muestras con una latencia de transmisión y renderizado de 3,4 segundos. El análisis espectral identificó la frecuencia fundamental de vibración en torno a 120 Hz a 125 Hz, presentando una desviación aceptable respecto al esperado teórico (120 Hz) dada la resolución espectral impuesta por la ventana de muestreo. Se concluye que el modelo desarrollado es eficaz para el triaje y monitoreo remoto, compensando las limitaciones del hardware a través de técnicas robustas de procesamiento y visualización, ofreciendo una alternativa con una favorable relación costo-beneficio que actúa como una central de monitoreo para auxiliar a los equipos de mantenimiento.

Palabras clave: Análisis de vibraciones. Desarrollo de software. IoT. Mantenimiento predictivo. Procesamiento de señales.

Introdução

Nas indústrias tradicionais, as práticas de manutenção são frequentemente classificadas como corretivas ou preventivas. Conforme aponta Baldissarelli e Fabro (2018), a manutenção corretiva é aplicada somente após a falha de um equipamento, o que pode levar a paradas não programadas, altos custos operacionais e prejuízos na produção. Já a manutenção preventiva busca evitar essas falhas por meio de inspeções e substituições programadas, muitas vezes baseadas no tempo de uso ou na experiência técnica. Embora essas abordagens sejam amplamente utilizadas, elas nem sempre conseguem antecipar falhas de maneira precisa, o que limita sua eficácia em ambientes industriais modernos.

Com o advento da Indústria 4.0, surgiram novas oportunidades para superar as limitações das práticas tradicionais de manutenção. A integração de tecnologias como Internet das Coisas (*Internet of Things* - IoT), *Big Data* e inteligência artificial possibilitou a implementação de sistemas de manutenção mais inteligentes e conectados. A Indústria 4.0 visa transformar os processos industriais por meio de

digitalização e conectividade, promovendo maior eficiência e tomada de decisões baseadas em dados (Pereira; Simonetto, 2018; Lee *et al.*, 2015). Nesse contexto, a manutenção preditiva ganha destaque ao utilizar sensores inteligentes e análise em tempo real para prever falhas antes que ocorram, garantindo maior confiabilidade e disponibilidade dos equipamentos.

A manutenção preditiva é uma abordagem essencial para garantir a confiabilidade e a eficiência dos equipamentos industriais, sendo amplamente aplicada para monitorar e diagnosticar falhas em sistemas mecânicos. De acordo com Baldissarelli e Fabro (2019, p. 14):

a manutenção preditiva também é conhecida como manutenção sob condição ou manutenção com base no estado do equipamento. Ela é baseada na tentativa de definir o estado futuro de um equipamento ou sistema, por meio dos dados coletados ao longo do tempo por uma instrumentação específica, verificando e analisando a tendência de variáveis do equipamento.

A análise de vibrações se destaca como uma das técnicas cruciais da manutenção preditiva no monitoramento de máquinas rotativas, como os compressores de sistemas de climatização. Isso ocorre porque os compressores emitem padrões de vibrações característicos quando estão em bom estado de funcionamento. Como destaca Cyrino (2017), ao longo do tempo, essas vibrações resultam em desgaste e deterioração, o que modifica a distribuição da energia vibratória entre os componentes que constituem a máquina ou o equipamento. Consequentemente, o aparelho foge de seus padrões, o que é caracterizado pelo aumento na intensidade de suas vibrações.

Diante do contexto apresentado, este trabalho apresenta o desenvolvimento de uma aplicação *desktop* para análise de padrões de vibração de compressores, utilizando dados provenientes de sensores de baixo custo. O projeto foi concebido no âmbito do GPSERS (Grupo de Pesquisa em Sistemas Embutidos e Redes de Sensores), localizado no Instituto Federal de Pernambuco - Campus Recife. A proposta é motivada pela busca de preencher a lacuna entre a necessidade de soluções acessíveis e os altos custos das ferramentas comerciais, demonstrando que técnicas eficazes de processamento digital podem extrair diagnósticos confiáveis mesmo a partir de hardware acessível. O objetivo principal é validar uma ferramenta prática e escalável, capaz de identificar precocemente falhas, otimizando a manutenção de sistemas de climatização.

A abordagem proposta fundamenta-se em uma análise multi-perspectiva, empregando um conjunto de técnicas de processamento de sinais que abrangem os domínios do tempo, da frequência e do tempo-frequência. Estes métodos foram implementados e validados no aplicativo proposto.

O sinal de vibração bruto capturado por um acelerômetro é uma série temporal complexa. Para extrair informações diagnósticas, é necessário aplicar técnicas de processamento de sinais, que transformam os dados em representações mais informativas.

A análise no domínio do tempo é a representação mais direta do sinal. A partir desta forma de onda, é possível extrair métricas estatísticas que fornecem uma visão geral da condição da máquina, como valor de pico, amplitude do sinal e Fator de Crista. Embora úteis, esses indicadores possuem limitações no diagnóstico da causa específica da falha e servem como um primeiro nível de alerta (Jardine; Lin; Banjevic, 2006).

A análise no domínio da frequência é empregada para identificar as causas subjacentes de uma vibração excessiva. Para isso, é essencial decompor o sinal em suas frequências constituintes. A ferramenta matemática para essa tarefa é a Transformada de Fourier. Na prática, utiliza-se o algoritmo da Transformada Rápida de Fourier (FFT), que é computacionalmente eficiente para calcular o espectro de frequência de um sinal digital (Smith, 1997; Oppenheim; Schaffer, 2010). O espectro resultante mostra a amplitude da vibração em cada frequência, permitindo associar picos de energia a fenômenos físicos específicos, como as falhas de desbalanceamento e desalinhamento.

Muitas técnicas de análise partem do pressuposto de que o sinal de vibração é estacionário, o que nem sempre é verdade, especialmente quando defeitos começam a surgir. Para sinais não-estacionários, as análises no domínio tempo-frequência são mais adequadas. A Transformada de Fourier de Curto Prazo (STFT) é uma das principais técnicas dessa categoria. Ela funciona dividindo o sinal em segmentos menores, que podem ser considerados localmente estacionários, e aplicando a FFT a cada um deles. O resultado, geralmente visualizado como um espectrograma, revela como o conteúdo de frequência do sinal evolui com o tempo, sendo crucial para capturar eventos transitórios e entender a dinâmica de falhas em desenvolvimento (Safizadeh; Lakis; Thomas, 2005).

A combinação dessas três abordagens de análise oferece uma visão completa do comportamento vibratório do compressor, permitindo um diagnóstico informado sobre a natureza e evolução de possíveis falhas.

Materiais e métodos

O foco metodológico deste trabalho é o desenvolvimento de software e o processamento de sinais. A validação experimental foi realizada utilizando um *setup* de hardware simplificado apenas como fonte de dados para a aplicação. O desenvolvimento ocorreu entre setembro de 2024 e agosto de 2025.

O desenvolvimento intelectual, a concepção da pesquisa, a codificação do software, a coleta de dados experimentais e a análise dos resultados foram realizados integralmente pelos autores. Ferramentas de Inteligência Artificial baseadas em Grandes Modelos de Linguagem (LLMs) foram utilizadas estritamente como auxílio para a revisão gramatical, aprimoramento da fluidez textual, tradução dos resumos (*Abstract* e *Resumen*) e formatação das referências bibliográficas conforme as normas da ABNT. Todo o conteúdo gerado ou revisado pela ferramenta foi validado criticamente pelos autores para garantir a precisão e a originalidade da obra.

Prospecção de tecnologia

O aplicativo foi desenvolvido na linguagem Python, escolhida por sua vasta gama de bibliotecas para análise de dados. Abaixo estão representados os recursos de desenvolvimento:

- Visual Studio Code (VISUAL STUDIO CODE, 2025): IDE utilizada para a implementação das funcionalidades e implantação das bibliotecas;
- NumPy (HARRIS et al., 2020): Biblioteca utilizada para a manipulação dos dados brutos de vibração e implementação do algoritmo da Transformada Rápida de Fourier (FFT), permitindo a conversão do sinal do domínio do tempo para a frequência;
- Librosa (MCFEE et al., 2015): Biblioteca utilizada para calcular a transformada de Fourier de curto prazo (STFT) e gerar as matrizes de dados necessárias para a construção dos espectrogramas;

- Matplotlib (HUNTER, 2007): Biblioteca de visualização de dados utilizada para renderizar graficamente os sinais processados (formas de onda e espectros);
- PostgreSQL (POSTGRES GLOBAL DEVELOPMENT GROUP, 2025): Sistema gerenciador de banco de dados relacional (SGDB) utilizado para estruturar e armazenar os metadados das análises e os caminhos dos arquivos gerados;
- ThingsBoard (THINGSBOARD, 2025): Plataforma *open-source* de IoT utilizada para gerenciar dispositivos conectados e receber dados de telemetria, permitindo a criação de *dashboards* para monitoramento remoto em tempo real;
- Pdoc (PDO, 2025): Biblioteca utilizada para automatizar a geração da documentação técnica do sistema em formato HTML, baseada nas *docstrings* do código.

Etapas realizadas

O desenvolvimento do aplicativo progrediu utilizando as metodologias ágeis Scrum (SCHWABER; SUTHERLAND, 2020) e Kanban (KANBAN UNIVERSITY, 2021), organizando as entregas em *sprints* mensais com revisões periódicas. As etapas metodológicas executadas foram:

1. **Estruturação básica do aplicativo:** Foi desenvolvida a arquitetura modular do software e integradas as bibliotecas de cálculo numérico (NumPy). Nesta etapa, validou-se o método de processamento gerando ondas senoidais sintéticas e aplicando o algoritmo da Transformada Rápida de Fourier (FFT) para decompor o sinal em seus componentes de frequência, plotando os resultados tanto no domínio do tempo quanto da frequência para verificar a precisão matemática da conversão.
2. **Implementação de registro e persistência dos dados:** Foi modelado e implementado o esquema do banco de dados relacional (PostgreSQL). O método de acesso aos dados foi desenvolvido utilizando o padrão DAO (*Data Access Object*), garantindo a abstração das operações de leitura e escrita dos metadados das análises vibratórias.
3. **Implementação de análise Tempo-Frequência (Espectrograma):** Para analisar sinais não estacionários, implementou-se o método da Transformada de Fourier de Curto Prazo (STFT) via biblioteca Librosa. Este método segmenta o sinal em janelas temporais para calcular a evolução das frequências ao longo do tempo, gerando gráficos de intensidade (mapas de calor) que facilitam o diagnóstico visual de falhas intermitentes.
4. **Implementação de Servidor Socket TCP:** Foi desenvolvido um serviço de escuta em rede utilizando Sockets TCP. Este método foi escolhido para garantir a integridade e a ordem de recebimento do fluxo contínuo (*stream*) de dados brutos no formato JSON enviados pelo sensor, minimizando a latência na comunicação local.
5. **Implementação de estratégia híbrida de armazenamento de imagens:** Desenvolveu-se uma rotina para persistência dos resultados gráficos. O método adotado foi híbrido: os arquivos de imagem (gráficos gerados) são salvos fisicamente em diretórios organizados no servidor, enquanto seus caminhos de referência são indexados no banco de dados, otimizando o desempenho do armazenamento.
6. **Implementação de gestão de arquivos:** Foram criados algoritmos para a busca e exclusão segura de registros. O método garante a integridade referencial, assegurando que, ao deletar um registro do banco de dados, o arquivo físico correspondente no servidor também seja removido, e vice-versa.

7. *Implementação da interface do usuário*: Foi construída a *interface* gráfica (GUI) do sistema, organizada em abas para separar as responsabilidades (visualização em tempo real, lista de imagens armazenadas e configurações). A aba de configurações permite parametrizar dinamicamente o endereçamento IP e as credenciais do Broker MQTT — um componente centralizador de mensagens explicado na etapa seguinte — além de definir os caminhos de armazenamento, sem a necessidade de alterar o código-fonte.
8. *Implementação de Broker mosquitto MQTT*: Para viabilizar a arquitetura IoT, foi implementado um contêiner Docker rodando o serviço Mosquitto. Este componente atua como um Broker MQTT, que é um intermediário responsável por receber as mensagens publicadas pelos sensores (tópicos de vibração) e distribuí-las para os clientes inscritos (*subscribers*), desacoplando a fonte dos dados da aplicação de análise.
9. *Integração com a plataforma ThingsBoard*: Foi implementado o módulo de telemetria utilizando a API HTTP da plataforma. O método consiste em formatar os dados processados em estruturas JSON e enviá-los através de requisições HTTP do tipo POST. Essa abordagem permite atualizar os *dashboards* na nuvem de forma assíncrona, utilizando a *interface* de telemetria padrão do ThingsBoard.
10. *Implementação da Documentação*: Foi gerada a documentação técnica automatizada do sistema utilizando a biblioteca Pdoc, garantindo a rastreabilidade e manutenibilidade do código desenvolvido.
11. *Validação experimental do software*: Realização de testes funcionais utilizando dados reais provenientes de uma bancada com motor de indução monofásico e acelerômetro MPU-6050. Esta etapa teve como objetivo validar a capacidade do software em processar sinais ruidosos, calcular a FFT corretamente e apresentar os resultados com a latência esperada.

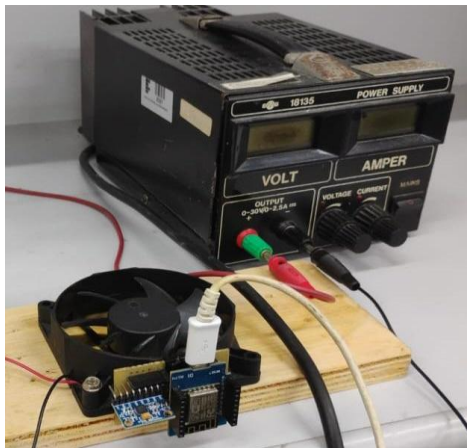
Resultados e discussões

Apesar da relevância da manutenção preditiva, sua implementação em larga escala é frequentemente inviabilizada pelos custos elevados de ferramentas comerciais e sensores de alta precisão, que podem ser superdimensionados para ativos menos críticos, como compressores de climatização. Tentativas de mitigar essa barreira, como o protótipo desenvolvido por Vavassori et al. (2022) ao custo de R\$ 295,10, demonstram que, embora a redução do investimento em hardware seja possível, ela introduz desafios técnicos, como divergências de amplitude e limitações na taxa de amostragem. Esse cenário evidencia que a viabilização de soluções acessíveis não depende apenas de componentes econômicos, mas exige estratégias robustas de compensação via software para garantir a confiabilidade dos diagnósticos.

Neste contexto, a solução aqui apresentada busca comprovar que uma arquitetura de software bem otimizada é capaz de extrair diagnósticos úteis mesmo a partir de fontes de dados limitadas. Para "estressar" os algoritmos desenvolvidos e validar essa hipótese, utilizou-se como fonte de sinal o acelerômetro MPU-6050 (custo médio de R\$ 30,00). A escolha deste componente não foi apenas econômica, mas estratégica: se o software for capaz de mitigar o ruído e extrair frequências de falha de um sensor MEMS de baixo custo, sua eficácia estaria comprovada para qualquer cenário de hardware superior.

Para validar a proposta, inicialmente, foi criado um emulador de vibrações (Figura 1) com um acelerômetro que envia os dados via protocolo TCP e MQTT para o aplicativo, que processa as informações e exibe gráficos na tela.

Figura 1 – Emulador de Vibrações



Fonte: Própria (2025)

O emulador representado é composto pelo sensor acelerômetro MPU-6050 que capta as vibrações geradas por um *cooler*. O acelerômetro foi escolhido por sua capacidade de detectar movimentos e vibrações de maneira eficiente, além de ser acessível em termos de custo.

O objetivo do emulador é replicar o comportamento vibracional típico de um compressor em funcionamento. O acelerômetro mede a aceleração em três eixos (X, Y e Z), proporcionando uma visão tridimensional das vibrações geradas pelo compressor durante seu funcionamento. Esses dados são transmitidos em tempo hábil para o aplicativo de análise via protocolo TCP/IP ou MQTT.

Com o ciclo de desenvolvimento do aplicativo finalizado e todas as suas funcionalidades, desde a análise de sinais até a *interface* do usuário e o armazenamento de dados, implementadas e testadas em ambiente simulado, a etapa subsequente foi a validação final do sistema. O objetivo desta fase foi comprovar a eficácia e a robustez da solução completa ao ser submetida a dados de vibração provenientes de um equipamento físico real.

Para este propósito, foi utilizada uma bancada de testes com um motor elétrico industrial com o sensor MPU-6050 acoplado em sua carcaça, conforme ilustrado na Figura 2. O equipamento é um motor de indução monofásico do tipo TFVE (Totalmente Fechado com Ventilação Externa), escolhido por ser representativo dos compressores e ventiladores encontrados em sistemas de climatização, alvo principal desta pesquisa.

A análise espectral dos dados coletados pelo aplicativo, por meio da Transformada Rápida de Fourier (FFT), revelou um pico de frequência dominante e claramente definido. Conforme mostram os gráficos da Figura 3, este pico foi identificado em aproximadamente 123 Hz. Este resultado é plenamente condizente com a assinatura de vibração esperada para um motor de indução monofásico operando em uma rede elétrica de 60 Hz. O componente em 120 Hz, correspondente a duas vezes a frequência da linha (2x FL), é o mais característico e resulta da pulsação do torque eletromagnético intrínseco a esse tipo de motor (FINLEY et al., 1999). A pequena diferença entre o valor teórico (120 Hz) e o medido (123 Hz) é atribuída à resolução da análise espectral, uma consequência do uso de uma janela

de 256 amostras para garantir o processamento em tempo hábil dos dados provenientes do sensor de baixo custo.

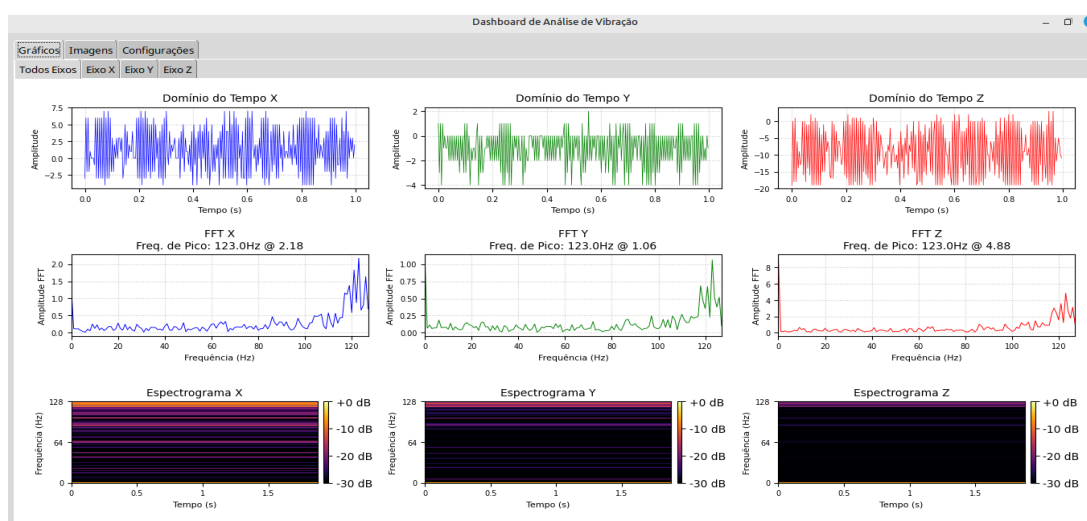
Figura 2 – Motor Monofásico



Fonte: Própria (2025)

A aba principal da *interface* do aplicativo, conforme exibida na Figura 3, permite a visualização dos gráficos no domínio do tempo, FFT e STFT, facilitando a identificação de possíveis anormalidades no comportamento do equipamento.

Figura 3 – Gráficos de análise (aba todos os eixos)



Fonte: Própria (2025)

Os gráficos na primeira linha da *interface* referem-se ao domínio do tempo, funcionando de maneira análoga a um sismógrafo ou eletrocardiograma do motor. Eles representam a amplitude bruta da vibração em função do tempo, ou seja, a intensidade do movimento momento a momento. A leitura é intuitiva, pois picos elevados e repentinos saltam à vista e indicam eventos imediatos, como impactos mecânicos, batidas ou instabilidades súbitas no funcionamento do compressor, permitindo ao operador identificar rapidamente "o quanto" o equipamento está vibrando.

Já gráficos na segunda linha da *interface* resultam da aplicação da Transformada Rápida de Fourier (FFT), que decompõe o sinal para revelar a

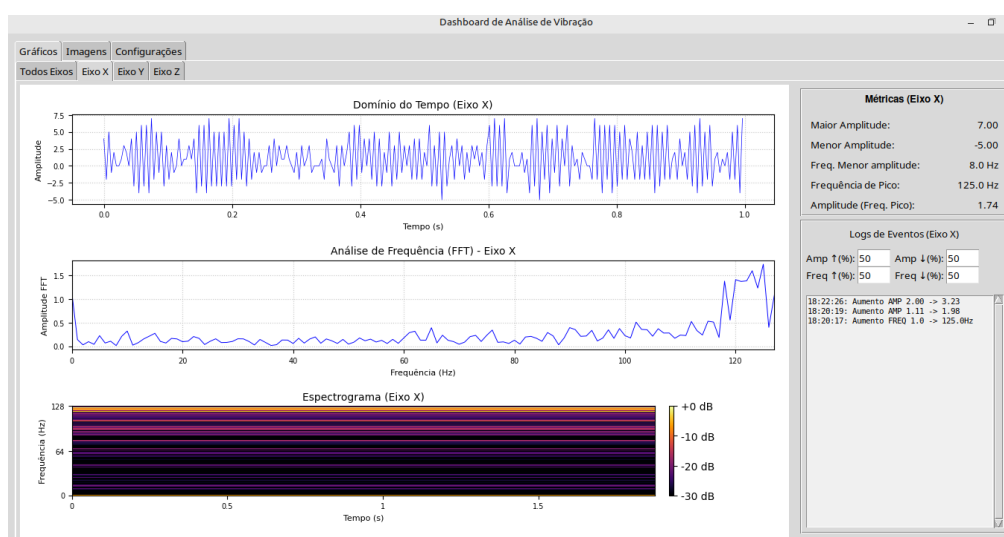
"assinatura" da vibração. Diferente da visão anterior, aqui o foco não é o momento, mas a frequência (em Hz), permitindo distinguir a origem do problema. Cada pico neste gráfico funciona como um identificador da falha, possibilitando ao operador discernir se a vibração provém, por exemplo, de um desbalanceamento mecânico (frequências baixas) ou de uma indução elétrica (frequências altas), agindo como um diagnóstico de identidade do defeito.

Por fim, os gráficos na terceira linha da *interface* representam a Transformada de Fourier de Curto Prazo (STFT), atuando como um mapa de calor histórico que combina as duas visões anteriores. A escala de cores facilita a interpretação visual por não especialistas: cores mais escuras indicam operação normal e de baixa energia, enquanto cores mais claras ou "quentes" alertam para zonas de alta intensidade vibratória. Essa visualização é essencial para monitorar a evolução do comportamento da máquina, revelando se uma frequência de falha é constante ou se aparece de forma intermitente ao longo da operação.

Um dos resultados mais significativos da plataforma é a sua capacidade de traduzir a análise visual para o domínio quantitativo, por meio da extração automática de métricas-chave. Conforme demonstrado na Figura 4, para cada eixo de análise, o sistema calcula e exibe em tempo real um painel com os seguintes indicadores diagnósticos:

- ***Maior e Menor Amplitude:*** Indicam os valores extremos da vibração no domínio do tempo, úteis para detectar impactos ou eventos transitórios.
- ***Frequência de Pico:*** Identifica a frequência dominante no espectro, sendo a métrica mais crítica para associar a vibração a falhas mecânicas específicas.
- ***Amplitude da Frequência de Pico:*** Quantifica a severidade da vibração na frequência dominante, permitindo acompanhar a evolução de um defeito.
- ***Frequência de Menor Amplitude:*** Fornece uma referência do nível de vibração de base ou ruído do sistema.
- ***Logs de Aumentos e Quedas:*** Registra eventos de aumento ou quedas bruscas de amplitude e frequência de pico.

Figura 4 – Métricas na *interface* (aba eixos individuais)



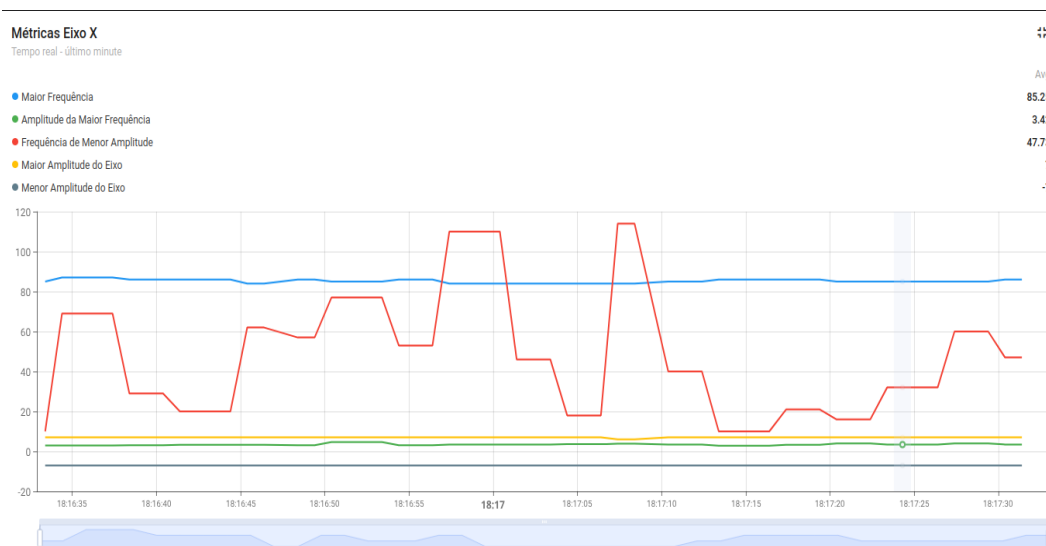
Fonte: Própria (2025)

Visando a escalabilidade e a conformidade com os princípios da Indústria 4.0, a plataforma foi integrada a um sistema externo de Internet das Coisas (IoT). Para

viabilizar essa comunicação, o dispositivo sensor é previamente registrado na plataforma ThingsBoard, que disponibiliza um endereço de API (*endpoint*) exclusivo para a ingestão de dados. As métricas quantitativas extraídas durante a análise são, então, enviadas para esta URL através de requisições HTTP do tipo POST. Uma vez recebidos, os dados são persistidos no banco de dados em nuvem da plataforma, servindo de base para a alimentação e atualização dinâmica dos *dashboards* de monitoramento em tempo real.

Essa integração expande as fronteiras do monitoramento para além da estação de análise local. Ela permite que gestores e equipes de manutenção supervisionem a condição do equipamento remotamente, configurem alertas automáticos por e-mail ou SMS caso os níveis de vibração excedam limiares de segurança e centralizem dados de múltiplos ativos em um único local. A Figura 5 apresenta o recebimento e a plotagem desses dados telemétricos na plataforma ThingsBoard, validando o fluxo de comunicação de ponta a ponta.

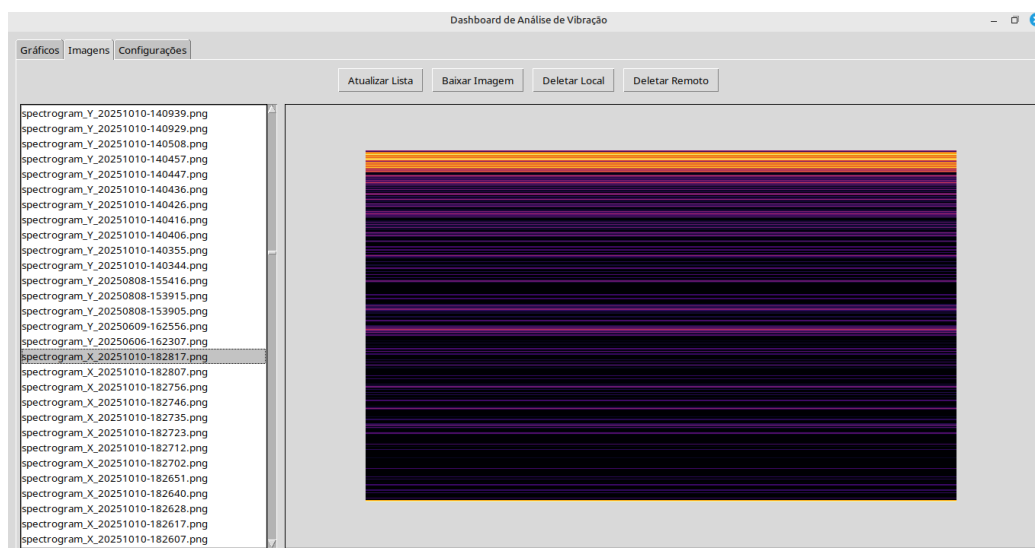
Figura 5 – *Dashboard* do ThingsBoard com a telemetria



Fonte: Própria (2025)

O aplicativo também oferece uma funcionalidade para exibir as imagens armazenadas no servidor. A figura 6 ilustra a *interface* do aplicativo na aba imagens, que apresenta os arquivos organizados em uma lista cronológica. A classificação e a identificação de cada registro são realizadas diretamente através da nomenclatura padronizada dos arquivos, permitindo ao usuário distinguir visualmente o tipo de análise (FFT ou STFT), o eixo monitorado e o carimbo de tempo (*timestamp*) da ocorrência. Embora a versão atual priorize a constituição de um banco de dados sequencial para o futuro treinamento de redes neurais — dispensando, neste estágio, filtros de busca avançados — a estrutura lógica dos nomes e a ordenação temporal garantem o acesso organizado e a rastreabilidade necessária para a gestão manual dos arquivos.

Figura 6 – Lista de Imagens



Fonte: Própria (2025)

Para a persistência dos resultados visuais, adotou-se uma estratégia de armazenamento híbrida. Embora o SGBD PostgreSQL suporte nativamente a gravação de dados binários (utilizando tipos como BYTEA ou Large Objects), nesta arquitetura optou-se por armazenar os arquivos físicos em diretórios organizados no sistema de arquivos do servidor (img_fft e img_espectrograma). Dessa forma, o banco de dados não é sobrecarregado com *blobs* de imagem, atuando estritamente como um índice de metadados que registra o caminho (*path*) referencial para cada arquivo.

Um componente fundamental deste formato é a implementação de uma convenção de nomenclatura padronizada e autodescritiva para cada arquivo gerado, seguindo o formato: tipoDaAnálise_eixo_data-hora.png (ex: fft_X_20250411-130233.png). Este padrão garante não apenas a unicidade de cada registro, evitando sobreposições, mas também oferece rastreabilidade completa, permitindo que cada imagem seja facilmente associada a um tipo de análise, a um eixo específico e ao momento exato em que foi gerada, apenas pela leitura de seu nome.

A abordagem é ilustrada na Figura 7, que apresenta os registros no banco de dados e os arquivos no servidor.

Figura 7 – Registro das imagens no servidor e no banco de dados

```
dbserve@dbserver:~/img_fft$ ls
fft_X_20250411-130233.png  fft_Y_20250411-130233.png  fft_Z_20250411-130233.png
fft_X_20250414-111237.png  fft_Y_20250414-111237.png  fft_Z_20250414-111237.png
fft_X_20250414-111247.png  fft_Y_20250414-111247.png  fft_Z_20250414-111247.png
fft_X_20250414-111258.png  fft_Y_20250414-111258.png  fft_Z_20250414-111258.png
```

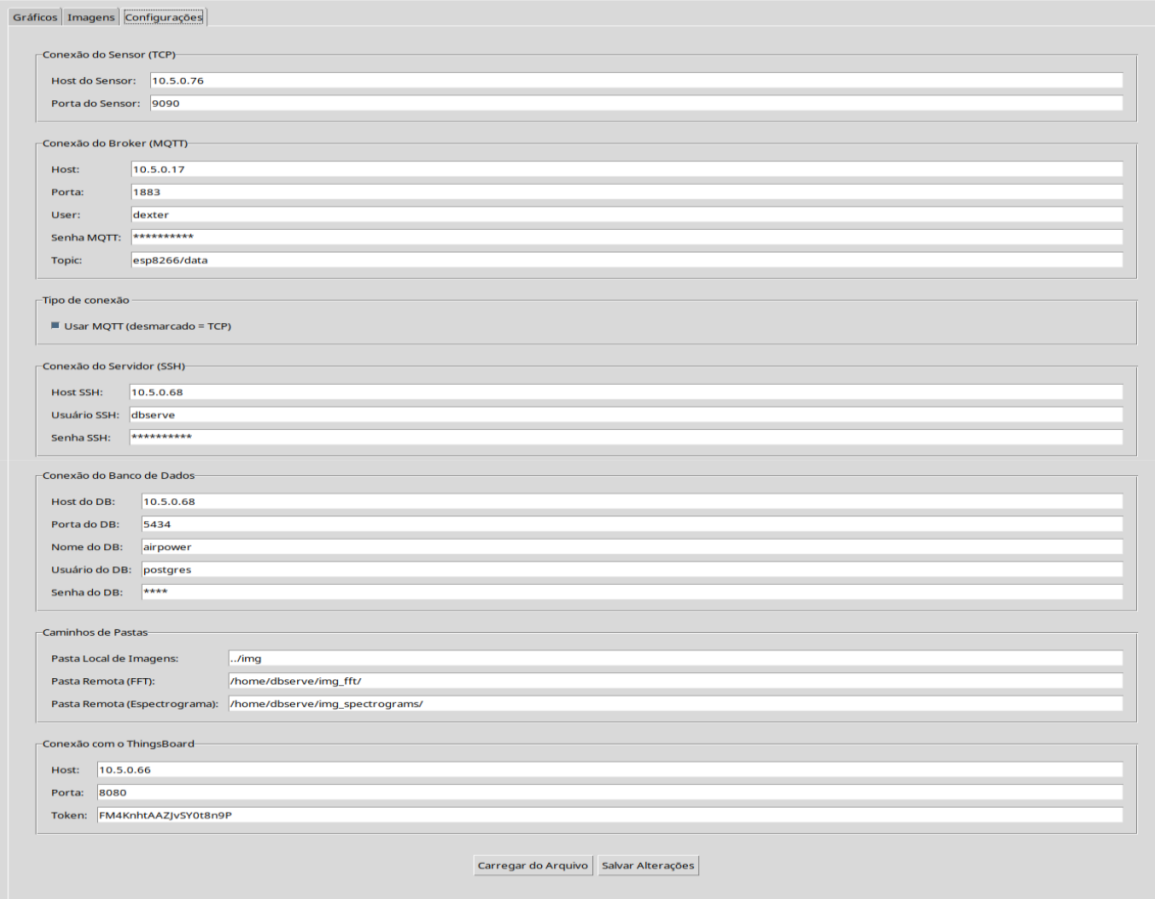
	id	Az nome	Az caminho	data criacao
1	17	fft_X_20250411-130233.png	/home/dbserve/img_fft/fft_X_2025-04-11 13:02:35.982 -0300	
2	18	fft_Y_20250411-130233.png	/home/dbserve/img_fft/fft_Y_2025-04-11 13:02:36.030 -0300	
3	19	fft_Z_20250411-130233.png	/home/dbserve/img_fft/fft_Z_2025-04-11 13:02:36.043 -0300	
4	20	spectrogram_X_20250411-13	/home/dbserve/img_spectrogr25-04-11 13:02:37.143 -0300	
5	21	spectrogram_Y_20250411-13	/home/dbserve/img_spectrogr25-04-11 13:02:37.146 -0300	
6	22	spectrogram_Z_20250411-13	/home/dbserve/img_spectrogr25-04-11 13:02:37.151 -0300	
7	23	fft_X_20250414-111237.png	/home/dbserve/img_fft/fft_X_2025-04-14 11:12:39.782 -0300	
8	24	fft_Y_20250414-111237.png	/home/dbserve/img_fft/fft_Y_2025-04-14 11:12:39.825 -0300	
9	25	fft_Z_20250414-111237.png	/home/dbserve/img_fft/fft_Z_2025-04-14 11:12:39.839 -0300	
10	26	spectrogram_X_20250414-11	/home/dbserve/img_spectrogr25-04-14 11:12:40.958 -0300	
11	27	spectrogram_Y_20250414-11	/home/dbserve/img_spectrogr25-04-14 11:12:40.962 -0300	
12	28	spectrogram_Z_20250414-11	/home/dbserve/img_spectrogr25-04-14 11:12:40.975 -0300	
13	29	fft_X_20250414-111247.png	/home/dbserve/img_fft/fft_X_2025-04-14 11:12:50.227 -0300	
14	30	fft_Y_20250414-111247.png	/home/dbserve/img_fft/fft_Y_2025-04-14 11:12:50.231 -0300	
15	31	fft_Z_20250414-111247.png	/home/dbserve/img_fft/fft_Z_2025-04-14 11:12:50.249 -0300	

Fonte: Própria (2025)

Para garantir a máxima flexibilidade, adaptabilidade e a fácil implantação da plataforma em diferentes cenários de hardware e rede, foi implementado um painel de configuração com campos para especificar os atributos de conexão. Este módulo desacopla os parâmetros operacionais do código-fonte, permitindo que o usuário final modifique todas as variáveis críticas da aplicação sem a necessidade de intervenção técnica no software.

Conforme apresentado na Figura 8, a *interface* agrupa de forma lógica todas as configurações essenciais, incluindo os detalhes de conexão com o sensor (via TCP ou MQTT), as credenciais de acesso ao servidor remoto (SSH) e ao banco de dados, os endereços da plataforma de IoT (ThingsBoard) e os caminhos para o armazenamento local e remoto dos arquivos de imagem. A capacidade de salvar e carregar essas configurações a partir de um arquivo externo assegura a consistência e a reprodutibilidade das análises, além de facilitar a migração do sistema entre diferentes ambientes de teste ou produção. Esta abordagem modular reforça a robustez e a usabilidade da ferramenta, tornando-a prontamente adaptável a novas infraestruturas.

Figura 8 – Aba de configurações do aplicativo



The screenshot displays the 'Configurações' (Settings) tab of an application. It features several sections for configuring different components:

- Conexão do Sensor (TCP):** Fields for 'Host do Sensor' (10.5.0.76) and 'Porta do Sensor' (9090).
- Conexão do Broker (MQTT):** Fields for 'Host' (10.5.0.17), 'Porta' (1883), 'User' (dexter), 'Senha MQTT' (masked), and 'Topic' (esp8266/data).
- Tipo de conexão:** A checkbox labeled 'Usar MQTT (desmarcado = TCP)' is checked.
- Conexão do Servidor (SSH):** Fields for 'Host SSH' (10.5.0.68), 'Usuário SSH' (dbserve), and 'Senha SSH' (masked).
- Conexão do Banco de Dados:** Fields for 'Host do DB' (10.5.0.68), 'Porta do DB' (5434), 'Nome do DB' (airpower), 'Usuário do DB' (postgres), and 'Senha do DB' (masked).
- Caminhos de Pastas:** Fields for 'Pasta Local de Imagens' (./img), 'Pasta Remota (FFT)' (/home/dbserve/img_fft/), and 'Pasta Remota (Espectrograma)' (/home/dbserve/img_spectrograms/).
- Conexão com o ThingsBoard:** Fields for 'Host' (10.5.0.66), 'Porta' (8080), and 'Token' (FM4KnhtAAZjvSY0t8n9P).

At the bottom right, there are two buttons: 'Carregar do Arquivo' and 'Salvar Alterações'.

Fonte: Própria (2025)

Por fim, visando a reprodutibilidade e a manutenibilidade do projeto, a aplicação foi inteiramente documentada. Utilizou-se a biblioteca Pdoc para gerar uma referência completa dos *scripts* em formato HTML, a partir dos comentários inseridos no código-fonte.

A validação econômica da proposta também foi confirmada. O custo total dos materiais para a montagem do nó de aquisição utilizado nos testes (Módulo ESP8266:

~R\$ 35,00; Sensor MPU-6050: ~R\$ 30,00; Cabo USB para alimentação: ~R\$ 15,00) totalizou aproximadamente R\$ 80,00. Este valor, significativamente inferior às soluções comerciais, ratifica a premissa de que a inteligência do sistema foi deslocada do hardware para a camada de software.

Considerações finais

Este trabalho se propôs a desenvolver uma arquitetura de software capaz de viabilizar a manutenção preditiva em cenários de recursos limitados. A principal contribuição da pesquisa foi demonstrar que o obstáculo financeiro, frequentemente associado a equipamentos de análise de vibração, pode ser superado por uma camada de aplicação robusta. Os resultados indicaram que técnicas de processamento digital de sinais (FFT e STFT) bem implementadas conseguem compensar as limitações de sensibilidade de sensores *low-end*, extraindo diagnósticos confiáveis de equipamentos como motores de indução.

A aplicação desenvolvida gerenciou com êxito todo o ciclo de vida dos dados: desde a aquisição via protocolos TCP e MQTT, passando pela extração automática de métricas quantitativas, até a persistência híbrida e visualização em *dashboards*. A integração com a plataforma de IoT validou a capacidade do sistema de atuar como uma central de monitoramento conectada, alinhada aos princípios da Indústria 4.0.

Entretanto, reconhece-se que o presente desenvolvimento representa um estágio inicial de validação tecnológica. A etapa experimental, embora suficiente para comprovar a lógica dos algoritmos e a estabilidade da comunicação, restringiu-se a ambientes controlados e emulados. Uma limitação operacional da versão atual é o suporte ao monitoramento de apenas um ponto de aquisição (um único sensor) por instância de execução.

Diante disso, como trabalhos futuros para a evolução desta ferramenta, sugere-se inicialmente o escalonamento da arquitetura do software através da refatoração do *backend*. O objetivo é suportar a orquestração e o processamento simultâneo de múltiplos sensores via técnicas de *multi-threading*, habilitando o monitoramento de parques de máquinas complexos. Em uma etapa subsequente, recomenda-se a validação em campo através da instalação do sistema em condicionadores de ar operando em regime real. Essa exposição a diversas condições de carga e temperatura é fundamental para testar a robustez do software em ambientes industriais hostis e para calibrar os limiares de um futuro sistema de alertas. Este módulo de notificação automática enviaria avisos (via e-mail ou SMS) sempre que os níveis de amplitude ou a frequência de pico ultrapassassem os limites de segurança pré-estabelecidos. Por fim, vislumbra-se a aplicação de Inteligência Artificial e Visão Computacional sobre o banco de espectrogramas gerado, treinando redes neurais para analisar visualmente as imagens e classificar automaticamente tipos de falhas, como desbalanceamentos ou defeitos elétricos, eliminando a necessidade de intervenção humana na triagem.

Conclui-se que o software apresentado oferece uma base sólida e modular para a democratização da manutenção preditiva, transferindo a inteligência da análise do hardware caro para o processamento lógico acessível.

Referências

BALDISSARELLI, L. N.; FABRO, E. Manutenção preditiva na indústria 4.0. **Scientia cum Industria**, [S. l.], v. 7, n. 2, p. 12-22, 2019. Disponível em: <https://sou.ucs.br/etc/revistas/index.php/scientiacumindustria/article/view/6835>. Acesso em: 9 fev. 2026.

CYRINO, L. **Diagnóstico de falhas das vibrações**. 2017. Disponível em: <https://www.manutencaoemfoco.com.br/diagnostico-de-falhas-das-vibracoes/>. Acesso em: 8 jan. 2025.

FINLEY, W. R.; HODOWANEC, M. M.; HOLTER, W. G. An analytical approach to solving motor vibration problems. In: IEEE INDUSTRY APPLICATIONS SOCIETY ANNUAL MEETING, 46., 1999, Phoenix. **Conference Record** [...]. Phoenix: IEEE, 1999. p. 217–232.

HARRIS, C. R. et al. Array programming with NumPy. **Nature**, London, v. 585, n. 7825, p. 357–362, 2020.

HUNTER, J. D. Matplotlib: A 2D graphics environment. **Computing in Science & Engineering**, New York, v. 9, n. 3, p. 90–95, 2007.

JARDINE, A. K. S.; LIN, D.; BANJEVIC, D. A review on machinery diagnostics and prognostics implementing condition-based maintenance. **Mechanical Systems and Signal Processing**, Amsterdam, v. 20, n. 7, p. 1483-1510, 2006.

KANBAN UNIVERSITY. **The Kanban Guide**. 2021. Disponível em: <https://kanban.university/kanban-guide/>. Acesso em: 20 jan. 2025.

LEE, J.; BAGHERI, B.; KAO, H.-A. A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems. **Manufacturing Letters**, [S.l.], v. 3, p. 18-23, 2015.

MCFEE, B. *et al.* Librosa: Audio and Music Signal Analysis in Python. In: PYTHON IN SCIENCE CONFERENCE, 14., 2015, Austin. **Proceedings** [...]. Austin: SciPy, 2015. p. 18-24.

OPPENHEIM, A. V.; SCHAFER, R. W. **Discrete-Time Signal Processing**. 3. ed. Upper Saddle River: Pearson, 2010.

PDOC. **Pdoc Documentation**. 2025. Disponível em: <https://pdoc.dev/docs/pdoc.html>. Acesso em: 25 set. 2025.

PEREIRA, A. C.; SIMONETTO, E. O. Indústria 4.0: conceitos e perspectivas para o Brasil. **Revista da Universidade Vale do Rio Verde**, Três Corações, v. 16, n. 1, p. 1-12, 2018.

POSTGRESQL GLOBAL DEVELOPMENT GROUP. **PostgreSQL 16.1 Documentation**. 2025. Disponível em: <https://www.postgresql.org/docs/>. Acesso em: 11 jan. 2025.

SAFIZADEH, M. S.; LAKIS, A. A.; THOMAS, M. Using Short-Time Fourier Transform in Machinery Diagnosis. In: INTERNATIONAL CONFERENCE ON CONDITION MONITORING AND DIAGNOSTIC ENGINEERING MANAGEMENT, 18., 2005, Cranfield. **Proceedings** [...]. Cranfield: COMADEM, 2005. p. 305-314.

SCHWABER, K.; SUTHERLAND, J. **The Scrum Guide**: The definitive guide to Scrum. 2020. Disponível em: <https://scrumguides.org/scrum-guide.html>. Acesso em: 20 jan. 2025.

SMITH, S. W. **The Scientist and Engineer's Guide to Digital Signal Processing**. San Diego: California Technical Publishing, 1997.

THINGSBOARD. **ThingsBoard Documentation**. 2025. Disponível em: <https://thingsboard.io/docs/>. Acesso em: 25 set. 2025.

VAVASSORI, M. M. *et al.* Análise de falhas em equipamentos utilizando sensores de baixo custo, como método de manutenção preditiva. **Periódico Científico da UniSATC**, Criciúma, v. 7, n. 2, p. 136–194, 2022.

VISUAL STUDIO CODE. **Documentation**. 2025. Disponível em: <https://code.visualstudio.com/docs>. Acesso em: 11 jan. 2025.

Informações complementares

Descrição		Declaração
Financiamento		Este trabalho foi financiado pelo Instituto Federal de Pernambuco (IFPE), por meio de uma bolsa do Programa Institucional de Bolsas de Iniciação Científica (PIBIC).
Aprovação ética		Não se aplica.
Conflito de interesses		Não há.
Disponibilidade dos dados de pesquisa subjacentes		Não. Os dados brutos e o código-fonte do aplicativo desenvolvido fazem parte de um projeto de pesquisa em andamento. A disponibilização pública neste momento poderia comprometer o desenvolvimento futuro e a potencial proteção de propriedade intelectual associada à ferramenta.
Uso de Inteligência Artificial		Sim. Em conformidade com as diretrizes editoriais, declaramos que o uso de Inteligência Artificial neste trabalho limitou-se a funções de assistência editorial e revisão textual. Não houve geração de dados sintéticos, manipulação de imagens ou produção de texto que substituísse o julgamento técnico e a originalidade dos autores-pesquisadores. Procedimento e técnicas empregadas: • Revisão Textual: Utilizou-se ferramenta de IA (modelo de linguagem natural) para revisar a ortografia, gramática e coesão textual dos parágrafos escritos originalmente. • Formatação: A ferramenta auxiliou na padronização das referências bibliográficas e citações de acordo com as normas da ABNT solicitadas pela revista. • Tradução: A IA foi empregada para fornecer a base de tradução dos resumos para a língua inglesa (Abstract) e espanhola (Resumen), sendo o resultado final revisado quanto à terminologia técnica. • Validação: Todas as sugestões fornecidas pela IA foram analisadas e validadas pelos autores, que assumem total responsabilidade pelo conteúdo final.
CrediT	João Vitor Gouveia de Lima	Funções: conceitualização, análise formal, investigação, programas, visualização, escrita – rascunho original.
	Meuse Nogueira de Oliveira Júnior	Funções: conceitualização, aquisição de financiamento, metodologia, administração do projeto, recursos, supervisão, validação, escrita – revisão e edição.

Avaliadores: Dr. Angelo Augusto Frozza* (Instituto Federal Catarinense. Santa Catarina, Brasil). O avaliador "A" optou pela avaliação fechada e pelo anonimato.

Revisora do texto em português: Poliana Alves Brito.

Revisora do texto em inglês: Poliana Alves Brito.

Revisora do texto em espanhol: Graziani França Claudino de Anicézio.

Como citar:

LIMA, João Vitor Gouveia de; OLIVEIRA JÚNIOR, Meuse Nogueira de. Desenvolvimento de aplicativo para análise de padrões de vibração de compressores. **Revista Sítio Novo**, Palmas, v. 10, p. e1887, 2026. DOI: 10.47236/2594-7036.2026.v10.1887. Disponível em: <https://sitionovo.ifto.edu.br/index.php/sitionovo/article/view/1887>.

* Optou pela avaliação aberta e autorizou a divulgação da identidade no trabalho publicado e do parecer na página da Revista.