INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE MINAS GERAIS - *CAMPUS* SABARÁ
BACHARELADO EM ENGENHARIA DE CONTROLE E AUTOMAÇÃO

JOÃO PEDRO SILVA DE OLIVEIRA

SISTEMA DE AQUISIÇÃO DE DADOS DE TORQUE PARA EIXOS ROTATIVOS COM COMUNICAÇÃO WIRELESS E BAIXO CUSTO

SABARÁ
2025

JOÃO PEDRO SILVA DE OLIVEIRA

SISTEMA DE AQUISIÇÃO DE DADOS DE TORQUE PARA EIXOS ROTATIVOS COM COMUNICAÇÃO WIRELESS E BAIXO CUSTO

Trabalho de Conclusão de Curso apresentado à banca examinadora do curso de Engenharia de Controle e Automação do Instituto Federal de Educação, Ciência e Tecnologia de Minas Gerais *Campus* Sabará, como parte dos requisitos para obtenção do título de Bacharel em Engenharia de Controle e Automação.

Orientador: Prof. Dr. Diego Oliveira Miranda

Coorientador: Eng. Me. Moisés Martins Gonçalves

SABARÁ
2025

Oliveira, João Pedro Silva de

O48s Sistema de aquisição de dados de torque para eixos rotativos com comunicação wireless e baixo custo [manuscrito]. / João Pedro Silva de Oliveira. - 2025.

84 f. : il.

Orientação: Prof. Dr. Diego Oliveira Miranda.
Coorientador: Eng. Me. Moisés Martins Gonçalves.

Trabalho de Conclusão de Curso (Bacharelado em Engenharia de Controle e Automação) – Instituto Federal de Minas Gerais, *Campus* Sabará.

1. Veículos - Desempenho. – Monografia. 2. Força (Mecânica). – Monografia. 3. Extensômetro. – Monografia. 4. Microcontroladores. – Monografia. 5. Conversores analógicos-digitais. – Monografia. 6. Sistemas de comunicação sem fio. – Monografia. I. Miranda, Diego Oliveira. II. Gonçalves, Moisés Martins. III. Instituto Federal de Minas Gerais, *Campus* Sabará. IV. Bacharelado em Engenharia de Controle e Automação. V. Título.

CDU 681.3

César dos Santos Moreira / CRB6-2229
Biblioteca do IFMG *Campus* Sabará

MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE MINAS GERAIS
Campus Sabará
Diretoria de Ensino, Pesquisa e Extensão
Conselho de Área - Controle e Processos Industriais
Rodovia MGC 262, Km 10 - Bairro Sobradinho - CEP 34590-390 - Sabará - MG
- www.ifmg.edu.br

ATA DE DEFESA DE TRABALHO DE CONCLUSÃO DE CURSO

Aos 29 dias do mês de agosto do ano de 2025, às 14:30, nas dependência do Campus Sabará, reuniu-se a banca examinadora presidida por mim, Diego Oliveira Miranda (Orientador) Moises Martins Goncalves (Coorientador), Daniel Neves Rocha (Membro) e Maicon Vaz Moreira (Membro) Nesta ocasião o discente JOÃO PEDRO SILVA DE OLIVEIRA do curso de Bacharelado em Engenharia de Controle e Automação , com registro acadêmico de número 0059763 do IFMG – *Campus* , defendeu seu Trabalho de Conclusão de Curso

intitulado "**SISTEMA DE AQUISIÇÃO DE DADOS DE TORQUE PARA EIXOS ROTATIVOS COM COMUNICAÇÃO WIRELESS E BAIXO CUSTO** " e foi Aprovado com nota de 80 pontos.

Este resultado reflete o cumprimento parcial dos critérios de avaliação estabelecidos pelo curso e o encerramento do respectivo processo está condicionado ao cumprimento dos procedimentos pós-defesa conforme previstos nos regulamentos vigentes.

A sessão foi encerrada às 16:00. Para constar, eu, Diego Oliveira Miranda, redigi a presente ata que segue assinada pelos membros da banca examinadora.

Sabará, 01 de setembro de 2025.

Documento assinado eletronicamente por **Diego Oliveira Miranda**, **Professor EBTT**, em 01/09/2025, às 18:06, conforme Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Maicon Vaz Moreira**, **Professor EBTT**, em 01/09/2025, às 19:28, conforme Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Daniel Neves Rocha**, **Professor EBTT**, em 01/09/2025, às 21:13, conforme Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade do documento pode ser conferida no site https://sei.ifmg.edu.br/consultadocs informando o código verificador **2436894** e o código CRC **EF0F8979**.

23714.001466/2025-48 2436894v1

"O valor de uma coisa às vezes não está no que se ganha ao tê-la, mas no que se paga ao conquistá-la."
— Friedrich Nietzsche.

RESUMO

A crescente demanda por soluções eficazes e acessíveis para o monitoramento de torque em veículos automotivos impulsiona o desenvolvimento de tecnologias inovadoras. Este trabalho apresenta o projeto e a implementação de um sistema de aquisição de torque sem fio e de baixo custo, otimizado para a medição no eixo de veículos. Diferente das soluções convencionais, este sistema oferece uma alternativa prática e econômica, eliminando a necessidade de componentes com alto custo, como *slip rings* e modificações estruturais invasivas nos semieixos. A arquitetura do sistema integra *strain gauge* em configuração de ponte de *Wheatstone*, acoplado a um módulo de conversão analógico-digital HX711 de 24 *bits*. Os dados de torque são coletados e transmitidos via wireless por um microcontrolador ESP32-WROOM-32D, alimentado por uma bateria de lítio . A precisão e confiabilidade são garantidas por uma metodologia de calibração em ambiente prático e por uma interface gráfica em *Python*, que permite o processamento, análise e visualização dos dados em tempo real. Este trabalho visa fornecer uma solução de monitoramento de torque viável, com potencial aplicação em pesquisas acadêmicas e no setor automotivo, contribuindo para o avanço da instrumentação veicular e sistemas de monitoramento de desempenho.

Palavras-chave: ESP32; extensômetro; HX711; Torque; *python*; *Sensor Wireless*; Monitoramento de Desempenho Veicular.

ABSTRACT

The increasing demand for effective and affordable solutions for torque monitoring in automotive vehicles drives the development of innovative technologies. This work presents the design and implementation of a low-cost, wireless torque acquisition system, optimized for precise measurement on the front axle of front-wheel-drive vehicles. Unlike conventional solutions, this system offers a practical and economical alternative by eliminating the need for expensive components, such as slip rings, and invasive structural modifications to the half-shafts. The system's architecture integrates a strain gauge in a Wheatstone bridge configuration, coupled with a 24-bit HX711 analog-to-digital converter module for high resolution. Torque data is collected and transmitted wirelessly by an ESP32-WROOM-32D microcontroller, powered by a 3.7V lithium battery and an MT3608 converter for efficient power management. Accuracy and reliability are ensured by a practical environment calibration methodology and a graphical interface in Python, which enables real-time data processing, analysis, and visualization. This work aims to provide a viable, accessible, and efficient torque monitoring solution, with potential application in academic research and the automotive sector, contributing to the advancement of vehicular instrumentation and automotive performance monitoring systems.

Keywords: ESP32; Strain Gauge; HX711; Torque; Python; Wireless Sensor; Automotive Performance Monitoring.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

3D	Três Dimensões (utilizado no contexto de impressão 3D)
A	Ampère, unidade de medida de corrente elétrica
A/D	Conversor Analógico-Digital (*Analog-to-Digital*)
ABS	Sistema de Freios Antibloqueio (*Anti-lock Braking System*)
ADC	Conversor Analógico-Digital (*Analog-to-Digital Converter*)
ESP32	Microcontrolador com conectividade Wi-Fi e Bluetooth, utilizado para aquisição e transmissão de dados sem fio
GUI	Interface Gráfica do Usuário (*Graphical User Interface*)
HX711	Conversor analógico-digital de 24 bits com amplificação de sinal para leitura de *strain gauges*
Hz	Hertz, unidade de medida de frequência (ciclos por segundo)
I2C	Protocolo de comunicação serial de dois fios (*Inter-Integrated Circuit*)
IDE	Ambiente de Desenvolvimento Integrado (*Integrated Development Environment*), utilizado para programação de microcontroladores
IFMG	Instituto Federal de Minas Gerais
IQR	Intervalo Interquartílico
IoT	Internet das Coisas (*Internet of Things*)
KG	Quilograma, unidade de medida de massa
M2M	Comunicação Máquina a Máquina (*Machine-to-Machine*) – comunicação direta entre dispositivos sem intervenção humana
MAD	Desvio Absoluto da Mediana (*Median Absolute Deviation*), utilizado para filtragem estatística de dados)
MAE	Erro Absoluto Médio MAE (*Mean Absolute Error*)
N	Newton, unidade de medida de força
OLS	Mínimos Quadrados Ordinários (*Ordinary Least Squares*), método estatístico de regressão linear
TCC	Trabalho de Conclusão de Curso

TCP	Protocolo de Controle de Transmissão (*Transmission Control Protocol*)
USB	Barramento Serial Universal (*Universal Serial Bus*), utilizado para comunicação entre dispositivos
V	Volt, unidade de medida de tensão elétrica
Wi-Fi	Tecnologia de comunicação sem fio baseada no padrão IEEE 802.11
CAN	*Controller Area Network* – Protocolo de comunicação entre módulos eletrônicos, usado em veículos e sistemas embarcados.

SUMÁRIO

1 INTRODUÇÃO

Historicamente, a variação do deslocamento angular entre as extremidades de um eixo sujeito à torção foi utilizada para determinar o torque por meio de métodos como medições fotoelétricas, eletromagnéticas ou indutivas (NORTON, 1989). Apesar dos avanços nas tecnologias de medição de torque com dispositivos sofisticados, os *strain gauges* continuam a ser empregados na indústria devido à sua simplicidade operacional, baixo custo e precisão na detecção de torque aplicado (MORRIS, 1991). Os *strain gauges* resistivos fornecem um sinal de saída proporcional à deformação relativa do material, pois a variação de sua resistência elétrica acompanha a deformação mecânica sofrida (WILSON, 2005). Essa característica os diferencia de outros tipos de extensômetros, como os mecânicos ou indutivos, nos quais a deformação precisa ser inferida indiretamente a partir de deslocamentos, variações de pressão ou frequência.

O uso de sensores na indústria automotiva está em expansão, contribuindo para o aprimoramento do desempenho e da segurança dos veículos. Com a crescente integração de sistemas computacionais nos automóveis, os sensores desempenham um papel crucial ao fornecer informações detalhadas sobre o estado operacional do veículo. Eles monitoram uma variedade de parâmetros, como por exemplo o desempenho do motor e a velocidade das rodas, além de fornecer dados essenciais para sistemas de controle avançados como o ABS (*Anti-lock Braking System*) e o ESP (*Electronic Stability Program*). Esses sensores fornecem dados importantes para sistemas de controle que influenciam o desempenho e a segurança veicular.

A quantidade de sensores incorporados nos veículos tem aumentado substancialmente ao longo das últimas décadas. Em um modelo de luxo de 1995, o número de sensores era aproximadamente dez, focados em funções como o monitoramento do motor e dos sistemas de frenagem. Por volta de 2010, esse número já havia aumentado para cerca de 30 sensores. Atualmente, veículos modernos estão equipados com mais de 100 sensores, que monitoram e controlam virtualmente todos os sistemas do automóvel (COSTA, 2024).

A medição precisa do torque é crucial para garantir o desempenho e a segurança dos veículos, e a demanda por soluções que sejam ao mesmo tempo eficazes e econômicas está crescendo no setor automotivo. Neste contexto, este trabalho propõe um sensor de torque sem fio que supera as limitações das tecnologias atuais, como o alto custo e a complexidade dos *slip rings*. A ideia é oferecer um protótipo para teste e validação de componentes em veículo prototípico, usando o ESP32 para transmitir os dados sem fio e componentes que são baratos e de fácil acesso.

O sensor desenvolvido busca um equilíbrio entre precisão e custo, facilitando sua integração em aplicações automotivas. A proposta atende a demandas atuais da indústria, oferecendo uma alternativa para medição de torque sem fio. Este desenvolvimento pode contribuir para futuras melhorias em sensores automotivos.

1.1 Objetivos

1.1.1 *Objetivo geral*

Desenvolver um sensor de torque de baixo custo e preciso para veículos em estágio prototípico, utilizando componentes eletrônicos econômicos, com transmissão de dados via comunicação sem fio. O objetivo é substituir sensores tradicionais que demandam dispositivos caros e complexos, como *slip rings* e sensores *wireless*, oferecendo uma solução mais viável e simplificada para medições de torque.

1.1.2 *Objetivos específicos*

1. **Projeto e Construção do Protótipo:**
    - Desenvolver um sistema de aquisição de dados de torque funcional utilizando um módulo conversor HX711 e uma célula de carga, integrando-os com um microcontrolador ESP32 para transmissão sem fio;
    - Criar o circuito eletrônico necessário e instalar o sensor em um veículo.
    - Programar o ESP32 para a coleta e transmissão de dados de torque via comunicação sem fio;
2. **Implementação de Software:**
    - Desenvolver o código em Python utilizando o Visual Studio Code, implementando a leitura e o tratamento dos dados provenientes do módulo HX711 conectado ao ESP32.
3. **Calibração e Validação do Sistema:**
    - Implementar um método prático de calibração para correlacionar os dados da deformação com os valores reais de torque.
    - Validar a resposta do sistema em ambiente controlado, verificando a linearidade, repetibilidade e precisão das medições.
4. **Comparação de Desempenho:**
    - Comparar o desempenho do sensor desenvolvido com sensores tradicionais, focando em aspectos como custo, precisão e facilidade de instalação.
    - Realizar testes comparativos e analisar os resultados para avaliar a viabilidade do sensor proposto.

1.2 Justificativa

O desenvolvimento de um sensor de torque de baixo custo para eixos de veículos oferece uma alternativa de baixo custo para medição de torque em veículos e para o campo dos sensores. O microcontrolador ESP32, conhecido por suas capacidades de processamento de dados, tem sido usado com sucesso em diversas aplicações, incluindo sistemas de monitoramento residencial inteligente. Este projeto propõe uma alternativa econômica para medição de torque, que pode ampliar as possibilidades de monitoramento e análise do desempenho veicular. (MAIER; SHARP; VAGAPOV, 2017).

Além de sua contribuição para o conhecimento acadêmico, o projeto pode ser adaptado para diferentes tipos de veículos e condições de teste na engenharia automotiva. A acessibilidade do sensor *strain gauge* proposto permite sua aplicação em veículos de baixo custo e em cenários de teste mais rigorosos. Além disso, o sensor pode ser usado em diversas mudanças de temperatura (MUFTAH; AL., 2018) e é insensível a torções ou tensões axiais (PASTOR *et al.*, 2023), anteriormente limitados por tecnologias caras e complexas. O projeto pode auxiliar no desenvolvimento de sensores e tecnologias relacionadas na área.

De acordo com Dhull et al. (DHULL *et al.*, 2022), a tendência de crescimento exponencial dos dispositivos conectados à Internet das Coisas (*IoT*) reforça a necessidade de soluções inovadoras que unam eficiência energética e comunicação sem fio. Nesse cenário, tecnologias como o SWIPT (*Simultaneous Wireless Information and Power Transfer*) surgem como alternativas para viabilizar sensores em larga escala, especialmente em ambientes automotivos e industriais. O presente trabalho acompanha essa evolução ao propor uma solução de aquisição de torque sem fio, de baixo custo e energeticamente viável, utilizando o módulo ESP32. A proposta alinha-se à demanda por sistemas autônomos e conectados, destacando-se pelo detalhamento de sua implementação, incluindo o projeto eletrônico, a calibração via regressão linear e a interface gráfica para aquisição dos dados. Assim, este TCC contribui diretamente com a comunidade acadêmica e com o avanço de aplicações práticas em IoT para a engenharia automotiva.

Os produtos resultantes deste trabalho incluem um protótipo funcional do sensor de torque e uma documentação abrangente do desenvolvimento. Esses produtos oferecem uma protótipo de baixo custo que pode ser replicado em outros estudos. Este trabalho contribui para o desenvolvimento de sensores e pode servir como referência para pesquisas futuras.

Este projeto é importante porque oferece uma solução prática e acessível para a medição de torque, útil para aplicações acadêmicas e testes industriais. A inovação trazida por este projeto pode diminuir o custo e complexidade das medições de torque, além de contribuir para o desenvolvimento de veículos mais acessíveis e tecnologicamente avançados. Para o autor, representa uma oportunidade de se destacar na área da engenharia automotiva, demonstrando habilidades no desenvolvimento de sensores e aplicação prática de conhecimentos acadêmicos. Para as empresas, oferece a chance de reduzir custos e melhorar a qualidade dos produtos. Para a

sociedade, as mudanças incluem a possibilidade de veículos mais acessíveis e o estímulo a novas pesquisas e inovações.

1.3 Organização do Texto

Este trabalho está organizado em seis capítulos que cobrem o desenvolvimento e implementação de um sensor de torque sem fio para eixos automotivos. O primeiro capítulo aborda a introdução do tema, apresentando a importância da medição de torque no contexto automotivo. São discutidos os objetivos do trabalho, que incluem a criação de um sistema de medição de torque de baixo custo, utilizando o ESP32, um microcontrolador eficiente, e o amplificador HX711 para leitura de strain gauges.

No segundo capítulo, é feita uma revisão bibliográfica sobre os avanços recentes na área de sensores de torque. Este capítulo explora as inovações tecnológicas que têm facilitado a aplicação de sensores em contextos automotivos e industriais. Além disso, estudos anteriores são analisados para fornecer uma base teórica sólida, garantindo que o desenvolvimento do sensor proposto esteja em linha com as pesquisas mais recentes.

No terceiro capítulo, a fundamentação teórica é apresentada com explicações detalhadas sobre o funcionamento dos strain gauges e da ponte de Wheatstone, que formam a base do sistema de medição. Em seguida, é descrito o papel do amplificador HX711, responsável por converter o sinal analógico em digital com alta resolução, e do ESP32, que possibilita a comunicação sem fio com o computador via Wi-Fi. Também são abordados os conceitos de torque e sua aplicação no eixo de tração de automóveis, evidenciando a importância da sua medição precisa para testes de desempenho. Além desses elementos, são exploradas técnicas estatísticas utilizadas no tratamento dos dados adquiridos, como o MAD (*Median Absolute Deviation*), que permite identificar e remover valores atípicos (*outliers*) com base na mediana e no desvio absoluto, aumentando a confiabilidade das medições. Por fim, é apresentada a regressão linear como método de calibração do sistema, sendo aplicada com sete cargas conhecidas para gerar uma equação que correlaciona os valores brutos do sensor com os valores reais de torque, garantindo que as leituras em tempo real reflitam com precisão as condições físicas do sistema.

O quarto capítulo detalha a metodologia adotada neste trabalho, apresentando todas as etapas do projeto de forma sistemática. Inicialmente, é descrita a instrumentação do eixo com os strain gauges, seguida da conexão e integração do módulo HX711 com o microcontrolador ESP32 para a aquisição dos sinais analógicos convertidos em digitais. Em seguida, aborda-se o processo de calibração dos sensores, que envolve a coleta de dados com cargas conhecidas e a aplicação de técnicas estatísticas para garantir a precisão das medições. Também é detalhado o desenvolvimento do software em Python, responsável pela captura, filtragem e processamento dos dados, incluindo a implementação da interface gráfica para visualização em tempo real. Essa metodologia assegura que todos os componentes do sistema estejam devidamente sincronizados

e calibrados, garantindo a confiabilidade e qualidade dos dados obtidos na fase de coleta final.

O quinto capítulo apresenta os resultados obtidos com o sistema de aquisição de torque sem fio baseado no ESP32, incluindo testes práticos em altas rotações realizados com um motor que demonstraram a estabilidade da conexão sem perda de dados; os valores medidos foram comparados com os de um sistema convencional com fio e strain gauge, revelando boa correlação e precisão equivalente. Também foi realizada uma análise comparativa de custos entre a solução desenvolvida e sistemas tradicionais do mercado, evidenciando a vantagem econômica da proposta. Adicionalmente, foram avaliados o consumo energético e a durabilidade da bateria de lítio, que mostraram baixo consumo e autonomia adequada para aplicações práticas, confirmando a funcionalidade do sensor e fornecendo uma base consistente para as análises críticas e considerações finais do trabalho.

Por fim, o sexto capítulo traz as conclusões do trabalho, resumindo os principais resultados alcançados e refletindo sobre o processo de desenvolvimento. Além disso, são feitas considerações sobre possíveis melhorias e perspectivas para trabalhos futuros, como o aprimoramento do sensor de torque e sua aplicação em outros contextos além do automotivo.

2 REVISÃO BIBLIOGRÁFICA

O desenvolvimento de sensores de torque para aplicações automotivas sem fio demanda uma sólida base teórica e prática, com raízes históricas profundas nas técnicas de medição de forças. A evolução dessas técnicas é fundamental para compreender o contexto atual da instrumentação veicular. Um marco no desenvolvimento, importante para a fundamentação dos sensores de torque modernos, é o estudo de Clark e Duwez (1948). Eles apresentaram um método pioneiro para medir forças dinâmicas em testes de impacto de tração utilizando um dinanômetro, comparando relações força-tempo teóricas e experimentais. Este trabalho pioneiro estabeleceu fundamentos essenciais para a compreensão e evolução dos *strain gauges* e sua aplicação em sistemas de medição sofisticados, fornecendo base para o desenvolvimento de tecnologias subsequentes na tecnologia de sensores de força e torque.

A compreensão dos fundamentos de sensores, indispensáveis para este projeto, é amplamente discutida por Wilson (2005) em seu *Sensor Technology Handbook*. Em particular, o Capítulo 19 detalha o funcionamento, aplicação e técnicas de calibração de *strain gauges*, elementos centrais para a detecção de deformações no eixo do veículo. Complementarmente, o Capítulo 22 aprofunda-se em sensores sem fio, abordando as tecnologias de transmissão de dados sem fio que são cruciais para a integração do microcontrolador *ESP32* no sistema proposto. A consolidação dessas tecnologias de sensoriamento e comunicação sem fio é um pilar para soluções automotivas que buscam facilidade de instalação e na coleta de dados.

A demanda por sensores de torque de baixo custo tem aumentado na área, conforme explorado por Pastor *et al.* (2023). Embora focados em manipuladores, o estudo oferece *insights* sobre métodos de calibração e técnicas de medição que podem ser adaptadas para aplicações automotivas. A relevância desse trabalho reside na sua abordagem econômica e na integração com sistemas de controle, alinhando-se diretamente aos objetivos de desenvolver uma solução de baixo custo para medição de torque veicular, com potencial para integração em sistemas de gestão. No contexto mais amplo, Stone e Ball (2004) fornecem uma base teórica essencial sobre sistemas de medição e controle. Sua obra destaca a importância da precisão nas medições de torque para a avaliação do desempenho veicular e a integridade dos componentes de transmissão, princípios cruciais para a correta aquisição e interpretação dos dados em testes veiculares.

A metodologia experimental apresentada por Pastor *et al.* (2023), que consiste na aplicação de torques em etapas controladas e na análise da resposta dinâmica dos *strain gauges*, será utilizada como referência para o desenvolvimento dos testes do presente trabalho. Embora o contexto de aplicação seja distinto, manipuladores robóticos no estudo original e eixos automotivos neste projeto, os princípios adotados para avaliar a eficiência, precisão e repetibilidade do sensor mostram-se amplamente aplicáveis ao sistema proposto. A forma como os autores validam a confiabilidade do sensor, por meio de análises sucessivas de carga e descarga com controle das variáveis envolvidas, fornece uma base sólida para estruturação dos testes deste trabalho. Nesse sentido, a Figura 1, foi extraída do artigo de Pastor *et al.* (2023) e corresponde a um teste

de repetibilidade realizado pelos autores com sensores de torque baseados em *strain gauges*, aplicado a manipuladores robóticos.

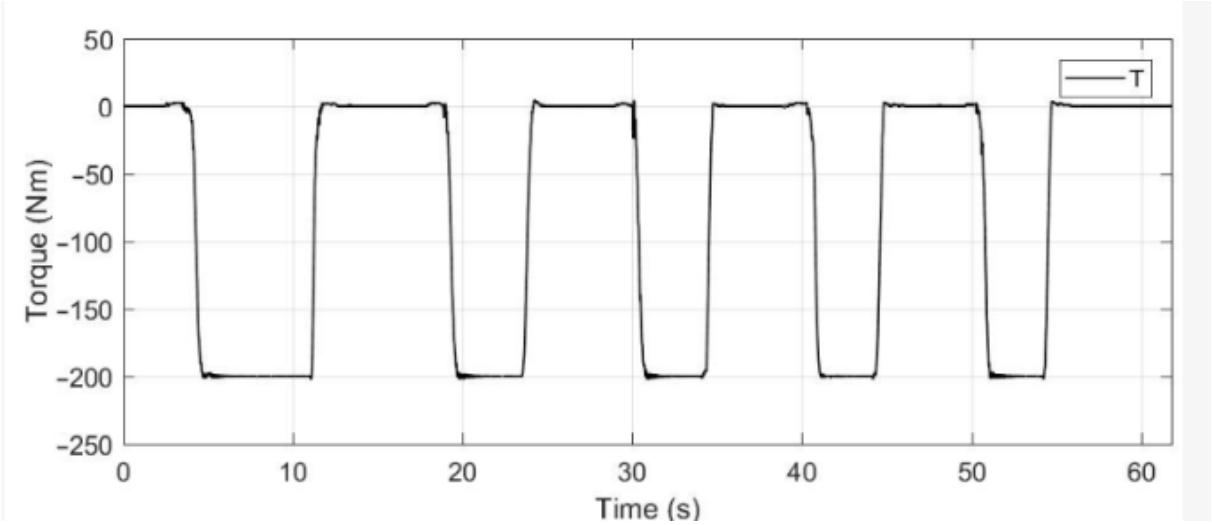

Figura 1 – Resposta dinâmica dos *strain gauges* durante teste de repetibilidade. Fonte:Pastor *et al.* (2023)

A integração de microcontroladores em sistemas embarcados é um ponto fundamental para a coleta e transmissão de dados sem fio. Mazidi, Mazidi e McKinney (2016) oferecem uma visão abrangente sobre o uso e a programação de microcontroladores como o *ESP32* em sistemas embarcados. Complementarmente, embora com foco em plataformas como Arduino, Blum (2013) apresenta técnicas para o desenvolvimento de sistemas de sensores e controle que podem ser adaptadas. A aplicação do *ESP32* em sistemas de monitoramento inteligente é demonstrada por El-Khozondar e al. (2022), reforçando sua capacidade para coleta e transmissão de dados. A ascensão do IoT impulsionou o desenvolvimento de redes de sensores sem fio, tema explorado por Singh e al. (2020) no contexto de sistemas de transporte e por Liu e al. (2020) em sistemas de transmissão de energia, fornecendo perspectivas aplicáveis a veículos automotivos. Os desafios e arquiteturas dessas redes são discutidos por Manshahia (2020) e revisados por Khalifeh e al. (2021) no contexto de microcontroladores. Um aspecto vital para a operação contínua de sensores sem fio, a alimentação sem fio, é abordado por Popovic e al. (2019), enquanto Escobedo e al. (2021) apresentam a aplicação de sensores sem fio em dispositivos médicos, oferecendo *insights* para a integração e comunicação em sistemas de medição diversos.

Para a garantia de medições precisas e confiáveis, a calibração e validação de sensores são aspectos críticos. Doebelin (2003) explora metodologias de medição e a importância da precisão nos sistemas de medição. Doebelin aborda aspectos críticos como métodos de calibração e precição de medição, que são fundamentais para garantir a performance e a confiabilidade dos sensores. Santamaría e al. (2020) detalha métodos de calibração para balanças de *strain gauge*, diretamente relevantes para o sensor de torque. Além disso, Muftah e al. (2018) apresenta

métodos aprimorados para a medição dinâmica de torque com *strain gauges*, fornecendo técnicas avançadas para melhorar a precisão e a eficácia do sistema de medição. Adicionalmente, a fim de mitigar o impacto de ruídos e perturbações inerentes a ambientes automotivos, a robustez para medição é crucial. A detecção de *outliers* nos dados coletados, empregando métodos estatísticos robustos como o *Median Absolute Deviation* (*MAD*), torna-se um passo fundamental para garantir a confiabilidade e a precisão das leituras de torque. O *MAD*, em particular, é uma medida de dispersão menos suscetível a valores extremos em comparação com o desvio padrão, tornando-o ideal para identificar anomalias em séries temporais de dados de sensores (ROMO-CHAVERO *et al.*, 2024). Persson e Persson (2019) investiga técnicas específicas para a medição de torque em sistemas de transmissão automotiva, oferecendo metodologias diretamente aplicáveis ao desenvolvimento do sensor de torque para o eixo do veículo. Complementarmente, o entendimento dos circuitos eletrônicos para amplificação e transmissão de sinais é fundamental, e Sedra e Smith (2017) é uma referência essencial para o entendimento dos circuitos eletrônicos envolvidos na amplificação e transmissão dos sinais dos sensores (SEDRA; SMITH, 2017). Compreender os conceitos apresentados neste livro é crucial para o desenvolvimento do circuito eletrônico que integra o *HX711* e o *ESP32* (SEDRA; SMITH, 2017). O livro aborda em detalhes os princípios de circuitos eletrônicos, oferecendo uma base sólida para a implementação de circuitos amplificadores e de transmissão de sinais, que são fundamentais para a operação eficiente dos sensores de torque. As técnicas de amplificação de sinal e transmissão de sinal descritas por Sedra e seus co-autores são particularmente relevantes para o seu projeto, fornecendo o conhecimento necessário para otimizar o desempenho e a precisão do sistema de sensores.

Em síntese, a literatura revisada estabelece uma robusta base teórica e prática para o desenvolvimento de sistemas de medição de torque, desde os fundamentos dos extensômetros e microcontroladores até as complexidades das redes sem fio e calibração. Contudo, as soluções existentes frequentemente apresentam alto custo ou são invasivas, o que representa uma lacuna para aplicações de larga escala. Nesse contexto, a abordagem inovadora deste TCC se destaca ao propor um sistema de sensor de torque sem fio de baixo custo, que permite a calibração fora do veículo e a integração fluida com o *ESP32*, superando as limitações das metodologias tradicionais. Este projeto não apenas aplica e adapta as técnicas apresentadas na literatura, mas também avança a instrumentação veicular ao oferecer uma solução prática e otimizada para o monitoramento de torque em eixos.

3 FUNDAMENTAÇÃO TEÓRICA

3.1 Strain Gauge

O *strain gauge* demonstrado na Figura 2 é um sensor amplamente utilizado para medir deformações em materiais demonstrado na . Seu princípio de funcionamento baseia-se na variação da resistência elétrica de um condutor quando este é submetido a uma deformação mecânica. A deformação gera uma alteração no comprimento e na seção transversal do condutor, o que leva a uma mudança proporcional em sua resistência elétrica (WILSON, 2005).

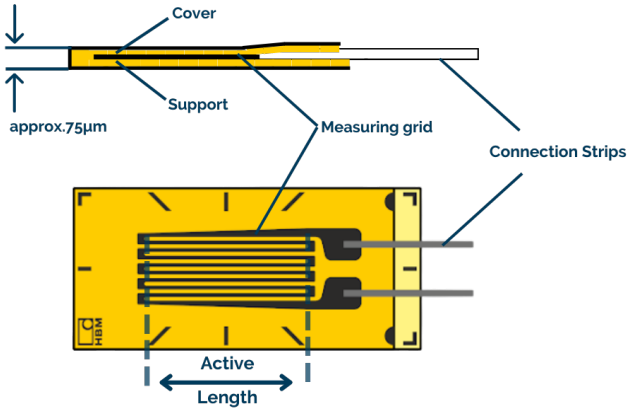

Figura 2 – Construção de um strain gauge. Fonte: (T&M, 2024)

Os *strain gauges* mais comuns são feitos de ligas metálicas, como o *Constantan* (cobre-níquel) ou o *Nichrome* (níquel-cromo). Quando uma força é aplicada sobre o material onde o sensor está instalado, ocorre uma mudança no comprimento L e, consequentemente, na resistência R, de acordo com a equação 3.1:

$$\frac{\Delta R}{R} = GF \cdot \frac{\Delta L}{L} \tag{3.1}$$

onde GF é o *fator de gauge*, um parâmetro que relaciona a deformação mecânica com a variação da resistência. Para metais, o valor típico do *fator de gauge* varia entre 2,0 e 4,5, enquanto para materiais semicondutores pode ultrapassar 150 (WILSON, 2005). Isso torna os semicondutores mais sensíveis a pequenas deformações, o que pode ser vantajoso em algumas aplicações.

Os *strain gauges* são frequentemente organizados em uma configuração de ponte de Wheatstone para maximizar a sensibilidade e garantir uma medição precisa, mesmo em ambientes onde variações de temperatura podem influenciar o resultado. Nesta configuração, quatro *strain gauges* são usados para medir, por exemplo, a flexão de uma viga, com dois sensores localizados na parte superior e dois na parte inferior da estrutura, o que resulta em uma leitura linear e compensada termicamente.

A instalação dos *strain gauges* é um processo crítico para garantir a precisão das medições. Ela requer uma preparação da superfície onde o sensor será fixado, incluindo processos de limpeza

e aplicação de adesivos especiais. Além disso, é necessário proteger o sensor contra efeitos ambientais, como umidade e temperaturas extremas, que poderiam comprometer os resultados (WILSON, 2005).

Os *strain gauges* são amplamente utilizados em diversas indústrias para medir força, pressão e torque. No setor automotivo, por exemplo, esses sensores são fundamentais para a medição de torque em eixos, fornecendo dados essenciais para o desenvolvimento de novos componentes de transmissão e para a validação de projetos (WILSON, 2005).

3.2 Ponte de Wheatstone

O circuito de ponte de Wheatstone é uma configuração amplamente utilizada para medir resistências com alta precisão, comparando uma resistência desconhecida com resistores de valores conhecidos (WILSON, 2005). A estrutura do circuito é formada por quatro resistores dispostos em uma configuração de ponte. Dois desses resistores são conhecidos (*R1* e *R2*), enquanto o terceiro (*R3*) é o resistor cuja resistência queremos medir, e o quarto (*R4*) é um resistor de ajuste.

A tensão de entrada V_{in} é aplicada entre R1/R3 e R2/R4 conforme a Figura 3, e a tensão de saída V_{out} é medida entre R1/R2 e R3/R4,ou seja a região G. Quando o circuito está em equilíbrio, a tensão na região G é zero. Isso ocorre quando a relação entre os resistores é:

$$\frac{R1}{R2} = \frac{R3}{R4}. \tag{3.2}$$

Neste estado, a tensão de saída V_{out} é:

$$V_{out} = 0. \tag{3.3}$$

Se o circuito não estiver em equilíbrio, haverá uma tensão diferencial entre R1/R2 e R2/R4, ou seja, na região G. Essa tensão pode ser calculada pela equação 3.4:

$$V_{out} = V_{in} \times \frac{R3 \cdot R2 - R1 \cdot R4}{(R1 + R2) \cdot (R3 + R4)} \tag{3.4}$$

A Figura 3 ilustra a configuração do circuito de ponte de Wheatstone, evidenciando a disposição dos resistores e a aplicação da tensão.

Esse circuito apresenta alta precisão. A configuração da ponte permite que pequenas mudanças na resistência de *R3* sejam detectadas com exatidão, já que qualquer variação na resistência *R3* resulta em uma alteração proporcional na tensão de saída V_{out}. Além disso, a ponte proporciona uma boa compensação de erro comuns, como variações de temperatura, uma vez que quaisquer mudanças nos resistores afetam a ponte de maneira equilibrada. Essa capacidade de minimizar erros comuns e melhorar a sensibilidade do sistema faz com que o circuito de ponte de Wheatstone seja uma escolha confiável para medições de alta precisão (OBERST, 2010).

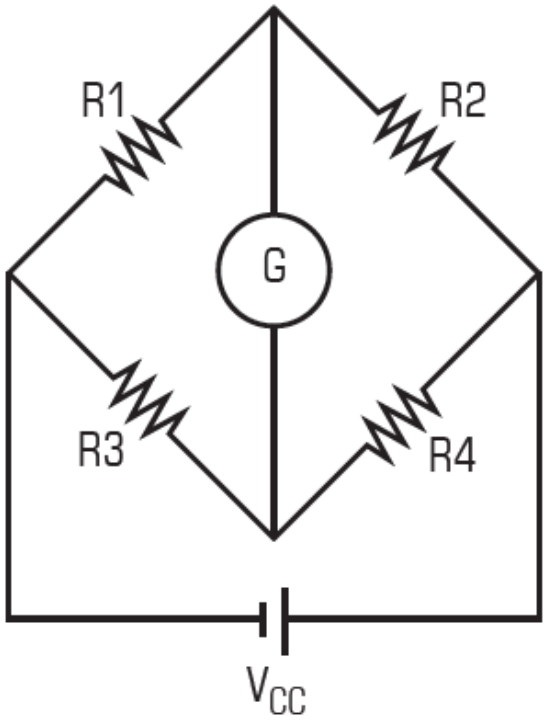

Figura 3 – Ponte de Wheatstone. Fonte: (MMA, 2024).

3.3 Amplificador HX711

O *HX711* é um amplificador de precisão amplamente utilizado em sistemas de medição de força e pesagem, como balanças digitais e outros sensores baseados em células de carga. Desenvolvido para converter sinais analógicos de baixo nível provenientes desses sensores, o *HX711* utiliza um conversor analógico-digital (A/D) de 24 bits, proporcionando medições altamente precisas e estáveis (ELECTRONICS, 2024).

O *HX711* se destaca pelas seguintes características:

- **Alta resolução**: Com um conversor A/D de 24 bits, o *HX711* é capaz de detectar pequenas variações de sinal, tornando-o ideal para aplicações que requerem medições de precisão, como sistemas de pesagem e monitoramento de torque.
- **Baixo consumo de energia**: Projetado para operar em sistemas de baixa potência, o *HX711* pode ser alimentado por fontes que variam de 2,7 V a 5,5 V, garantindo flexibilidade em projetos com restrições de energia (ELECTRONICS, 2024).
- **Interface simplificada**: A comunicação com microcontroladores, como o *ESP32*, é realizada por meio de uma interface de dois pinos (*DAT* e *CLK*), facilitando a integração com sistemas embarcados. Além disso, o *HX711* conta com um ganho programável que permite amplificar os sinais de entrada de forma adequada para a aplicação (ELECTRONICS, 2024).

O *HX711* exemplificado pela figura 4 é amplamente utilizado em sistemas de medição de força, especialmente em projetos que utilizam *strain gauges*, como o sensor de torque proposto neste trabalho. A alta resolução e a capacidade de amplificação permitem que ele detecte com precisão pequenas deformações mecânicas no eixo do veículo, transformando essas variações em sinais digitais que podem ser processados pelo microcontrolador *ESP32*.

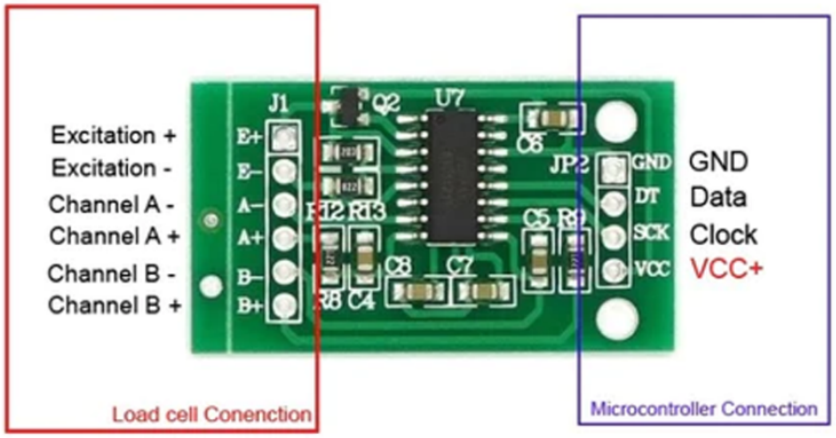

Figura 4 – Amplificador HX711. Fonte: (SANTAMARíA; AL., 2020).

Abaixo, apresentamos a descrição dos pinos do HX711, que é um amplificador de sinal utilizado em medições de células de carga. O dispositivo possui um total de 8 pinos principais, cada um com uma função específica:

- **VCC**: Fonte de alimentação (conectar ao VCC do circuito).
- **GND**: Terra do circuito.
- **DT**: Saída de dados.
- **SCK**: Pino de clock.
- **A+**: Conectar ao terminal positivo da célula de carga.
- **A-**: Conectar ao terminal negativo da célula de carga.
- **B+**: Conectar ao terminal positivo de uma segunda célula de carga (se usada).
- **B-**: Conectar ao terminal negativo de uma segunda célula de carga (se usada).
- **E**: Conectar à fonte de alimentação para excitação.

3.4 Microcontrolador ESP32

O *ESP32* conforme ilustrado pela Figura 5 é um microcontrolador de baixo custo, desenvolvido pela *Espressif Systems*, amplamente utilizado em aplicações de IoT e sistemas embarcados. Ele oferece suporte integrado para *Wi-Fi* e *Bluetooth*, tornando-o ideal para projetos que requerem conectividade sem fio, como o sistema de medição de torque proposto neste trabalho (SYSTEMS, 2024b).

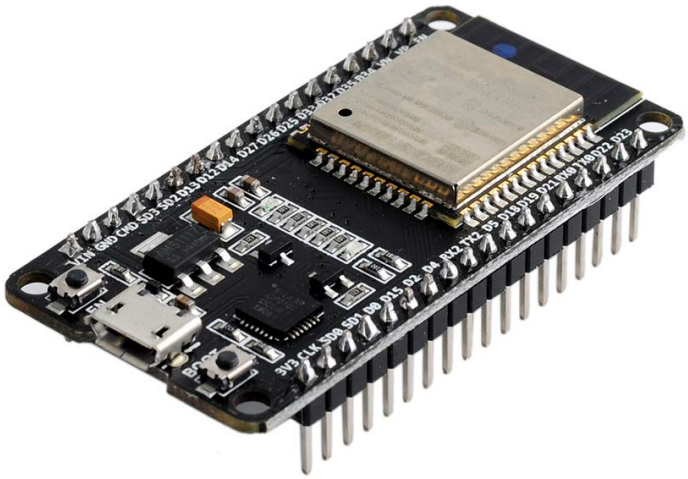

Figura 5 – Esp32. Fonte: (SYSTEMS, 2024a).

O *ESP32* possui várias características que o tornam uma escolha para projetos de aquisição e transmissão de dados sem fio(SYSTEMS, 2024b):

- **Processador de alto desempenho**: Equipado com um processador *Xtensa® 32-bit LX6*, o *ESP32* pode operar com dois núcleos de processamento em até 240 MHz, garantindo alta capacidade de processamento para tarefas de controle e comunicação.
- **Conectividade sem fio**: Suporte integrado para *Wi-Fi* 802.11 b/g/n e *Bluetooth* 4.2 (incluindo *Bluetooth Low Energy*), permitindo que o *ESP32* transmita dados de maneira eficiente para dispositivos externos, como computadores ou servidores.
- **Baixo consumo de energia**: O *ESP32* oferece modos de baixa potência, como *Deep-sleep*, o que é especialmente útil em aplicações de monitoramento contínuo, onde a economia de

energia é essencial.

- **Interface de comunicação versátil**: O *ESP32* possui várias interfaces de comunicação, incluindo *UART*, *SPI*, *I2C*, e *PWM*, permitindo a integração com diversos sensores e dispositivos externos. Essa versatilidade facilita a conexão com o amplificador *HX711* utilizado neste projeto.

No contexto deste projeto, o *ESP32* será utilizado para receber os sinais amplificados provenientes do *HX711*, que converte o sinal analógico dos *strain gauges* em dados digitais. A conectividade sem fio do *ESP32* será aproveitada para transmitir os dados adquiridos para um computador, onde serão tratados e analisados a linguagem de programação *Python*. Essa abordagem facilita a coleta de dados de torque de maneira remota e em tempo real, sem a necessidade de cabos conectados diretamente ao veículo.

3.5 Conceitos de Torque

O torque, também conhecido como momento de força conforme ilustrado pela Figura 6 , é uma medida da força rotacional que age sobre um objeto em torno de um eixo. É uma quantidade vetorial que pode ser calculada pela equação 3.5:

$$T = F \times r \times \sin(\theta) \tag{3.5}$$

. onde:

- T representa o torque aplicado,
- F é a força aplicada perpendicularmente ao eixo,
- r é a distância do ponto de aplicação da força ao eixo de rotação,
- θ é o ângulo entre a direção da força e o braço de alavanca.

No contexto de um eixo de transmissão, o torque é essencial para compreender como a força gerada pelo motor é transferida para as rodas do veículo (STONE; BALL, 2004).

Quando um torque é aplicado ao longo do comprimento de um eixo, ele provoca uma deformação angular no eixo, conhecida como torção. A deformação angular (θ) pode ser descrita pela seguinte equação 3.6:

$$\theta = \frac{T \cdot L}{G \cdot J} \tag{3.6}$$

onde:

- θ é a deformação angular em radianos,

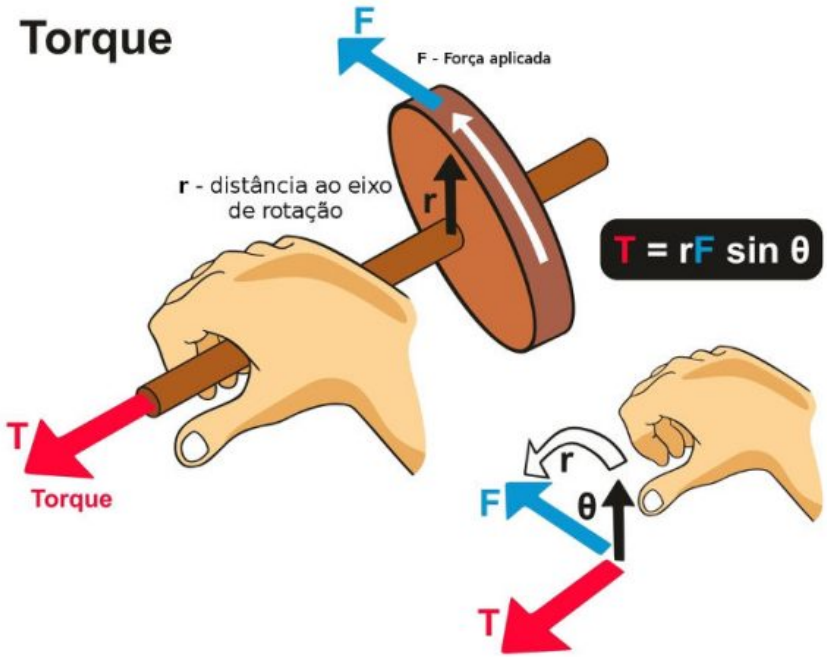

Figura 6 – O que é o Torque?. Fonte: (AUTOS, 2024).

- T é o torque aplicado ao eixo,
- L é o comprimento do eixo,
- G é o módulo de elasticidade em cisalhamento do material do eixo,
- J é o momento de inércia polar da seção transversal do eixo.

O momento de inércia polar (J) é uma medida que descreve como a massa está distribuída em torno do eixo de rotação e afeta a resistência do eixo à torção. Para um eixo circular sólido, o momento de inércia polar é calculado pela equação 3.7:

$$J = \frac{\pi \cdot d^4}{32} \tag{3.7}$$

onde d é o diâmetro do eixo.

A equação 3.7 é amplamente usada em engenharia para descrever a distribuição de massa e a resistência à torção (STONE; BALL, 2004).

Como o objetivo do sistema desenvolvido é medir o torque em um ponto específico do eixo rotativo, a relação acima pode ser invertida para calcular o torque a partir da deformação angular medida localmente. Assim, o torque é calculado pela equação 3.7:

$$T = \frac{G \cdot J}{L} \cdot \theta \tag{3.8}$$

Essa abordagem permite que o torque aplicado seja determinado diretamente a partir da medição local da torção, garantindo precisão e especificidade na avaliação do esforço mecânico naquele ponto do sistema.

A relação entre o torque aplicado e a deformação angular é crucial para avaliar o desempenho e a durabilidade do eixo. A equação 3.8 mostrada demonstra que a deformação angular aumenta com o aumento do torque ou do comprimento do eixo e diminui com o aumento do módulo de elasticidade em cisalhamento ou do momento de inércia polar (STONE; BALL, 2004).

A medição precisa do torque é vital para avaliar o desempenho e a integridade dos componentes da linha de transmissão. Sensores de torque, como os baseados em *strain gauges*, são essenciais para capturar as variações no torque aplicado e fornecer dados precisos para a análise e otimização dos sistemas de transmissão (WILSON, 2005).

3.6 Software Arduino IDE

O *Arduino IDE* (Integrated Development Environment) é uma plataforma de desenvolvimento de código aberto que facilita a programação de microcontroladores e placas de prototipagem, como o ESP32. O ambiente pode ser baixado e instalado a partir do site oficial da Arduino (ARDUINO, 2024). Este IDE fornece um ambiente acessível para a escrita, compilação e *upload* de código para microcontroladores, permitindo um desenvolvimento eficiente de projetos eletrônicos e sistemas embarcados.

Figura 7 – Arduino IDE. Fonte: (PLUSPNG, 2024).

O *Arduino IDE* possui diversas características que são cruciais para o desenvolvimento com o ESP32 (ARDUINO, 2024):

- **Interface Gráfica Intuitiva**: O IDE oferece uma interface gráfica de usuário que inclui um editor de código, um monitor serial e ferramentas para compilação e *upload*. A interface foi projetada para ser simples e acessível, facilitando o processo de desenvolvimento e depuração.

- **Bibliotecas e Suporte a Diversas Placas**: O IDE é compatível com uma ampla gama de bibliotecas e placas, incluindo o ESP32. Bibliotecas são pacotes de código que fornecem funcionalidades específicas, como comunicação sem fio e controle de sensores. O suporte a múltiplas placas permite a reutilização de código e a simplificação do desenvolvimento.
- **Compilação e Upload Simplificados**: O IDE realiza a compilação do código fonte e o converte em um formato que pode ser carregado no microcontrolador. O processo de *upload* é automatizado, o que reduz a complexidade e o tempo necessário para programar o microcontrolador.
- **Monitor Serial**: O monitor serial integrado ao IDE permite a comunicação em tempo real entre o computador e o microcontrolador. Esse recurso é essencial para a depuração e o monitoramento dos dados gerados pelo ESP32.
- **Comunidade e Suporte**: O *Arduino IDE* é amplamente suportado por uma comunidade ativa e por diversos recursos educacionais, como tutoriais e fóruns. Essa ampla base de apoio facilita a resolução de problemas e o aprendizado contínuo.

Para utilizar o *Arduino IDE* com o ESP32, é necessário configurar o ambiente de desenvolvimento adequadamente. O processo inclui(SYSTEMS, 2024a):

- **Instalação do Pacote ESP32**: É necessário adicionar o pacote de suporte para o ESP32 no *Arduino IDE*. Isso é feito através do Gerenciador de Placas, que pode ser acessado nas Preferências do IDE. Adicionando a URL do gerenciador de pacotes, o pacote ESP32 será instalado, permitindo o reconhecimento da placa pelo IDE.
- **Seleção da Placa e Porta Serial**: Após a instalação, o usuário deve selecionar a placa ESP32 correta e a porta serial à qual o microcontrolador está conectado. Esta configuração é crucial para garantir que o IDE possa se comunicar corretamente com o dispositivo .
- **Programação e Upload**: Com o ambiente configurado, o código pode ser escrito utilizando a linguagem de programação Arduino (baseada em C/C++). O IDE permite a compilação e o upload do código para o ESP32. A integração com bibliotecas específicas para o ESP32 facilita a implementação de funcionalidades avançadas, como comunicação sem fio.
- **Monitoramento Serial**: O monitor serial do IDE é utilizado para visualizar dados enviados pelo ESP32 e para realizar a depuração em tempo real. Este recurso é essencial para o ajuste e a verificação do funcionamento do código.

O uso do *Arduino IDE* para programar o ESP32 oferece várias vantagens. O IDE fornece um ambiente simplificado para a criação e depuração de código, o que é vantajoso tanto para iniciantes quanto para desenvolvedores experientes. A integração com bibliotecas e exemplos específicos acelera o processo de desenvolvimento e reduz a complexidade de projetos.

O *Arduino IDE* também facilita a exploração das capacidades do ESP32, permitindo a implementação de soluções inovadoras em áreas como automação, Internet das Coisas (IoT) e sistemas embarcados. O suporte da comunidade e a disponibilidade de recursos educacionais adicionais contribuem para a evolução contínua de projetos baseados no ESP32 (ARDUINO, 2024).

3.7 Desvio Absoluto da Mediana (MAD)

O Desvio Absoluto da Mediana (MAD) é uma medida estatística robusta de dispersão, ideal para identificar *outliers* e avaliar a variabilidade em conjuntos de dados sujeitos a ruídos ou valores extremos. Diferente do desvio padrão, o MAD é menos influenciado por anomalias, sendo, portanto, mais adequado para sistemas de medição como o desenvolvido neste trabalho.

Segundo (ROUSSEEUW; LEROY, 2003), o MAD é definido pela equação 3.8:

$$\text{MAD} = \text{mediana}(|x_i - \tilde{x}|) \tag{3.9}$$

onde x_i representa os valores da amostra e $\tilde{x}$ é a mediana do conjunto de dados. Para permitir a comparação com o desvio padrão em distribuições normais, aplica-se um fator de correção:

$$\hat{\sigma} = 1.4826 \cdot \text{MAD} \tag{3.10}$$

Essa estimativa robusta de escala é amplamente utilizada em contextos onde a presença de ruído e *outliers* compromete a eficácia de métodos tradicionais baseados em média e desvio padrão.

Conforme destacado por (ROMO-CHAVERO *et al.*, 2024), o MAD oferece detecção de anomalias sem ser distorcido por valores extremos. Além disso, por ser uma medida não paramétrica, não pressupõe distribuição normal dos dados, ampliando sua aplicabilidade prática.

No contexto deste trabalho, o uso do MAD contribui diretamente para a filtragem de valores atípicos nas leituras de deformação obtidas pelos *strain gauges*. Essa filtragem é fundamental para garantir a integridade dos dados utilizados na análise do torque em eixos automotivos.

3.8 Regressão Linear pelos Mínimos Quadrados Ordinários (OLS)

A regressão linear é uma técnica estatística amplamente utilizada para modelar a relação entre uma variável dependente y e uma variável independente x, assumindo que essa relação é linear. No modelo de regressão linear simples, essa relação é expressa pela equação:

$$y = ax + b + \varepsilon \tag{3.11}$$

onde a é o coeficiente angular, b o intercepto, e ε o termo de erro ou ruído residual (MONTGOMERY; PECK; VINING, 2012; DRAPER; SMITH, 1998).

O método dos mínimos quadrados ordinários (*Ordinary Least Squares – OLS*) é o procedimento padrão para estimar os coeficientes a e b, encontrando os valores que minimizam a soma dos quadrados das diferenças entre os valores observados y_i e os valores ajustados pelo modelo $\hat{y}_i = ax_i + b$, formalmente definido pela equação 3.12:

$$\min_{a,b} \sum_{i=1}^{n} (y_i - (ax_i + b))^2 \tag{3.12}$$

Essa minimização garante o melhor ajuste linear dos dados observados segundo o critério da menor soma dos resíduos quadráticos (MONTGOMERY; PECK; VINING, 2012).

A solução analítica do problema é obtida pelas chamadas equações normais, que fornecem os valores dos coeficientes por meio das equações 3.13 e 3.14:

$$a = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{n\sum x_i^2 - (\sum x_i)^2} \tag{3.13}$$

$$b = \bar{y} - a\bar{x} \tag{3.14}$$

onde n é o número total de observações, e $\bar{x}$ e $\bar{y}$ são as médias dos conjuntos de dados x e y, respectivamente (DRAPER; SMITH, 1998; WEISBERG, 2005).

Neste trabalho, a regressão linear OLS foi empregada para calibrar o sensor de torque, relacionando as leituras obtidas do sensor às cargas reais aplicadas durante o processo de calibração. Assim, o modelo possibilita converter os valores digitais do sensor em grandezas físicas de peso, fundamental para a precisão nas medições de torque.

3.9 Intervalo Interquartílico (IQR) e Detecção de *Outliers*

A análise de dados experimentais pode ser fortemente impactada por valores extremos, que comprometem a representação real do fenômeno estudado. Nesse contexto, a Análise Exploratória de Dados, proposta por Tukey *et al.* (1977), fornece ferramentas robustas para compreender o comportamento dos dados antes da aplicação de modelos estatísticos mais elaborados.

Dentre essas ferramentas, destaca-se o uso do *boxplot*, um gráfico que resume a distribuição de um conjunto de dados por meio de seus quartis. A partir desse conceito, Tukey *et al.* (1977) introduziu o Intervalo Interquartílico (IQR), uma medida robusta de dispersão que representa a diferença entre o terceiro quartil ($Q3$) e o primeiro quartil ($Q1$):

$$\text{IQR} = Q3 - Q1 \tag{3.15}$$

Com base nessa medida, é possível identificar *outliers* (valores atípicos) que se situam fora dos seguintes limites:

$$\text{Limite inferior} = Q1 - 1.5 \times \text{IQR} \tag{3.16}$$

$$\text{Limite superior} = Q3 + 1.5 \times \text{IQR} \tag{3.17}$$

Observações que ultrapassam esses limites são consideradas potenciais *outliers*. Esse método é amplamente adotado por sua simplicidade, especialmente em dados assimétricos ou com contaminação por ruídos.

Na prática, esse critério é frequentemente aplicado por meio de bibliotecas de linguagem Python, como `NumPy` e `pandas`, em rotinas de pré-processamento de dados experimentais, com o objetivo de remover valores espúrios antes de análises mais sofisticadas, como regressão ou modelagem.

A Figura 8 apresenta um exemplo ilustrativo de um *boxplot* com os limites baseados no IQR, evidenciando os *outliers*.

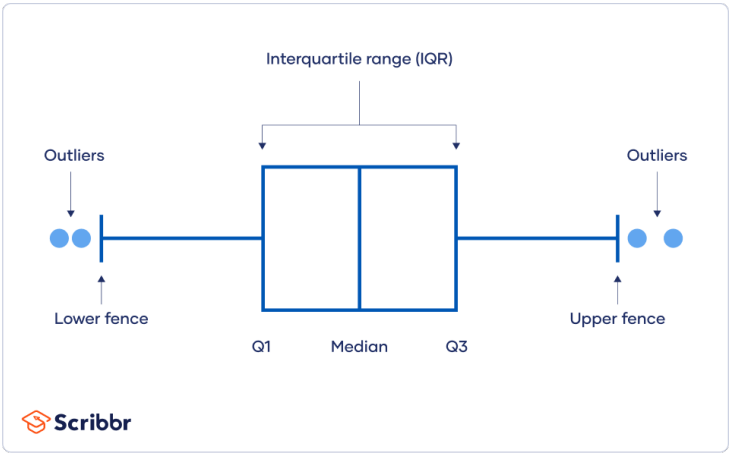

Figura 8 – Boxplot ilustrando o intervalo interquartílico (IQR) e os limites para detecção de outliers segundo Tukey *et al.* (1977). Fonte:(TAYLOR, 2023)

Esse critério foi empregado neste trabalho como parte do tratamento de sinais provenientes de sensores, com o intuito de eliminar picos isolados ou anomalias geradas por ruídos de transmissão ou falhas momentâneas de leitura.

3.10 Erro Absoluto Médio (MAE)

O Erro Absoluto Médio (MAE,*Mean Absolute Error*) constitui uma métrica amplamente empregada na avaliação da acurácia de sistemas de medição e modelos preditivos. Essa métrica

quantifica a média das diferenças absolutas entre os valores observados e os valores estimados, conforme expressa a Equação 3.18:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{3.18}$$

Segundo Chai e Draxler (2014), o MAE apresenta vantagens em comparação a outras métricas de erro, como a raiz do erro quadrático médio (RMSE). Entre esses aspectos, destaca-se a menor sensibilidade a valores extremos, o que torna o MAE mais robusto em contextos nos quais outliers podem influenciar de forma desproporcional a análise estatística. Além disso, sua interpretação é direta, uma vez que expressa o erro médio absoluto nas mesmas unidades do fenômeno avaliado no presente trabalho, em *newton* metro (N.m), unidade de medida de torque.

A adoção do MAE neste estudo justifica-se pela necessidade de comparar, de forma objetiva e transparente, o desempenho de um sistema de aquisição de torque sem fio em relação a um sensor de referência com fio. Dessa forma, o MAE permite quantificar, com precisão, o erro médio absoluto entre os dois sinais, contribuindo para a avaliação da confiabilidade do sistema proposto.

3.11 Correlação de Pearson

Para avaliar a precisão dos dados coletados pelos sensores de torque, tanto o sensor com fio quanto o sem fio desenvolvido neste trabalho, utilizou-se o coeficiente de correlação de Pearson. Este coeficiente é uma medida estatística que quantifica o grau de associação linear entre duas variáveis quantitativas, indicando a força e a direção dessa relação (DANCEY; REIDY, 2013).

Matematicamente, o coeficiente de correlação de Pearson, denotado por r, é calculado pela pela equação 3.19:

$$r = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \bar{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \bar{Y})^2}}, \tag{3.19}$$

onde X_i e Y_i são os valores das duas variáveis amostradas, $\bar{X}$ e $\bar{Y}$ são as suas médias, e n é o número de observações.

O valor de r varia entre -1 e 1. Valores próximos a 1 indicam uma forte correlação positiva, ou seja, quando uma variável aumenta, a outra também tende a aumentar. Valores próximos a -1 indicam uma forte correlação negativa, enquanto valores próximos a zero indicam pouca ou nenhuma correlação linear (DANCEY; REIDY, 2013).

No contexto deste trabalho, a correlação de Pearson foi utilizada para comparar os sinais de torque obtidos pelo sensor sem fio desenvolvido com o sinal de referência coletado pelo sensor com fio (HBM). Dessa forma, foi possível avaliar a precisão e a confiabilidade do sistema sem

fio ao quantificar o grau de similaridade entre os dados obtidos pelas duas metodologias de aquisição.

4 METODOLOGIA

4.1 Instrumentação do Eixo com *Strain Gauges*

Inicialmente, foi realizado a instrumentação do eixo com *strain gauges* para medir o torque. Esse processo compreende as seguintes etapas:

- **Preparação do eixo**: Inicialmente, a superfície foi lixada manualmente com lixa específica para aço, removendo imperfeições, oxidação e irregularidades que pudessem comprometer o contato com o adesivo. Em seguida, procedeu-se à limpeza com álcool para eliminar resíduos de pó e oleosidade.
- **Posicionamento dos *strain gauges***: Os sensores foram posicionados estrategicamente nos locais de maior concentração de esforços, seguindo recomendações teóricas e o projeto experimental. A orientação e o alinhamento dos *strain gauges* foram cuidadosamente realizados para garantir medições acuradas.
- **Fixação dos *strain gauges***: Foram utilizados adesivo à base de cianoacrilato(*Super Bonder* apropriados para a fixação permanente dos sensores, assegurando a estabilidade durante o processo de medição e evitando deslocamentos ou falhas de aderência.

4.2 Conexão ao Amplificador *HX711*

Após a instrumentação do eixo, os *strain gauges* foram conectados ao amplificador *HX711*, que tem a função de amplificar o sinal analógico gerado pelos sensores. O procedimento de conexão foi feito conforme descrito abaixo:

- **Conexão elétrica**: Os terminais de saída dos *strain gauges* foram conectados aos pinos de entrada do *HX711*. Essa etapa seguiu o esquema de ligação recomendado pelo fabricante.
- **Configuração do amplificador**: Foi utilizada a biblioteca *HX711* disponível na IDE Arduino utilizada para facilitar a comunicação entre o *HX711* e o microcontrolador *ESP32*. A configuração da interface *I2C* foi essencial para a correta comunicação entre os dispositivos.

4.3 Integração com o *ESP32*

O *ESP32* foi utilizado como microcontrolador para processar os dados provenientes do amplificador *HX711* e enviar esses dados ao sistema de análise. A integração entre o *ESP32* e os demais componentes foi feita da seguinte maneira:

- **Conexão física do *ESP32***: O amplificador *HX711* foi fisicamente conectado ao *ESP32*, utilizando os pinos de entrada e saída adequados para comunicação.

- **Programação do *ESP32***: O *software* Arduino IDE, em conjunto com o pacote *Espressif Systems*, foi utilizada para programar o *ESP32* em C++. Um código específico foi desenvolvido para receber os dados amplificados do *HX711* e preparar sua transmissão sem fio.

- **Configuração *wireless***: O módulo *wireless* do *ESP32* foi configurado para possibilitar a transmissão dos dados coletados para o computador de análise.

4.4 Alimentação do Sistema com Bateria e Regulador MT3608

Para garantir a autonomia e a mobilidade do sistema de aquisição de torque sem fio, o ESP32 foi alimentado por uma bateria recarregável de íons de lítio. Devido à variação da tensão da bateria durante o uso, foi utilizado o módulo conversor elevador de tensão MT3608 para estabilizar a alimentação.

O MT3608 conforme ilustrado pela Figura 9é um regulador DC-DC *step-up* que converte a tensão variável da bateria para um valor fixo e estável adequado ao funcionamento do ESP32, geralmente 5 V ou 3,3 V conforme a necessidade do projeto. Essa configuração assegura que o microcontrolador e os demais componentes eletrônicos operem dentro das especificações recomendadas, prevenindo falhas e oscilações no sistema.

As etapas para a implementação da alimentação foram:

- **Seleção da bateria:** Foi escolhida uma bateria de íons de lítio com capacidade adequada para suportar o tempo de operação desejado do sistema, considerando a corrente consumida pelo ESP32 e demais componentes.

- **Conexão do MT3608:** O módulo foi conectado à bateria e ao ESP32, ajustando a saída do MT3608 para a tensão ideal de operação.

Figura 9 – MT3608 DC To DC Step Up Converter Module. Fonte: (Electrothinks, 2023).

Essa abordagem de alimentação por bateria com regulagem via MT3608 foi fundamental para garantir a portabilidade do sistema de aquisição de torque, tornando-o viável para aplicações automotivas onde a fonte de energia fixa não está disponível.

4.5 Projeto e Fabricação da Caixa Protetora

A caixa protetora conforme ilustrado pela Figura 10 para os componentes eletrônicos foi projetada utilizando o *software* NX, ferramenta que permitiu a criação de um modelo tridimensional preciso, facilitando o desenvolvimento de uma estrutura adequada para o sistema. As dimensões externas da caixa são 36 mm de largura, 29 mm de altura e 136 mm de comprimento, dimensões estas definidas para acomodar confortavelmente os módulos ESP32, HX711 e MT3608, além de possibilitar um manuseio prático e seguro.

Figura 10 – Caixa protetora impressa em 3D para acomodação dos componentes eletrônicos do sistema. Foto de autoria própria.

O projeto da caixa priorizou a fixação segura dos componentes, garantindo que todos os módulos permanecessem firmemente posicionados durante o uso, minimizando vibrações e deslocamentos que poderiam comprometer a qualidade das medições e a integridade do sistema. Essa fixação foi projetada de forma a facilitar a montagem e manutenção dos equipamentos, além de proteger os circuitos contra possíveis impactos e interferências externas.

A fabricação da caixa foi realizada por meio de impressão 3D, o que possibilitou a rápida prototipagem e ajustes no design conforme as necessidades do projeto desenvolvido.

4.6 Coleta e Transmissão dos Dados

Após a montagem do sistema de aquisição baseado em *strain gauges*, iniciou-se a etapa de coleta e transmissão dos dados. Essa etapa antecede a calibração, pois é por meio dela que se obtêm os valores brutos a serem utilizados na construção do modelo de conversão de sinais.

- **Aquisição dos Dados via HX711:**O módulo HX711 é responsável por amplificar e converter os sinais analógicos gerados pelos *strain gauges* em sinais digitais com alta resolução. O ESP32 lê os valores do HX711 apenas quando o módulo está pronto, garantindo a integridade das amostras. Caso contrário, a leitura é adiada.
- **Transmissão Cliente-Servidor com ESP32:**O ESP32 atua como servidor TCP(*Transmission Control Protocol*) e transmite os dados adquiridos para um cliente executado em um computador local. Os dados são enviados em pacotes contendo o valor lido e o tempo da leitura em milissegundos desde o início da execução. Essa estrutura garante sincronização e rastreabilidade dos dados.

O funcionamento básico do código embarcado responsável por essa etapa pode ser visualizado no ApêndiceC.

4.7 Calibração do *Strain Gauge*

A calibração do sistema de medição é uma etapa essencial para garantir que os valores obtidos representem fielmente o torque real aplicado ao eixo. O processo foi conduzido conforme os seguintes passos:

- **Aplicação de cargas padrão:** Cargas conhecidas foram aplicadas ao eixo, gerando deformações específicas detectadas pelos *strain gauges*. Esses valores físicos serviram como referência para o modelo de calibração.
- **Coleta de dados do sensor:** Para cada carga aplicada, foram coletadas amostras do sinal digital convertido pelo HX711, conforme descrito anteriormente na seção 4.6. As leituras representam valores brutos que variam conforme a deformação do eixo.
- **Filtragem dos dados:** Os dados coletados foram submetidos a um procedimento de filtragem utilizando o MAD, visando reduzir a influência de ruídos e outliers no conjunto de dados. Esse processo é fundamental para assegurar a qualidade das médias utilizadas no ajuste da calibração.
- **Ajuste por regressão linear:** Utilizou-se o método dos OLS para ajustar uma equação da forma $y = ax + b$, relacionando os valores médios filtrados provenientes do sensor

aos respectivos pesos aplicados. O objetivo é obter uma equação de calibração capaz de converter futuras leituras em valores físicos reais.

O código utilizado para implementar o processo de calibração descrito encontra-se no ApêndiceB ele demonstra a aplicação da regressão linear simples, bem como o código tratamento estatístico dos dados com o uso do MAD encontra-se no ApêndiceA.

Essa calibração foi fundamental para obter os coeficientes a e b, responsáveis por converter as leituras digitais dos sensores em valores reais de torque. A técnica aplicada está baseada na regressão linear pelos OLS, conforme descrito por autores como (MONTGOMERY; PECK; VINING, 2012), (DRAPER; SMITH, 1998) e (WEISBERG, 2005).

4.8 Aquisição e Exportação dos Dados

O processo de aquisição e exportação dos dados foi conduzido conforme descrito a seguir:

- **Aquisição dos dados:** O ESP32 realiza a leitura contínua dos sinais amplificados pelos strain gauges através do HX711. Os dados são coletados em intervalos regulares de aproximadamente 100 ms e transmitidos via conexão *Wi-Fi* para o computador cliente, garantindo que as informações sejam capturadas em tempo real.
- **Processamento dos dados no cliente:** O computador recebe os dados brutos enviados pelo ESP32, realiza a filtragem e o pré-processamento necessários para remover ruídos e valores atípicos, conforme a calibração realizada previamente.
- **Exportação para arquivo:** Após o processamento, os dados são armazenados em arquivos no formato `.txt`, contendo os valores do sensor e os respectivos tempos de aquisição. Essa exportação é feita de forma automatizada para facilitar a análise posterior e o arquivamento dos dados coletados.
- **Formato dos dados exportados:** Cada linha do arquivo contém um registro com o valor bruto do sensor e o tempo (em milissegundos desde o início da coleta), separados por vírgula, o que permite fácil manipulação em softwares de análise ou planilhas eletrônicas.

  O trecho de código responsável pela aquisição e exportação automatizada dos dados está apresentado no ApêndiceD.

4.9 Testes para Validação do Funcionamento do Sistema

Esta seção apresenta os testes realizados para avaliar e validar o funcionamento do sistema desenvolvido. As seções a seguir descrevem cada um dos procedimentos experimentais aplicados, detalhando os métodos utilizados para verificar a integridade, a estabilidade e a precisão

da comunicação e aquisição de dados entre o ESP32 e o computador em condições reais de operação.

4.9.1 Teste de Durabilidade da Bateria

Para verificar a autonomia do sistema em operação contínua, foi realizado um teste de durabilidade utilizando uma bateria recarregável de íons de lítio modelo KP18650, da marca Knup, com capacidade nominal de 3800 mAh. A bateria foi conectada a um conversor DC-DC do tipo *boost* (MT3608), regulado para fornecer uma tensão constante de 5 V, valor necessário para alimentar o microcontrolador ESP32.

Durante o ensaio, o ESP32 permaneceu operando continuamente com o módulo *Wi-Fi* ativo e o código de aquisição em execução. O objetivo foi simular uma condição real de uso e monitorar a durabilidade da alimentação fornecida pelo conjunto bateria + conversor. As medições da corrente e da tensão da bateria foram realizadas a cada 30 minutos, utilizando um multímetro digital. As leituras da tensão foram feitas diretamente nos terminais da bateria, enquanto a corrente foi medida em série com a alimentação do sistema.

As coletas foram realizadas até que a tensão da bateria caísse a níveis que comprometiam a operação do ESP32, permitindo determinar a autonomia prática do sistema sob as condições propostas.

4.9.2 Teste de Conectividade em Ambiente Ruidoso

Para avaliar a estabilidade da comunicação entre o ESP32 e o computador em um ambiente eletromagneticamente ruidoso, utilizou-se um sistema de aquisição de dados baseado em transmissão via rede *Wi-Fi*.

O ESP32 foi programado para enviar continuamente um valor de tempo interno (em milissegundos, via função `millis()`) a um computador conectado na mesma rede. Paralelamente, no computador, foi executado um *script* em *Python* responsável por detectar automaticamente o ESP32 na rede, estabelecer conexão via socket TCP, receber os dados enviados e registrá-los em arquivo texto para análise posterior.

Os códigos-fonte utilizados para a comunicação do ESP32 e para a coleta dos dados via Python estão anexados a este trabalho, nos ApêndiceE e ApêndiceF, respectivamente.

A montagem do sistema para o teste foi realizada de forma a simular uma condição real de operação automotiva, com foco na avaliação da comunicação em ambiente ruidoso.à caixa sensora por meio de um adaptador desenvolvido especificamente para essa finalidade foi acoplado mecanicamente ao motor trifásico modelo LVS 63-4 . Esse adaptador permitiu a conexão entre o eixo do motor e a estrutura impressa em 3D que abriga os módulos eletrônicos do sistema, incluindo o ESP32, o regulador MT3608, o HX711 e a célula de carga com extensômetro. O controle da rotação foi realizado por meio do inversor de frequência WEG CFW500, permitindo

variações precisas na velocidade do motor durante os testes. A Figura 11 apresenta uma visão geral da montagem do sistema utilizado nos experimentos.

Figura 11 – Sistema montado para o teste de conectividade, incluindo motor trifásico, inversor de frequência, adaptador e caixa 3D com sensor ESP32.

O teste de comunicação foi realizado em intervalos de 30 segundos, período durante o qual o ESP32 enviava continuamente seus dados temporais e o script Python registrava o instante de recebimento correspondente. Dessa forma, foi possível observar a qualidade da transmissão, latência e eventuais perdas de pacotes em condições reais de interferência eletromagnética, simuladas pela operação do motor trifásico em diferentes velocidades.

A latência em redes de comunicação refere-se ao tempo necessário para que um dado percorra o caminho entre o emissor e o receptor. Em sistemas sem fio, como no caso da transmissão via *Wi-Fi* entre o ESP32 e o computador, a latência envolve diversas etapas, incluindo o tempo de processamento interno do microcontrolador, o tempo de propagação do sinal pela rede e o tempo de resposta do sistema receptor. Uma latência baixa é desejável em aplicações que exigem respostas rápidas ou sincronização precisa, como no monitoramento em tempo real de variáveis físicas, como o torque automotivo.

No contexto dos testes realizados neste trabalho, a latência representa o intervalo entre o momento em que o ESP32 envia um valor de tempo (via função `millis()`) e o instante em que

esse dado é efetivamente recebido e registrado pelo computador. Esse valor pode ser afetado por fatores como a taxa de envio configurada, a carga de tráfego na rede *Wi-Fi* e, especialmente, a presença de interferência eletromagnética causada pelo motor trifásico em operação. Portanto, ao analisar a latência nas diferentes combinações de taxa de envio e velocidade do motor, é possível avaliar a robustez da comunicação e sua adequação a ambientes ruidosos típicos da aplicação automotiva.

Durante o teste prático de conectividade, a taxa de envio dos dados pelo ESP32 foi ajustada manualmente no código, utilizando intervalos de 100 ms, 10 ms e 2 ms para o envio dos sinais temporais. Para cada configuração de frequência, o motor elétrico foi acionado em diferentes velocidades controladas pelo inversor de frequência WEG CFW500.

As frequências de operação do motor testadas foram 20 Hz, 25 Hz, 30 Hz e 35 Hz, correspondendo, respectivamente, às velocidades reais medidas com o tacômetro digital de 598 rpm, 748 rpm, 898 rpm e 1048 rpm Conforme apresentado na Tabela 1.

Assim, o teste compreendeu a combinação de três taxa de transmissão do ESP32 com quatro velocidades distintas do motor, totalizando doze condições diferentes. Em cada condição, a comunicação entre o ESP32 e o computador foi mantida por um intervalo de 30 segundos, permitindo avaliar a estabilidade do sistema sob diferentes níveis de interferência eletromagnética e taxas de envio de dados.

Tabela 1 – Condições de Teste de Comunicação entre ESP32 e Computador

Tempo de Amostragem do ESP32 (ms)	**Frequência do Motor (Hz)**	**Velocidade Real do Motor (rpm)**
100	20	598
100	25	748
100	30	898
100	35	1048
10	20	598
10	25	748
10	30	898
10	35	1048
2	20	598
2	25	748
2	30	898
2	35	1048

4.9.3 Teste prático de comparação entre sensor sem fio e sistema com fio

O teste prático teve como objetivo validar o desempenho do sensor desenvolvido, comparando suas medições com aquelas obtidas por um sistema de aquisição de dados tradicional com fio, utilizando o equipamento *QuantumX*, da HBM, como referência.

Para a realização do experimento, o eixo foi cuidadosamente instrumentado com quatro sensores do tipo *strain gauge*, posicionados de maneira equidistante ao longo da circunferência da seção transversal, com espaçamento angular de 90°. Essa configuração visa capturar, de forma simétrica, as deformações geradas pelo torque aplicado ao eixo durante seu funcionamento.

Os sensores foram organizados em dois pares diametralmente opostos (separados por 180°), formando duas pontes completas de *Wheatstone*. Essa topologia de instrumentação é comumente utilizada para maximizar a sensibilidade à deformação por torção, ao mesmo tempo em que minimiza a influência de esforços axiais e de flexão. Cada ponte foi montada garantindo o equilíbrio do circuito e a linearidade da resposta elétrica frente à variação de deformação mecânica.

A Figura 12 ilustra a montagem dos *strain gauges* sobre o eixo, bem como a integração do protótipo do sensor sem fio, responsável por realizar a leitura e a transmissão dos sinais de deformação.

Figura 12 – Eixo instrumentado com quatro *strain gauges* dispostos a 90° e protótipo do sensor sem fio acoplado.

O método com fio utiliza um *strain gauge* instalado no eixo rotativo, cujos sinais elétricos são transmitidos da parte rotativa para a parte estática por meio de um *slip ring* (anel deslizante), conforme ilustrado na Figura 13. Esse dispositivo permite a passagem contínua do sinal elétrico sem interferir na rotação do eixo. O fio proveniente do *strain gauge* é conectado ao sistema de aquisição *QuantumX* da HBM, que amplifica e converte os sinais analógicos em digitais com

alta resolução. O *QuantumX* é integrado ao computador por meio do *software* Catman da HBM, possibilitando a visualização, armazenamento e análise em tempo real dos dados adquiridos.

Figura 13 – *Slip ring* instalado na roda do veículo, responsável por transmitir o sinal do *strain gauge* rotativo para o sistema com fio.

Com ambos os sistemas devidamente calibrados, foram realizadas as aquisições simultâneas dos dados de torque durante a aplicação de cargas variáveis no eixo. Essa etapa permitiu a coleta de sinais para posterior análise comparativa entre as medições do sensor sem fio e do sistema com fio.

5 RESULTADOS

Foram realizados testes experimentais com o sistema proposto com o objetivo de avaliar seu funcionamento sob diferentes condições de operação. Cada teste foi estruturado para analisar um aspecto específico do sistema, permitindo a verificação de seu desempenho em situações controladas.

Inicialmente, foi conduzido um teste para verificar a durabilidade da bateria utilizada na alimentação do dispositivo. Esse ensaio visou determinar o tempo máximo de operação contínua com base no consumo energético real do sistema em funcionamento.

Em seguida, avaliou-se a estabilidade da comunicação *Wi-Fi* em um ambiente com potencial de interferência eletromagnética. Para isso, o sistema foi exposto a diferentes níveis de ruído eletromagnético, induzidos pela variação da frequência de um motor trifásico operando com inversor de frequência. Foram utilizadas diferentes taxas de envio de dados para verificar a ocorrência de perdas de pacotes e a manutenção da conexão em situações adversas.

Posteriormente, foi efetuado um teste em um veículo automotivo, no qual os dados obtidos pelo sistema foram comparados com os de um equipamento de referência, utilizado como base para análise de compatibilidade entre as leituras.

Por fim, foi realizada uma análise de custo do sistema desenvolvido, considerando os componentes utilizados e os processos de montagem, com o intuito de estimar sua viabilidade econômica em comparação com soluções comerciais destinadas à mesma finalidade.

Os resultados obtidos nos testes citados são apresentados a seguir, com base em registros experimentais, medições quantitativas e observações realizadas durante as atividades.

5.1 Teste de Durabilidade da Bateria

Este teste teve como objetivo avaliar o desempenho de uma bateria de íons de lítio modelo 18650 Knup, com capacidade nominal de 3800 mAh e tensão nominal de 3,7 V, alimentando um módulo ESP32 por meio do conversor boost MT3608 regulado para 5 V na saída. O ESP32 foi alimentado pelo pino VIN (5 V), com o *Wi-Fi* ativo durante todo o teste, simulando um cenário de consumo elevado.

Os resultados experimentais conforme apresentado na Tabela 2 indicam que o sistema mantém a comunicação *Wi-Fi* estável enquanto a tensão da bateria permanece acima de aproximadamente 3,3 V. Nessa faixa, a corrente consumida pela bateria aumenta gradativamente, refletindo a tentativa do conversor boost de compensar a queda de tensão, mantendo uma saída regulada de 5 V para o módulo ESP32.

A partir do instante em que a tensão da bateria atinge cerca de 3,3 V, observa-se uma tendência de queda no desempenho do sistema, caracterizada por oscilações e perda progressiva da estabilidade da conexão *Wi-Fi*. Esta condição está associada ao limite prático da bateria em

Tabela 2 – Tensão, corrente e potência consumida da bateria ao longo do tempo

Tempo (min)	Tensão da Bateria (V)	Corrente da Bateria (mA)	Potência Consumida (mW)
0	4.20	140	588
30	4.10	145	595
60	4.00	150	600
90	3.90	155	605
120	3.80	160	608
150	3.70	165	611
180	3.60	170	612
210	3.50	180	630
240	3.40	190	646
270	3.30	200	660

fornecer corrente suficiente para a demanda do conversor, que aumenta sua corrente de entrada conforme a tensão da bateria diminui.

O teste durou aproximadamente 270 minutos (4 horas e 30 minutos) até que a tensão da bateria atingisse o limiar de 3,3 V, tempo este em que o sistema ainda operava com comunicação estável. Após este ponto, espera-se uma rápida degradação da performance do sistema, culminando na perda completa da conectividade.

Esta análise qualitativa evidencia que a autonomia prática da bateria sob as condições de teste com o ESP32 operando com *Wi-Fi* ativo e alimentação via MT3608 em 5 V é limitada a aproximadamente 4,5 horas. Tal resultado é fundamental para o dimensionamento correto da fonte de alimentação em projetos que exigem alta confiabilidade na comunicação sem fio.

Apesar da bateria possuir capacidade nominal de 3800 mAh, os dados obtidos indicam que essa carga total não foi efetivamente entregue durante o teste. Estimando uma corrente média de aproximadamente 165 mA ao longo dos 270 minutos de operação, a carga fornecida pela bateria foi de cerca de 742,5 mAh (165 mA × 4,5 h). Esse valor representa apenas uma fração da capacidade anunciada, evidenciando uma discrepância entre o valor nominal e o desempenho real observado.

Tal diferença pode estar relacionada à baixa qualidade da célula utilizada, à limitação de corrente sob tensões mais baixas, ou ainda a especificações comerciais inflacionadas — algo comum em baterias de fabricantes genéricos. Este resultado ressalta a importância de testes práticos para validação de componentes críticos, como a fonte de alimentação, especialmente em aplicações que exigem autonomia e estabilidade energética.

Adicionalmente, os dados ressaltam a importância de considerar não apenas a capacidade nominal da bateria, mas também a capacidade real de fornecimento de corrente estável em tensões baixas, especialmente quando se utiliza conversores boost que elevam a tensão para níveis fixos. A instabilidade gerada pela queda da tensão da bateria impacta diretamente o funcionamento do regulador interno do ESP32 e, consequentemente, do módulo *Wi-Fi*, que é o subsistema mais

sensível a variações na alimentação.

Em síntese, o presente teste contribui para a compreensão dos limites práticos de operação do sistema, fornecendo dados relevantes para a otimização da autonomia e da confiabilidade em aplicações embarcadas com comunicação *wireless*.

5.2 Teste de Conectividade em Ambiente Ruidoso

Este teste teve como objetivo avaliar a estabilidade da comunicação via *Wi-Fi* entre o ESP32 e o computador em um ambiente eletromagneticamente ruidoso, simulado pela operação de um motor trifásico em diferentes frequências.

Foram realizados ensaios para três taxas de envio dos dados pelo ESP32 (100 ms, 10 ms e 2 ms) combinadas a quatro velocidades distintas do motor (20 Hz, 25 Hz, 30 Hz e 35 Hz), totalizando 12 condições diferentes conforme Tabela 3.

Para cada condição, o sistema foi mantido em operação por aproximadamente 30 segundos, durante os quais foram registrados os pacotes recebidos, calculadas as perdas de pacotes e analisadas as latências da transmissão.

A Tabela 3 apresenta os principais indicadores de desempenho obtidos durante os testes de conectividade em ambiente ruidoso. Para cada combinação de taxa de envio de dados do ESP32 (100 ms, 10 ms e 2 ms) com as diferentes frequências de operação do motor trifásico (20 Hz, 25 Hz, 30 Hz e 35 Hz), foram registrados o número total de pacotes recebidos, a quantidade esperada de pacotes em um intervalo de 30 segundos, a perda absoluta e percentual de pacotes, além dos valores de latência média e latência máxima observados em cada condição.

Esses parâmetros permitem avaliar de forma abrangente não apenas a robustez e a estabilidade da conexão *Wi-Fi* em ambientes sujeitos a interferências eletromagnéticas, mas também o impacto que diferentes configurações de *hardware* e *software* exercem sobre a qualidade do sinal e o tempo de resposta da transmissão de dados. A análise detalhada dessas variáveis é fundamental para compreender como variações no ambiente e nas condições operacionais influenciam o desempenho do sistema de comunicação sem fio, especialmente em aplicações críticas que demandam alta confiabilidade e baixa latência.

Tabela 3 – Indicadores de desempenho da comunicação entre ESP32 e computador.

Taxa de envio ESP32 (ms)	Frequência Motor (Hz)	Pacotes Recebidos	Pacotes Esperados	Perda Absoluta	Perda (%)	Latência Média (ms)	Latência Máx. (ms)
100	35	299	300	1	0,33	122,25	511
100	30	298	300	2	0,67	119,90	521
100	25	298	300	2	0,67	263,32	2310
100	20	298	300	2	0,67	138,98	576
10	35	2817	3000	183	6,10	325,05	1963
10	30	2717	3000	283	9,43	500,83	1743
10	25	2717	3000	283	9,43	444,52	992
10	20	2734	3000	266	8,87	437,44	2246
2	35	10044	15000	4956	33,04	140,46	604
2	30	10000	15000	5000	33,33	154,30	621
2	30	9963	15000	5037	33,58	167,27	640
2	20	10005	15000	4995	33,30	143,94	437

5.2.1 Análise dos Resultados

- **Perda de Pacotes:**

A Figura 14 apresenta a perda percentual de pacotes registrada para cada frequência de operação do motor trifásico, considerando as diferentes taxas de envio do ESP32 (100 ms, 10 ms e 2 ms).

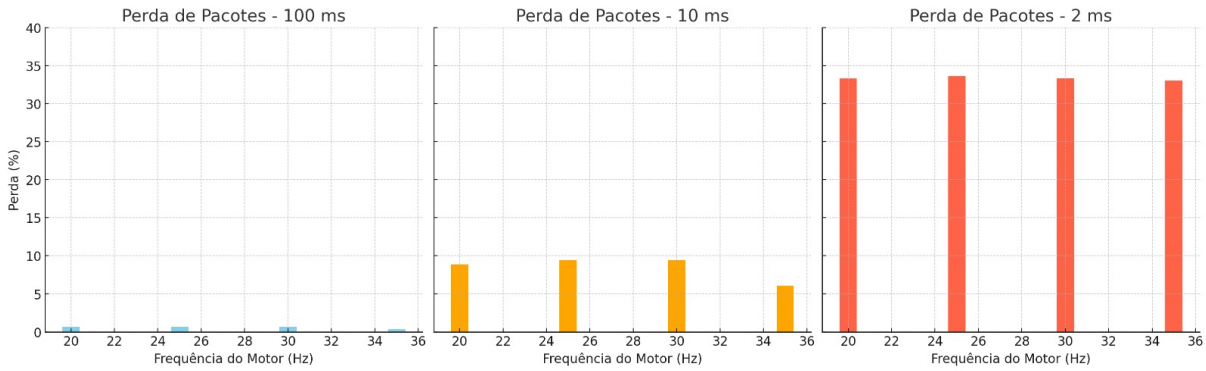

Figura 14 – Gráfico de barras demonstrando a perdas de pacotes.

Pode-se observar que, para a taxa de envio de 100 ms, a comunicação manteve estabilidade, com perdas inferiores a 1% em todas as condições. Para 10 ms, a perda de pacotes aumentou consideravelmente, situando-se entre 6% e 9%. Já a taxa de 2 ms apresentou perdas elevadas, acima de 33%, evidenciando a limitação da rede *Wi-Fi* em sustentar comunicações de taxa de envio sob interferência eletromagnética intensa.

- **Latência Média:**

A Figura 15 exibe os valores de latência média observados durante os testes, para cada configuração de frequência do motor e taxa de envio do ESP32.

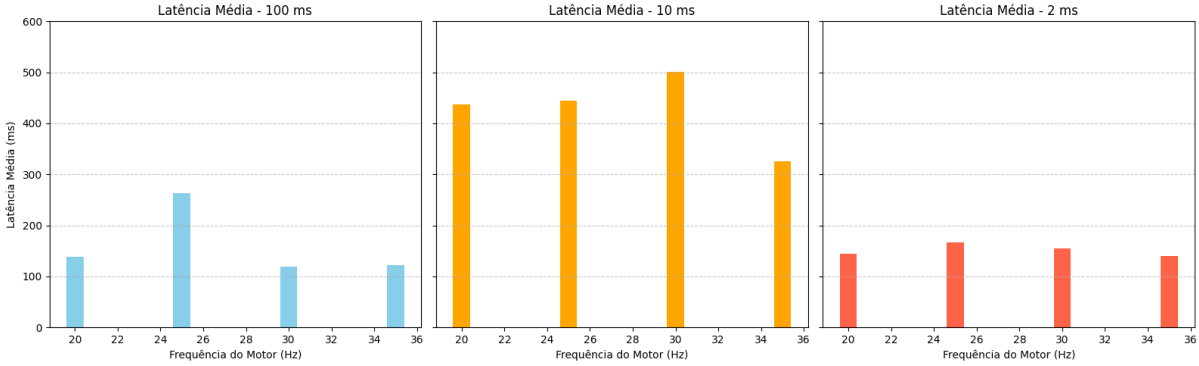

Figura 15 – Gráfico de barras demonstrando a latência média.

Nota-se que, para o envio a cada 100 ms, a latência média permaneceu inferior a 270 ms, com variação relativamente estável. Para 10 ms, os valores médios ultrapassaram 400 ms em quase todas as configurações, demonstrando atraso considerável. No caso de 2 ms, apesar das elevadas perdas, a latência média se manteve em patamares mais baixos (140 ms a 170 ms), sugerindo que os dados que conseguiram ser transmitidos chegaram com menor atraso, porém em menor quantidade.

- **Latência Máxima:**

A Figura 16 ilustra as latências máximas registradas durante os testes, refletindo os picos de atraso que o sistema pode apresentar.

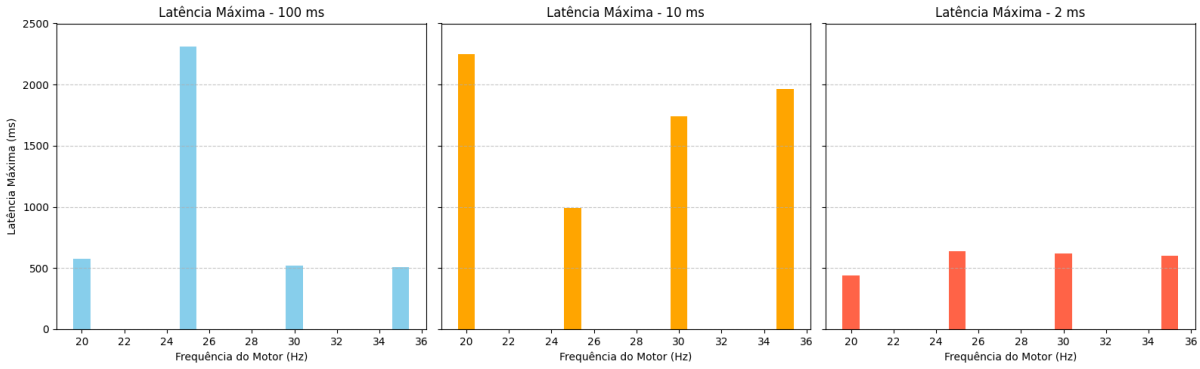

Figura 16 – Gráfico de barras demonstrando a latência máxima.

Os resultados revelam que a taxa de envio de 100 ms apresentou latências máximas pontuais elevadas (até 2310 ms), provavelmente causadas por eventos específicos de interferência.

Para 10 ms, os picos foram frequentes e superiores a 1700 ms em algumas condições. Já a taxa de 2 ms manteve a latência máxima abaixo de 650 ms, embora com elevada perda de pacotes. Esses valores indicam que o sistema pode apresentar atrasos imprevisíveis, especialmente em configurações com envio mais frequente ou sob forte ruído eletromagnético.

5.2.2 Considerações finais do teste

O sistema demonstrou bom desempenho para taxa de amostragem e transmissão superiores a 10 ms, com perdas de pacotes reduzidas e latências aceitáveis para aplicações automotivas que não demandem alta taxa de atualização.

Para aplicações que exijam maior taxa de amostragem, será necessário avaliar estratégias para mitigar as perdas, como melhoria na robustez da comunicação, compressão de dados ou protocolos alternativos.

5.3 Teste prático de comparação entre sensor sem fio e sistema com fio

Foram realizados três testes práticos com o objetivo de validar o desempenho do sensor de torque sem fio desenvolvido, comparando suas medições com as obtidas por um sistema tradicional de aquisição com fio. Os testes ocorreram no dia 17 de julho de 2025, imediatamente após a instrumentação do veículo, garantindo que as condições experimentais refletissem o cenário real de operação.

O modelo utilizado foi um Jeep Renegade equipado com sistema de tração 4x4 inteligente, característico das versões *off road* da linha. O Jeep Renegade é um SUV compacto que combina a dirigibilidade urbana com capacidade para enfrentar terrenos variados, graças ao seu sistema de tração nas quatro rodas (4x4).

O sistema de tração do Renegade, conhecido como *Jeep Active Drive*, opera predominantemente com tração nas rodas dianteiras para maximizar a eficiência de combustível em condições normais de uso. Quando sensores identificam perda de aderência nas rodas dianteiras, o sistema ativa automaticamente a tração nas rodas traseiras, distribuindo o torque de forma variável para o eixo que mais necessita, podendo chegar a 100% do torque em um eixo específico conforme a demanda. Além disso, o veículo conta com diferentes modos de condução, como *Auto*, *Snow* (neve), *Sand* (areia) e *Mud* (lama), que ajustam a resposta do sistema para otimizar a tração e a estabilidade em variados tipos de terreno. Para os testes realizados, o modo utilizado foi o modo *Auto*.

Para o experimento, o eixo traseiro esquerdo do Jeep Renegade foi cuidadosamente instrumentado. A instrumentação compreendeu a instalação de quatro sensores do tipo *strain gauge*, posicionados de forma equidistante ao longo da circunferência da seção transversal do eixo, com espaçamento angular de 90 graus. Essa configuração simétrica, distribuída em dois pares diametralmente opostos, formou duas pontes completas de *Wheatstone*, método amplamente

utilizado para maximizar a sensibilidade à deformação por torção e minimizar a influência de esforços axiais e de flexão. A montagem assegurou o equilíbrio do circuito e a linearidade da resposta elétrica frente às variações mecânicas.

O sistema tradicional de aquisição de dados utilizou um *strain gauge* instalado no eixo rotativo, conectado a um *slip ring*, um anel deslizante que permite a transmissão contínua do sinal elétrico da parte rotativa para a parte estática sem interferir na rotação do eixo. O sinal foi amplificado e convertido em digital pelo equipamento *QuantumX* da HBM, que permaneceu instalado dentro do veículo e conectado a um computador portátil. Esse computador executava o *software catman*, responsável pela aquisição, visualização e armazenamento dos dados em tempo real durante o teste.

Para realizar a conexão do sensor sem fio (ESP32) com o computador, foi utilizado um telefone celular atuando como roteador *Wi-Fi*. Durante os testes, o celular foi posicionado no banco traseiro, próximo ao eixo instrumentado, para garantir estabilidade da conexão. Experimentos preliminares demonstraram que, quando o celular estava localizado no banco dianteiro, ocorriam perdas frequentes de conexão, resultando na desconexão do ESP32 do computador e na interrupção do fluxo de dados.

O sensor sem fio desenvolvido realizou a leitura e transmissão dos sinais diretamente no eixo instrumentado, eliminando a necessidade do anel deslizante e de cabos rotativos, oferecendo maior robustez e facilidade de instalação.

Com ambos os sistemas calibrados, foram efetuadas aquisições simultâneas dos dados de torque durante a aplicação de cargas variáveis no eixo traseiro esquerdo do veículo, permitindo a coleta dos sinais para análise comparativa e validação da solução proposta.

5.3.1 Gráficos Gerados a Partir dos Testes Práticos

Após a aquisição do sinal de torque bruto via ESP32, foi realizada uma etapa de tratamento dos dados com o objetivo de remover valores atípicos e possibilitar uma análise fiel do comportamento do sistema.

Para isso, foi aplicado um filtro baseado no método IQR, implementado no código apresentado no Apêndice G. O método IQR consiste em calcular os limites inferior e superior com base nos quartis do conjunto de dados e identificar os pontos que se encontram fora desse intervalo como potenciais *outliers*. No entanto, para evitar a exclusão de variações reais do sistema, o código foi ajustado para aplicar a filtragem apenas em casos de até dois valores consecutivos fora do intervalo interquartílico. A partir do terceiro valor consecutivo, os dados são mantidos no conjunto final, sendo considerados parte do comportamento legítimo do sistema.

Essa estratégia foi definida com base em uma análise manual dos arquivos brutos gerados durante os testes práticos. Observou-se que os *outliers* ocorrentes eram majoritariamente isolados, o que justificou a escolha por um critério de filtragem conservador, evitando a eliminação de

sequências reais de variação de torque.

Em sequência, os dados filtrados foram utilizados para comparação com os sinais obtidos pelo sensor com fio, utilizando o código descrito no Apêndice H. Os gráficos resultantes dessa análise são apresentados a seguir, para os três testes práticos realizados. Em cada teste, são exibidos dois gráficos: o primeiro mostra o sinal do sensor ESP32 com os outliers isolados destacados, e o segundo apresenta a comparação entre os sinais do sensor com fio e sem fio.

Teste 1

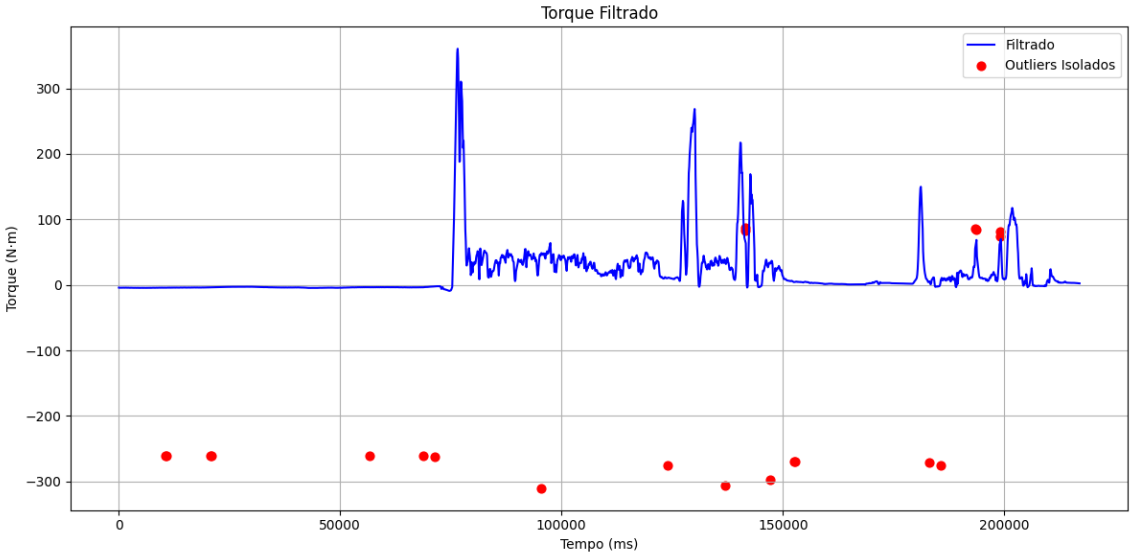

Figura 17 – Teste 1 — Torque com outliers isolados destacados (ESP32).

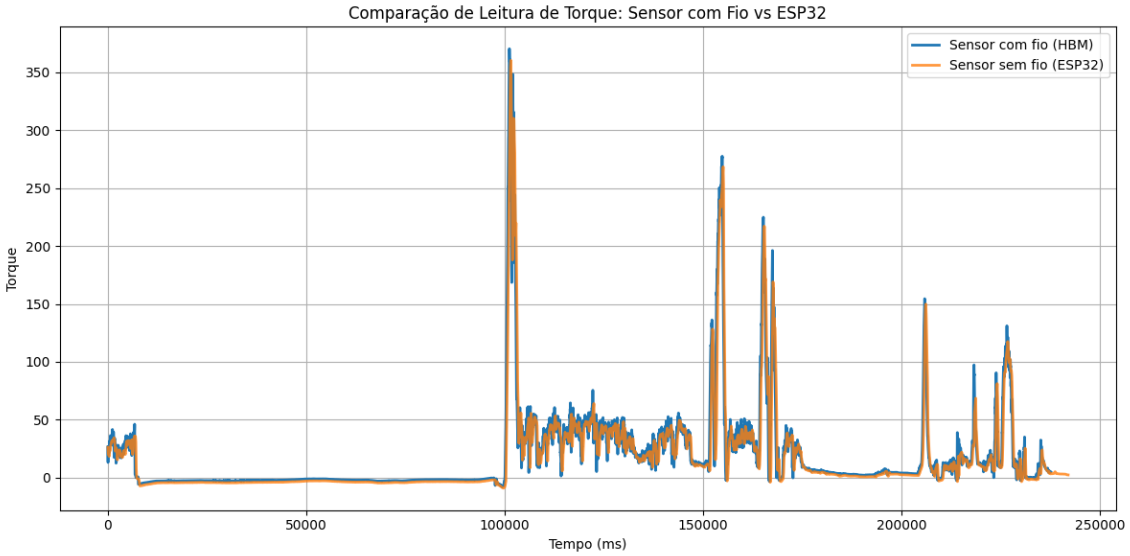

Figura 18 – Teste 1 — Comparação entre sensores com fio (HBM) e sem fio (ESP32).

No Teste 1, foi realizada inicialmente a filtragem do sinal obtido pelo ESP32, de forma a

identificar e remover valores discrepantes que poderiam comprometer a análise. Na Figura 17 é possível observar o sinal de torque após o processamento, no qual os *outliers* foram destacados em vermelho. Nota-se que esses pontos apresentam magnitudes diferentes em relação ao comportamento predominante do sinal, evidenciando a necessidade da etapa de filtragem para a correta interpretação dos resultados.

A Figura 18 apresenta a comparação direta entre o sinal obtido pelo sensor sem fio (ESP32) e o sensor de referência com fio (HBM). Observa-se que ambos apresentam comportamento bastante semelhante ao longo do ensaio, acompanhando de forma consistente as variações de torque aplicadas. Apesar da presença de pequenas discrepâncias pontuais, especialmente em regiões de maior oscilação, o sinal do ESP32 reproduz de maneira satisfatória a tendência do sensor de referência, demonstrando a viabilidade da solução proposta.

Teste 2

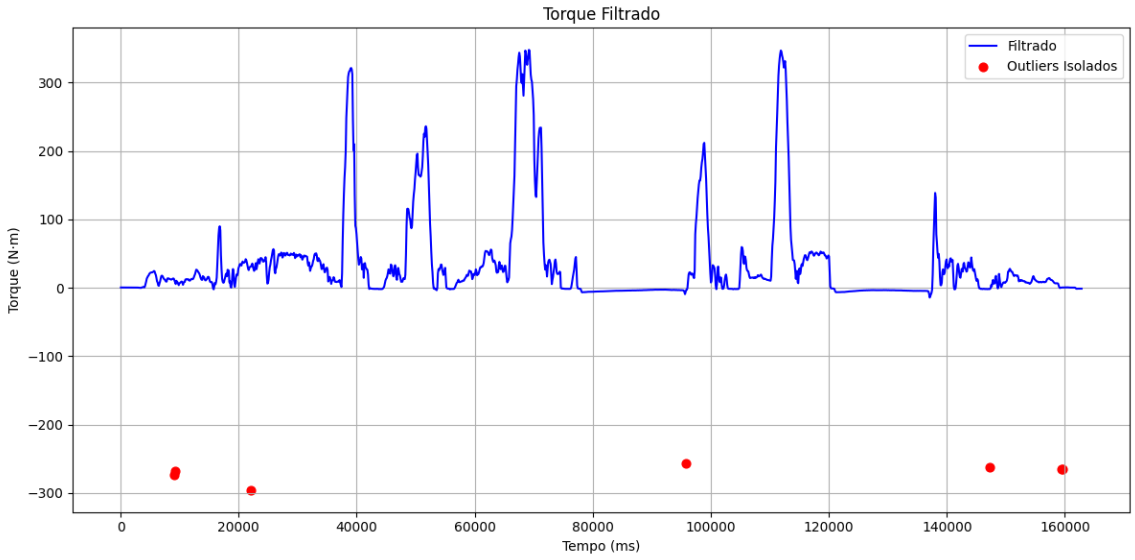

Figura 19 – Teste 2 — Torque com outliers isolados destacados (ESP32).

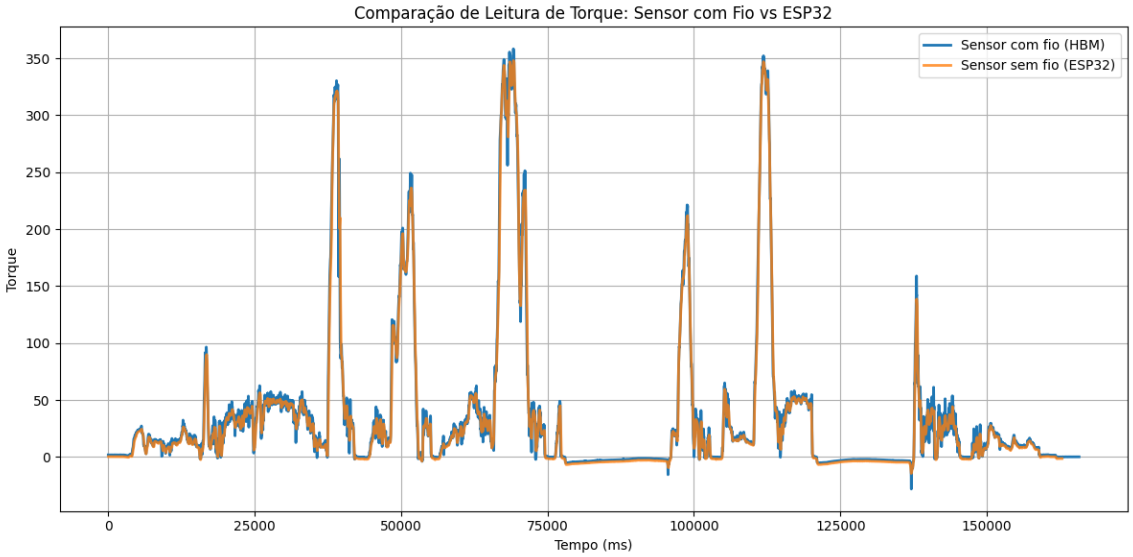

Figura 20 – Teste 2 — Comparação entre sensores com fio (HBM) e sem fio (ESP32).

No Teste 2, novamente foi realizada a filtragem do sinal proveniente do ESP32 para identificação de valores atípicos. A Figura 19 mostra o torque medido após o processamento, no qual alguns *outliers* isolados aparecem destacados em vermelho. Esses pontos destoam do padrão do sinal e poderiam interferir negativamente na análise estatística e na comparação entre os sensores, justificando a aplicação do filtro.

Na Figura 20 é apresentada a comparação entre os sinais do sensor de referência (HBM) e do sistema sem fio (ESP32). De modo geral, observa-se uma sobreposição bastante próxima entre os dois sinais, evidenciando que o ESP32 foi capaz de acompanhar com boa precisão as variações de torque registradas pelo HBM. Pequenas diferenças podem ser notadas em regiões de picos acentuados, mas o comportamento global confirma a coerência entre as medições.

Teste 3

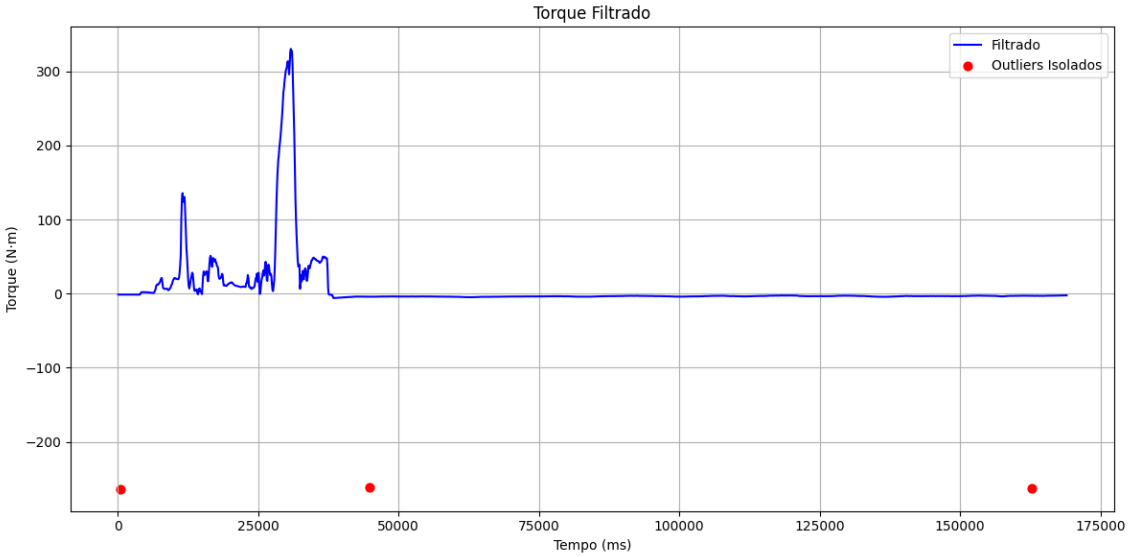

Figura 21 – Teste 3 — Torque com outliers isolados destacados (ESP32).

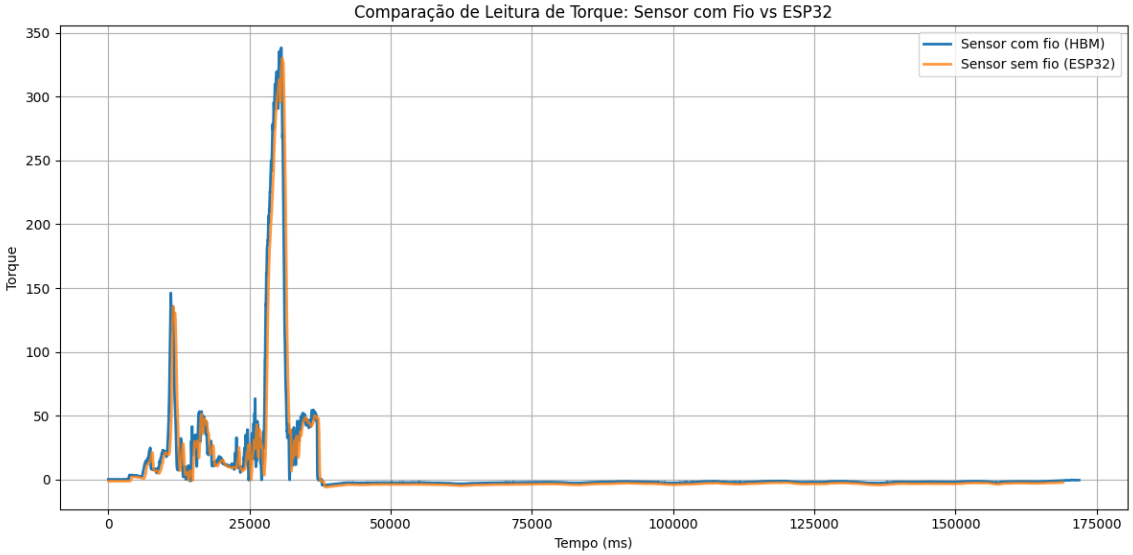

Figura 22 – Teste 3 — Comparação entre sensores com fio (HBM) e sem fio (ESP32).

No Teste 3, o sinal obtido pelo ESP32 apresentou menor quantidade de variações ao longo do tempo em comparação aos testes anteriores. Na Figura 21, observa-se o torque filtrado, com poucos pontos classificados como *outliers* e destacados em vermelho. Esses valores destoam do padrão do ensaio, mas aparecem de forma isolada, o que demonstra maior estabilidade geral do sinal nesta condição de teste.

A comparação entre o sensor de referência (HBM) e o sistema sem fio (ESP32), mostrada na Figura 22, evidencia uma boa concordância entre os dois sinais. Ambos acompanham as mesmas variações de torque, inclusive nos picos mais acentuados. Após aproximadamente

40.000 ms, nota-se a estabilização do torque próximo a zero em ambos os sensores, reforçando a consistência das medições.

5.3.2 Processamento e extração de métricas

Para permitir a comparação objetiva entre os sinais adquiridos pelos sistemas com fio e sem fio, foi desenvolvido um *script* em Python (disponível no Anexo I) que realiza o processamento e análise dos dados registrados em cada teste prático.

Inicialmente, os arquivos CSV gerados por cada sistema são lidos e pré-processados, com padronização dos nomes das colunas. Em seguida, o sinal obtido pelo sistema sem fio é interpolado linearmente para os instantes de tempo correspondentes ao sinal com fio. Esse procedimento garante que os vetores de torque estejam alinhados temporalmente, viabilizando a comparação ponto a ponto.

Com os sinais alinhados, calcula-se o erro absoluto entre os valores registrados, permitindo a obtenção das seguintes métricas:

- **Erro Absoluto Médio (MAE)**: representa a média das diferenças absolutas entre os sinais interpolados, expressa em N·m;
- **Erro Máximo Observado**: maior diferença pontual entre os sinais;
- **Tempo do Erro Máximo**: instante, em milissegundos, em que ocorreu o maior erro absoluto;
- **Correlação de Pearson**: mede a similaridade entre os sinais, indicando o quanto variam conjuntamente.

Além disso, o código também exibe o número total de amostras de cada sistema e a duração de cada aquisição. Essas informações são fundamentais para contextualizar os resultados obtidos nos testes práticos.

As métricas extraídas por esse procedimento serão utilizadas ao longo das próximas seções para a análise individual de cada teste.

5.3.2.1 Análise Quantitativa dos Resultados do Teste 1

A Tabela 4 apresenta as principais métricas obtidas a partir da análise comparativa entre os sinais de torque adquiridos pelo sistema com fio (HBM) e pelo sensor sem fio desenvolvido (ESP32). Os dados foram processados com interpolação para alinhar os sinais no tempo, considerando a diferença nas taxas de amostragem: aproximadamente 100 Hz para o sistema com fio e 10 Hz para o sensor sem fio.

Tabela 4 – Resumo dos resultados analíticos do Teste 1.

Parâmetro	**Valor**
Número de amostras (com fio)	23.765
Número de amostras (sem fio)	2.344
Duração da aquisição (com fio)	237,64 s
Duração da aquisição (sem fio)	241,95 s
Tempo do erro máximo	102.060 ms
Erro absoluto médio (MAE)	6,723 N.m
Erro máximo observado	159,607 N.m
Correlação de Pearson	0,922

Observa-se que o sistema com fio apresenta uma taxa de amostragem aproximadamente dez vezes maior que o sensor sem fio, o que resulta em um sinal mais detalhado e com menos suavização. Por outro lado, o sensor sem fio, operando próximo a 10 Hz, produz um sinal mais suavizado, devido à menor resolução temporal dos dados.

O pico de erro máximo, registrado no tempo de 102.060 ms, é resultado de um evento de torque abrupto detectado pelo sistema com fio, porém não captado pelo sensor sem fio. Essa diferença pode ser atribuída à menor taxa de amostragem e à suavização inerente ao sistema sem fio, o que limita a detecção de variações rápidas no sinal.

A Figura 23 ilustra visualmente esse evento de erro máximo, evidenciando a divergência entre os sinais e corroborando as métricas apresentadas na tabela.

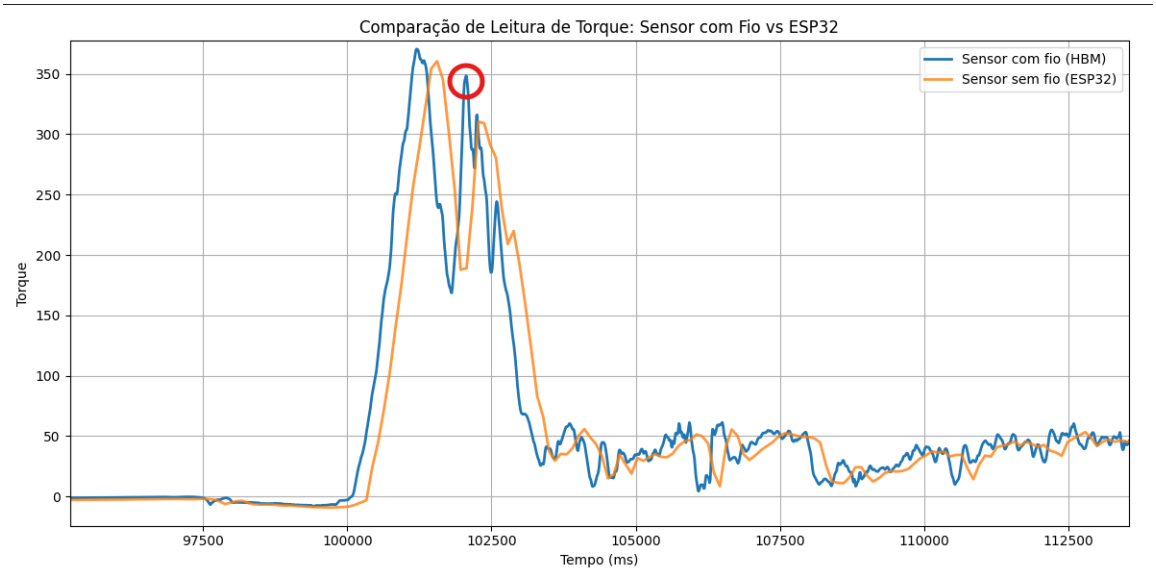

Figura 23 – Teste 1 — Analise detalhada do erro máximo observado.

Em geral, a alta correlação de Pearson ($r = 0,922$) demonstra boa concordância entre os dois sistemas de aquisição, indicando que o sensor sem fio apresenta desempenho satisfatório para medições dinâmicas de torque, apesar das limitações na detecção de picos abruptos.

5.3.2.2 Análise Quantitativa dos Resultados do Teste 2

A Tabela 5 apresenta as principais métricas obtidas a partir da comparação entre os sinais de torque adquiridos pelo sistema com fio (HBM) e pelo sensor sem fio baseado em ESP32. Assim como no Teste 1, os sinais foram interpolados para permitir a comparação temporal adequada.

Tabela 5 – Resumo dos resultados analíticos do Teste 2.

Parâmetro	**Valor**
Número de amostras (com fio)	16.573
Número de amostras (sem fio)	1.582
Duração da aquisição (com fio)	165,74 s
Duração da aquisição (sem fio)	162,79 s
Tempo do erro máximo	39.350 ms
Erro absoluto médio (MAE)	3,430 N.m
Erro máximo observado	95,562 N.m
Correlação de Pearson	0,996

Em comparação com o Teste 1, os resultados obtidos no Teste 2 demonstram uma melhoria na precisão do sistema sem fio. O MAE foi reduzido para 3,43 N.m, e a correlação de Pearson aumentou para 0,996, indicando concordância entre os sinais.

O pico de erro máximo foi registrado em 39.350 ms, novamente atribuído a um evento pontual de variação rápida de torque que não foi detectado pelo sensor sem fio, devido à menor taxa de amostragem (cerca de 10 Hz), em comparação aos cerca de 100 Hz do sistema com fio.

Essa diferença de resolução temporal implica que o sinal sem fio é naturalmente mais suavizado, o que pode resultar na atenuação ou ausência de picos agudos presentes no sinal de referência.

Ainda assim, os resultados do Teste 2 confirmam a boa capacidade do sistema sem fio em capturar com fidelidade o comportamento geral do torque aplicado, mesmo em situações com variações dinâmicas, desde que não envolvam transientes extremamente rápidos.

A Figura 24 mostra visualmente o instante de maior discrepância entre os sinais, permitindo contextualizar o valor observado na métrica de erro máximo.

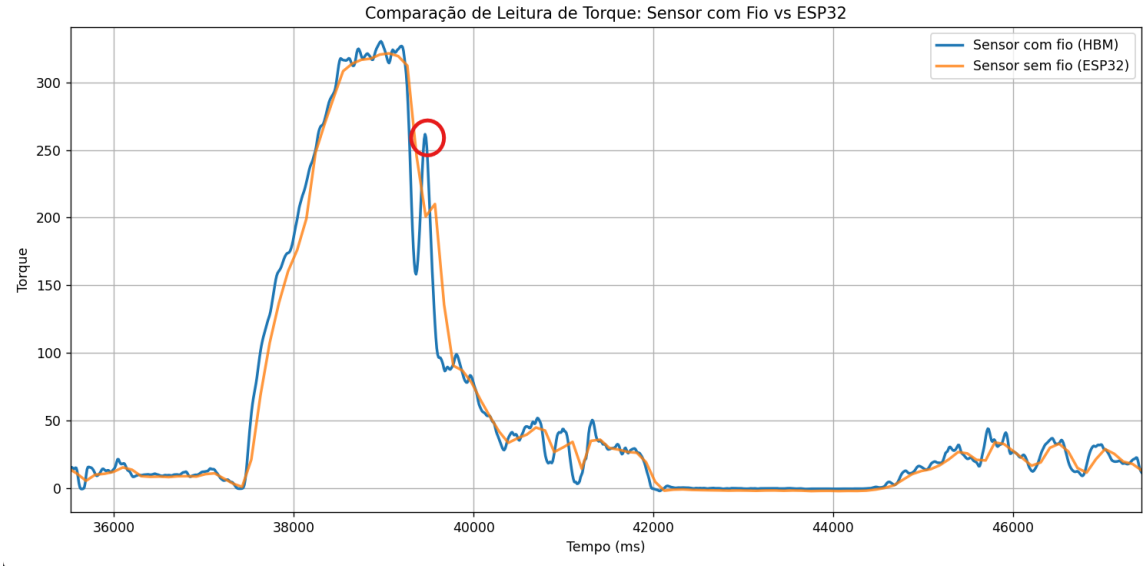

Figura 24 – Teste 2 — Analise detalhada do erro máximo observado.

5.3.2.3 Análise Quantitativa dos Resultados do Teste 3

A Tabela 6 apresenta os principais parâmetros obtidos na comparação entre os sinais com e sem fio no Teste 3. Este ensaio foi realizado durante o retorno do veículo à oficina, o que resultou em um longo intervalo com baixos níveis de torque, devido à condução em baixa rotação e sem solicitações bruscas.

Tabela 6 – Resumo dos resultados analíticos do Teste 3.

Parâmetro	**Valor**
Número de amostras (com fio)	17.187
Número de amostras (sem fio)	1.645
Duração da aquisição (com fio)	171,86 s
Duração da aquisição (sem fio)	168,71 s
Tempo do erro máximo	31.180 ms
Erro absoluto médio (MAE)	3,194 N.m
Erro máximo observado	92,951 N.m
Correlação de Pearson	0,980

Este foi o teste com menor MAE (3,194 N.m) entre todos os realizados, o que pode ser atribuído à ausência de variações bruscas na forma de condução. Como o veículo trafegava de forma estável e com torque reduzido por boa parte do tempo, o sistema sem fio, mesmo operando a uma taxa de amostragem inferior (10 Hz), conseguiu acompanhar adequadamente a dinâmica do sinal.

A Figura 25 ilustra a porção final do teste, evidenciando esse comportamento mais uniforme. Nota-se que ambos os sinais permanecem praticamente paralelos, com pequenas

variações, indicando uma aderência entre as duas leituras. Isso explica, de forma visual, o baixo valor de MAE registrado neste teste.

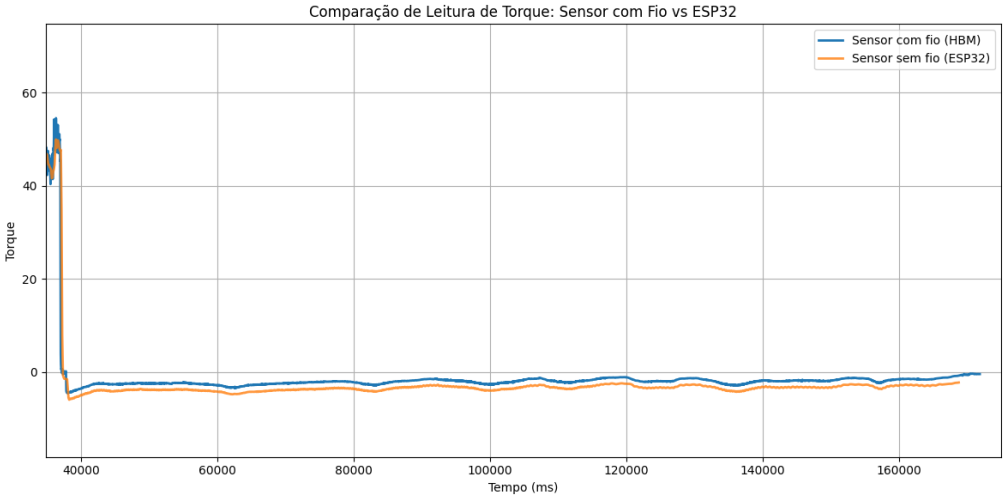

Figura 25 – Trecho final do Teste 3, destacando a estabilidade dos sinais com e sem fio em regime de torque reduzido.

Por fim, a elevada correlação de Pearson (0,980) reforça a boa fidelidade do sistema sem fio, mesmo em cenários com baixa resolução temporal, desde que a condição de operação seja estável.

5.3.3 Considerações Finais dos Testes Práticos

Os testes realizados demonstraram a eficácia e robustez do sistema de aquisição de torque sem fio desenvolvido. Os resultados obtidos, tanto quantitativos quanto qualitativos, revelam um bom desempenho mesmo frente às limitações naturais da comunicação *Wi-Fi* e da taxa de amostragem reduzida (10 Hz) em comparação ao sistema com fio (100 Hz).

Em todos os ensaios, foi possível observar uma correlação de Pearson elevada entre os sinais com e sem fio, variando entre 0,922 e 0,996, o que indica uma forte associação linear entre os dados adquiridos. Além disso, os valores do MAE permaneceram em níveis aceitáveis para aplicações automotivas experimentais, com destaque para o Teste 3, que apresentou o menor MAE (3,194 N.m) devido à condição de condução mais estável.

A precisão observada é resultado direto de três pilares fundamentais: a instrumentação bem executada, a calibração e a aplicação de técnicas de regressão linear. A etapa de calibração, realizada por meio da OLS, permitiu transformar a leitura bruta do sinal em valores confiáveis de torque, corrigindo não linearidades e removendo deslocamentos indesejados. Aliado a isso, o uso de técnicas de filtragem e análise robusta de outliers contribuiu para eliminar ruídos, garantindo sinais limpos.

Os testes ainda evidenciam a capacidade do sistema sem fio de captar variações de torque relevantes, mesmo com menor resolução temporal, desde que a dinâmica do sistema testado seja compatível com a taxa de amostragem utilizada. Esse comportamento torna o sistema para aplicações de monitoramento em tempo real ou coleta de dados em testes de desenvolvimento, sobretudo onde o uso de cabos seja inviável ou indesejado.

Conclui-se, portanto, que o sistema proposto atende satisfatoriamente aos objetivos do projeto, oferecendo uma solução de baixo custo, sem *slip rings*, com boa fidelidade de medição e potencial de aplicação prática em ambientes automotivos reais.

5.4 Comparativo de Soluções para Aquisição de Torque Automotivo: *LORD MicroStrain*, *Slip Ring* e Sistema Customizado Desenvolvido

Atualmente, diversas soluções estão disponíveis no mercado para a aquisição de dados provenientes de sensores do tipo *strain gauge* em aplicações automotivas. Esta seção apresenta uma análise comparativa, tanto técnica quanto econômica, entre três abordagens distintas: o sistema sem fio *LORD MicroStrain*, a solução convencional utilizando *Slip Rings* (anéis coletores) e o sistema de baixo custo desenvolvido neste projeto, denominado Sistema de Aquisição Customizado.

5.4.1 Sistema *LORD MicroStrain* Wireless

O sistema *LORD MicroStrain* é amplamente reconhecido pela robustez e confiabilidade em aplicações de campo. Seus principais componentes são os módulos *SG-Link-200-OEM* (2 canais) e *V-Link-200* (8 canais). De acordo com informações disponíveis na distribuidora *Mouser*, o módulo *SG-Link-200-OEM* possui custo aproximado de R$ 4.668,00 (cotação de 07/08/2025, com base em 1 dólar = R$ 5,20) (ELECTRONICS, 2025a). Já o *V-Link-200* apresenta valor estimado de R$ 7.468,00 (ELECTRONICS, 2025b).

O sistema é configurado e monitorado por meio do software *SensorConnect*, disponibilizado gratuitamente pela fabricante. A conexão do *strain gauge* é realizada diretamente no módulo *SG-Link*, que se encarrega da alimentação da ponte de *Wheatstone*, condicionamento do sinal e digitalização. Os dados são transmitidos sem fio por protocolo proprietário (*LXRS*®), com alcance de até 1 km, sendo recebidos por uma base conectada ao computador.

5.4.2 Sistema com *Slip Ring* (Anéis Coletores)

A solução convencional para aquisição de sinais em sistemas rotativos emprega *Slip Rings*, dispositivos eletromecânicos responsáveis pela transferência de sinais elétricos entre um eixo giratório e uma estrutura fixa. Modelos compactos com seis vias, suportando corrente

de até 2 A por canal, são encontrados no mercado com valores entre R$ 250,00 e R$ 480,00 (ALIEXPRESS, 2025).

A montagem desse sistema requer a fixação do *Slip Ring* no eixo rotativo, com cabeamento conectado ao *strain gauge*. Os sinais são então direcionados a um sistema de aquisição de alta precisão, como o *HBM QuantumX MX1615B*, responsável pela digitalização. Embora o custo inicial seja mais acessível, essa abordagem impõe limitações mecânicas relevantes, incluindo desgaste físico, atrito, ruído elétrico e restrição à rotação contínua.

5.4.3 Sistema de Aquisição Customizado Desenvolvido neste Projeto

A solução proposta neste projeto consiste em um sistema de aquisição de baixo custo, baseado nos seguintes componentes: microcontrolador *ESP32*, amplificador de carga *HX711*, duas baterias *18650* de 3,7 V e um módulo *MT3608 Step-Up*. Os custos atualizados em 07/08/2025 são:

- *ESP32* (30 pinos): R$ 43,90 (ELETROGATE, 2025b)
- *HX711*: R$ 9,90 (ELETROGATE, 2025a)
- 2 baterias *18650* (3,7 V, 3800 mAh): R$ 36,00 (R$ 18,00 cada) (LIVRE, 2025)
- Módulo *MT3608 Step-Up*: R$ 7,90 (ROBóTICA, 2025)

O custo total do sistema é de R$ 97,70. A montagem envolve a conexão direta do *strain gauge* ao módulo *HX711*, responsável pela amplificação e conversão do sinal para formato digital. O *ESP32* realiza o processamento e a transmissão dos dados via *Wi-Fi* (protocolo *TCP/IP*) para um *software* de visualização desenvolvido em *Python*. Apesar de apresentar limitações quanto à precisão e robustez, trata-se de uma alternativa viável para fins acadêmicos e experimentações iniciais.

5.4.4 Integração com o Sistema *QuantumX* e a Rede *CAN*

Tanto o sistema *LORD MicroStrain* quanto a abordagem com *Slip Ring* exigem a integração com um sistema de aquisição central, como o *HBM QuantumX*, com o objetivo de sincronizar os dados adquiridos com informações veiculares provenientes da rede *CAN*.

O *software Catman*® Easy/AP, utilizado em conjunto com o *QuantumX*, permite a sincronização de sinais provenientes de sensores *strain gauge* com parâmetros veiculares relevantes, tais como:

- Rotação do motor (*RPM*)
- Velocidade do veículo

- Abertura da borboleta (*Throttle Position Sensor - TPS*)
- Pressão do óleo do motor
- Temperatura do fluido de transmissão

Essa integração é fundamental para a realização de ensaios de torque e análise de desempenho veicular, possibilitando avaliações precisas sobre o comportamento dinâmico e a eficiência mecânica do sistema sob estudo.

5.4.5 Análise Técnica Comparativa entre Soluções de Aquisição de Torque

A seguir, conforme apresentado na Tabela 7 apresenta-se uma análise técnica comparativa entre três diferentes abordagens para aquisição de torque automotivo utilizando sensores do tipo *strain gauge*.

São comparados aspectos como o tipo de transmissão dos dados, a precisão e resolução dos sinais adquiridos, a taxa de amostragem possível, a robustez esperada para uso em ambientes automotivos e industriais, os custos estimados e os *softwares* de operação e interface. A proposta da tabela é destacar não apenas o desempenho de cada solução, mas também as vantagens e limitações envolvidas em cada abordagem, possibilitando ao leitor uma compreensão clara sobre os compromissos técnicos e financeiros de cada alternativa.

Tabela 7 – Comparativo: Modelo com fio (QuantumX) vs LORD MicroStrain vs Sistema Customizado

Característica	**Modelo com fio (QuantumX)**	**LORD MicroStrain (SG-Link)**	**Sistema Customizado (ESP32 + HX711)**
Tipo de Transmissão	Fio (Slip Ring)	Wireless (2.4 GHz, protocolo LXRS®)	Wireless (Wi-Fi TCP/IP via socket)
Precisão e Resolução	Alta (24 bits reais, ruído < 1 µV)	Alta (24 bits, ruído 1 µV)	Moderada (24 bits no HX711, maior ruído)
Taxa de Amostragem	Até 40 kHz (exemplo MX410B)	Até 1 kHz	Até 10–80 Hz (limitado pelo HX711 e Wi-Fi)
Robustez Industrial	Alta (laboratório/indústria)	Alta (campo e ambiente ruidoso)	Baixa (prototipagem)
Custo Aproximado	R$ 250,00 a R$ 480,00 (Slip Ring)	R$ 12.000 (SG-Link-200 OEM + V-Link-200)	R$ 97,70 (ESP32 + HX711)
Software	*catman® Easy/AP* (licença paga)	*SensorConnect* (gratuito) / *SensorCloud* (pago)	Interface própria em *Python*

A Figura 26 apresenta a distribuição percentual dos custos envolvidos em cada uma das três soluções analisadas. Nota-se que a maior parcela corresponde ao sistema *LORD MicroStrain*, representando aproximadamente 95,5% do custo total comparado. Em seguida, observa-se o sistema com *Slip Ring*, que corresponde a cerca de 3,8%, enquanto o sistema desenvolvido neste trabalho representa apenas 0,8% do custo total.

Essa diferença expressiva evidencia uma das principais vantagens do protótipo customizado: a viabilidade econômica para aplicações experimentais ou de pesquisa, sem comprometer a funcionalidade básica necessária para aquisição de torque.

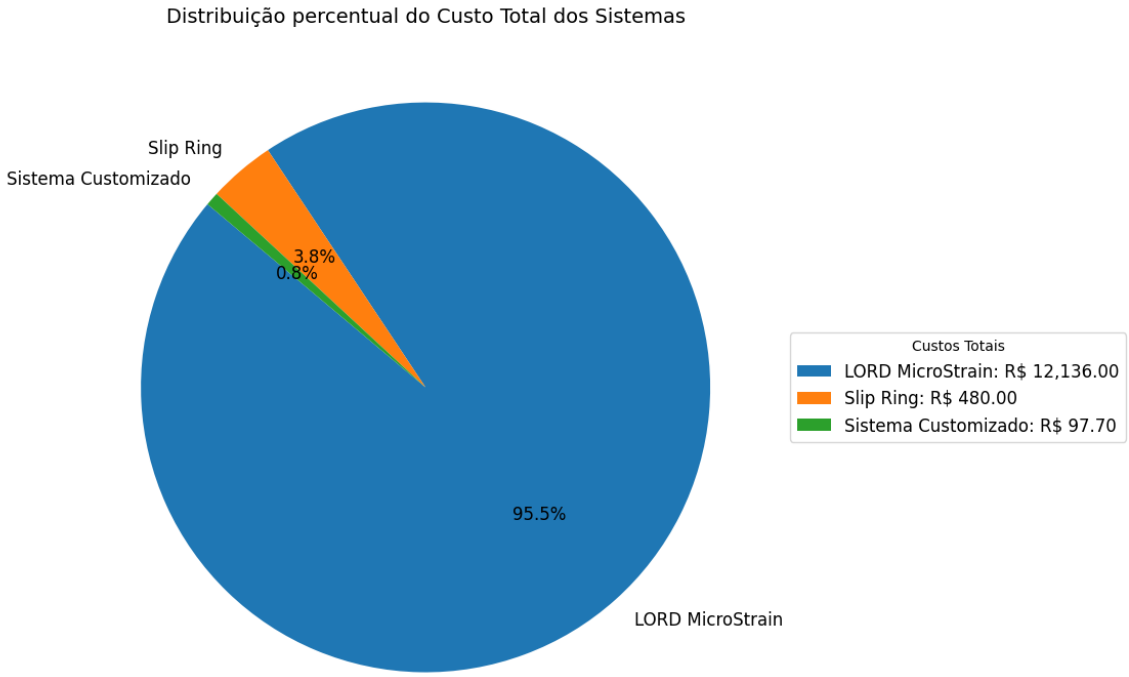

Figura 26 – Distribuição percentual dos custos entre as soluções analisadas: *LORD MicroStrain*, *Slip Ring* e o sistema desenvolvido neste trabalho.

Além do custo do *hardware*, os sistemas comerciais como o *LORD MicroStrain* e o *QuantumX* exigem o uso de softwares proprietários como o *SensorConnect*/*SensorCloud* e o *catman® Easy/AP*, respectivamente. Que podem representar custos adicionais, além de requerer infraestrutura específica e, em alguns casos, licenças pagas. Por outro lado, o sistema proposto neste projeto adota uma interface desenvolvida em *Python*, livre de custos adicionais, com maior flexibilidade para modificações conforme a necessidade do usuário.

Portanto, considerando o aspecto econômico e a proposta de acessibilidade tecnológica, o sistema baseado em *ESP32* e *HX711* apresenta-se como uma solução altamente vantajosa em termos de custo-benefício, especialmente para contextos acadêmicos ou de prototipagem inicial.

6 CONCLUSÃO E TRABALHOS FUTUROS

O presente trabalho teve como objetivo o desenvolvimento e a validação de um sistema de aquisição de torque automotivo sem fio, de baixo custo e aplicável a contextos experimentais. Os resultados demonstraram que todos os objetivos propostos foram alcançados com êxito. O sistema desenvolvido realizou medições com correlação de Pearson acima de 0,90 em relação ao sensor comercial de referência, mantendo precisão adequada mesmo com limitação de taxa de amostragem de 10 Hz.

A etapa de instrumentação, aliada a uma calibração precisa por meio de *Regressão Linear Ordinária (OLS)*, foi fundamental para os bons resultados práticos obtidos nos testes veiculares. A aplicação de técnicas de filtragem e remoção de *outliers* também contribuiu para a qualidade do sinal, permitindo que o sistema acompanhasse a dinâmica do torque, sobretudo em condições de condução estável.

A análise comparativa entre o sistema desenvolvido, a solução com *Slip Ring* e o sistema *LORD MicroStrain* evidenciou a vantagem econômica expressiva da abordagem proposta neste trabalho, que representa menos de 1% do custo das soluções comerciais. Isso demonstra o potencial do sistema para uso em projetos acadêmicos, protótipos e experimentações, especialmente em cenários onde a utilização de cabos é inviável ou indesejada.

Como trabalhos futuros, propõe-se o aumento da taxa de aquisição, que foi limitada a 10 Hz nos testes práticos. O *HX711*, em sua configuração máxima, permite operar a até 80 Hz, e essa capacidade poderá ser explorada em versões futuras do sistema. Alternativamente, pode-se considerar o uso de outros conversores *A/D* que viabilizem taxas de amostragem mais elevadas, na ordem de 100 Hz ou superiores, compatíveis com aplicações dinâmicas mais exigentes.

Outra possibilidade seria a sincronização com a rede CAN do veículo, de forma similar ao que é realizado pelos sistemas comerciais como o *HBM QuantumX*. Essa integração permitiria análises mais completas, correlacionando os dados de torque com parâmetros veiculares como rotação do motor, velocidade e pressão de óleo.

Além disso, ressalta-se a importância de investigar alternativas de comunicação sem fio mais robustas, uma vez que o uso de *Wi-Fi* demonstrou limitações quanto ao alcance e à estabilidade da conexão em ambientes automotivos. Tecnologias como *LoRa*, ou outros protocolos de longo alcance e baixa latência, podem representar um avanço importante na confiabilidade e na aplicabilidade do sistema em campo.

Conclui-se, portanto, que o sistema desenvolvido atinge os objetivos de baixo custo, funcionalidade e aplicabilidade prática, apresentando-se como uma plataforma para medições de torque em veículos. Entretanto, limitações como taxa de amostragem restrita, precisão inferior a sistemas comerciais, ausência de funcionalidades avançadas como integração com CAN e durabilidade da bateria devem ser consideradas, indicando potencial para aprimoramentos e expansões em futuras versões.

REFERÊNCIAS

ALIEXPRESS. **6 Wire 2A Capsule Slip Ring Electrical Rotating Connector**. 2025. Acesso em: 07 ago. 2025. Disponível em: <https://pt.aliexpress.com/item/1005007440277763.html>.

ARDUINO. **Arduino Software**. 2024. Acesso em: 16 set. 2024. Disponível em: <https://www.arduino.cc/en/software>.

AUTOS, C. **O que é o torque de um motor? Para que serve?** 2024. Disponível em: <https://autos.culturamix.com/noticias/o-que-e-o-torque-de-um-motor-para-que-serve>.

BLUM, J. **Exploring Arduino: Tools and Techniques for Engineering Wizardry**. [S.l.]: John Wiley & Sons, 2013.

CHAI, T.; DRAXLER, R. R. Root mean square error (rmse) or mean absolute error (mae)?–arguments against avoiding rmse in the literature. **Geoscientific model development**, Copernicus Publications Göttingen, Germany, v. 7, n. 3, p. 1247–1250, 2014.

CLARK, D. S.; DUWEZ, P. E. A method of measuring the forces acting on a specimen during a tension impact test. **Journal of Applied Mechanics**, v. 15, n. 3, p. 243–247, 1948.

COSTA, R. **Novas tecnologias: carros atuais têm até 100 sensores a bordo**. 2024. Acesso em: 09 ago. 2024. Disponível em: <https://quatrorodas.abril.com.br/noticias/novas-tecnologias-carros-atuais-tem-ate-100-sensores-a-bordo/>.

DANCEY, C. P.; REIDY, J. **Estatística sem matemática para psicologia**. 5. ed. Porto Alegre: Penso, 2013.

DHULL, P. *et al.* Internet of things networks: Enabling simultaneous wireless information and power transfer. **IEEE microwave magazine**, IEEE, v. 23, n. 3, p. 39–54, 2022.

DOEBELIN, E. O. **Measurement Systems: Application and Design**. 4th. ed. [S.l.]: McGraw-Hill, 2003.

DRAPER, N. R.; SMITH, H. **Applied Regression Analysis**. 3rd. ed. New York: Wiley, 1998.

EL-KHOZONDAR, H. J.; AL. et. A smart energy monitoring system using esp32 microcontroller. **International Journal of Electrical and Electronics Engineering**, International Association of Engineers, v. 15, n. 4, p. 78–87, 2022.

ELECTRONICS, M. **SG-Link-200-OEM MicroStrain by HBK**. 2025. Acesso em: 07 ago. 2025. Disponível em: <https://br.mouser.com/ProductDetail/MicroStrain-by-HBK/SG-Link-200-OEM?qs=DPoM0jnrROXlUfCPpzD%2B4Q%3D%3D>.

____. **V-Link-200 MicroStrain by HBK**. 2025. Acesso em: 07 ago. 2025. Disponível em: <https://br.mouser.com/ProductDetail/MicroStrain-by-HBK/V-Link-200?qs=DPoM0jnrROWnxd3%252B4oBlZQ%3D%3D>.

ELECTRONICS, S. **HX711: Precision 24-Bit Analog-to-Digital Converter**. 2024. Acesso em: 15 set. 2024. Disponível em: <https://cdn.sparkfun.com/datasheets/Sensors/ForceFlex/hx711_english.pdf>.

Electrothinks. **MT3608 DC to DC Step-Up Converter Module**. 2023. Acesso em: 27 jul. 2025. Disponível em: <https://www.electrothinks.com/2023/04/mt3608-dc-to-dc-step-up-converter-module.html>.

ELETROGATE. **Módulo Conversor 24bit HX711 para Célula de Carga**. 2025. Acesso em: 07 ago. 2025. Disponível em: <https://www.eletrogate.com/modulo-conversor-24bit-hx711-p-celula-de-carga>.

____. **Módulo WiFi ESP32 Bluetooth - 30 Pinos**. 2025. Acesso em: 07 ago. 2025. Disponível em: <https://www.eletrogate.com/modulo-wifi-esp32-bluetooth-30-pinos>.

ESCOBEDO, P.; AL. et. Smart bandage with wireless strain and temperature sensors and batteryless nfc tag. **IEEE Transactions on Biomedical Circuits and Systems**, IEEE, v. 15, n. 2, p. 451–460, 2021.

KHALIFEH, A.; AL. et. Microcontroller unit-based wireless sensor network nodes: A review. **IEEE Sensors Journal**, IEEE, v. 21, n. 11, p. 12211–12222, 2021.

LIU, J.; AL. et. Research and application of wireless sensor network technology in power transmission and distribution system. **IEEE Access**, IEEE, v. 8, p. 85001–85011, 2020.

LIVRE, M. **Bateria Recarregável Knup KP-18650 3.7V 3800mAh**. 2025. Acesso em: 07 ago. 2025. Disponível em: <https://www.mercadolivre.com.br/bateria-recarregavel-knup-kp-18650-37v-3800mah/p/MLB21892087>.

MAIER, A.; SHARP, A.; VAGAPOV, Y. Comparative analysis and practical implementation of the esp32 microcontroller module for the internet of things. In: IEEE. **2017 Internet Technologies and Applications (ITA)**. [S.l.], 2017. p. 143–148.

MANSHAHIA, M. S. Wireless sensor networks: A survey. **Journal of Network and Computer Applications**, Elsevier, v. 153, p. 102530, 2020.

MAZIDI, M. A.; MAZIDI, J. G.; MCKINNEY, R. D. **AVR Microcontroller and Embedded Systems**. [S.l.]: Pearson, 2016.

MMA, H. **Ponte de Wheatstone**. 2024. Acesso em: 22 de setembro de 2024. Disponível em: <https://hangarmma.com.br/glossary/glossary-categories/ponte-de-wheatstone/>.

MONTGOMERY, D. C.; PECK, E. A.; VINING, G. G. **Introduction to Linear Regression Analysis**. 5th. ed. Hoboken, NJ: Wiley, 2012.

MORRIS, A. S. **Measurement and calibration for quality assurance**. [S.l.]: Prentice Hall, 1991.

MUFTAH, M. H.; AL. et. An improved strain gauge-based dynamic torque measurement method. **Sensors**, MDPI, v. 18, n. 8, p. 2665, 2018.

NORTON, H. N. **Handbook of Transducers**. California: Altadena, 1989.

OBERST, S. **Instrumentation for Engineers**. [S.l.]: McGraw-Hill, 2010.

PASTOR, M.; HAGARA, M.; GAŠPÁR, Š.; SAPIETA, M. Design and implementation of a low-cost torque sensor for manipulators. **Applied Sciences**, MDPI, v. 13, n. 16, p. 9406, 2023.

PERSSON, G.; PERSSON, O. Torque sensor for automotive applications: An investigation of torque sensing techniques for drivetrain integration. **SAE International Journal of Passenger Cars - Mechanical Systems**, SAE International, v. 12, n. 1, p. 15–30, 2019.

PLUSPNG. **Arduino Logo PNG**. 2024. Disponível em: <https://pluspng.com/arduino-logo-png-10264.html>.

POPOVIC, Z.; AL. et. Low-power far-field wireless powering for wireless sensors. **IEEE Transactions on Microwave Theory and Techniques**, IEEE, v. 67, n. 5, p. 1933–1942, 2019.

ROBõTICA, C. da. **Módulo Regulador de Tensão Ajustável MT3608 Step-Up 2.5V a 28V**. 2025. Acesso em: 07 ago. 2025. Disponível em: <https://www.casadarobotica.com/fontes-conversores/conversores-dc-dc/step-up/modulo-regulador-de-tensao-ajustavel-mt3608-step-up-2-5v-a-28v>.

ROMO-CHAVERO, M. A.; CANTORAL-CEBALLOS, J. A.; PÉREZ-DÍAZ, J. A.; MARTINEZ-CAGNAZZO, C. Median absolute deviation for bgp anomaly detection. **Future Internet**, MDPI, v. 16, n. 5, p. 146, 2024.

ROUSSEEUW, P. J.; LEROY, A. M. **Robust regression and outlier detection**. [S.l.]: John wiley & sons, 2003.

SANTAMARíA, L.; AL. et. Different calibration methods for a three-component strain gauge balance to measure aerodynamic forces on airfoils. **Journal of Aircraft**, AIAA, v. 57, n. 6, p. 1121–1134, 2020.

SEDRA, A. S.; SMITH, K. C. **Microelectronic Circuits**. 8th. ed. [S.l.]: Oxford University Press, 2017.

SINGH, P.; AL. et. Internet of things for sustainable railway transportation: Past, present, and future. **IEEE Access**, IEEE, v. 8, p. 150890–150911, 2020.

STONE, R.; BALL, J. K. **Automotive Engineering Fundamentals**. [S.l.]: SAE International, 2004.

SYSTEMS, E. **ESP32 Arduino Core**. 2024. Acesso em: 16 set. 2024. Disponível em: <https://github.com/espressif/arduino-esp32>.

____. **ESP32 Datasheet**. 2024. Acesso em: 15 set. 2024. Disponível em: <https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf>.

TAYLOR, C. **Interquartile Range (IQR) | Meaning, Formula & Calculation**. 2023. Accessed: 2025-07-28. Disponível em: <https://www.scribbr.co.uk/stats/interquartile-range-meaning/>.

T&M. **What is Strain Gauge?** 2024. Acesso em: 22 de setembro de 2024. Disponível em: <https://tandm.co.za/news/what-is-strain-gauge/>.

TUKEY, J. W. *et al.* **Exploratory data analysis**. [S.l.]: Springer, 1977. v. 2.

WEISBERG, S. **Applied Linear Regression**. 3rd. ed. Hoboken, NJ: Wiley, 2005.

WILSON, J. S. **Sensor technology handbook**. [S.l.]: Elsevier, 2005.

APÊNDICE A – TRECHO DA FUNÇÃO PARA COLETAR MÉDIA FILTRADA DAS LEITURAS VIA SOCKET

Este anexo apresenta a função Python responsável por coletar as leituras do sensor via conexão socket, realizando uma filtragem robusta dos dados para reduzir o impacto de ruídos e valores atípicos. A função utiliza o Desvio Absoluto da Mediana (MAD) para eliminar outliers e retorna a média filtrada das leituras, garantindo maior confiabilidade no processamento dos sinais recebidos do sensor.

```
1
2  #  Funcao para coletar a media filtrada das leituras do sensor
3  def coletar_media(self, qtd=100):
4      # Define um timeout curto para limpar dados antigos do buffer do socket
5      self.client_socket.settimeout(0.1)
6      try:
7          # Tenta esvaziar o buffer de dados recebidos para evitar leituras
               antigas
8          while True:
9              self.client_socket.recv(1024)
10     except:
11         pass  # Quando nao houver mais dados, o timeout vai estourar e sai
               do loop
12     # Remove o timeout para as leituras seguintes (modo blocking)
13     self.client_socket.settimeout(None)
14
15     valores = []  # Lista para armazenar as leituras recebidas do sensor
16
17     # Loop para coletar a quantidade desejada de leituras validas
18     while len(valores) < qtd:
19         try:
20             data = self.client_socket.recv(1024)  # Recebe os dados do
                   socket (bytes)
21             if not data:
22                 break  # Sai se nao houver dados recebidos
23             decoded = data.decode('utf-8').strip()  # Decodifica os bytes
                   em string UTF-8 e remove espacos
24             partes = decoded.split(',')  # Divide a string em partes usando
                    virgula como separador
25
26             # Percorre cada pedaco separado para extrair numeros inteiros
                   validos
27             for p in partes:
28                 p = p.strip()  # Remove espacos extras
29                 if p == "":
30                     continue  # Ignora strings vazias
31                 try:
32                     number = int(p)  # Converte para inteiro
```

```
33                      valores.append(number)  # Adiciona o numero a lista de
                            leituras
34                      if len(valores) >= qtd:
35                          break  # Sai quando atingir o numero desejado de
                                leituras
36                  except:
37                      continue  # Ignora qualquer dado que nao puder ser
                            convertido para inteiro
38          except:
39              break  # Sai do loop se ocorrer algum erro na recepcao dos
                    dados
40
41      if not valores:
42          return None  # Retorna None se nenhuma leitura valida foi coletada
43
44      valores_np = np.array(valores)  # Converte lista para array NumPy para
            facilitar operacoes
45
46      # Calcula a mediana das leituras para usar como valor central robusto
47      mediana = np.median(valores_np)
48
49      # Calcula o Desvio Absoluto da Mediana (MAD) para estimar a dispersao
            robusta
50      mad = np.median(np.abs(valores_np - mediana))
51
52      if mad == 0:
53          # Se o MAD for zero, nao ha dispersao, entao usa todos os valores
54          filtrados = valores_np
55      else:
56          # Define limites inferior e superior para filtrar outliers (3 vezes
                o MAD)
57          limite_inf = mediana - 3 * mad
58          limite_sup = mediana + 3 * mad
59
60          # Mantem somente os valores dentro dos limites, filtrando possiveis
                outliers
61          filtrados = valores_np[(valores_np >= limite_inf) & (valores_np <=
                limite_sup)]
62
63      # Retorna a media dos valores filtrados, ou a mediana caso nao haja
            valores filtrados
64      return np.mean(filtrados) if len(filtrados) else mediana
```

Listing A.1 – Função para coletar a média filtrada das leituras do sensor via socket

APÊNDICE B – CÓDIGO DE CALIBRAÇÃO DO STRAIN GAUGE

Este anexo apresenta o trecho de código Python utilizado no processo de calibração do sistema de medição baseado em strain gauges. O código rebece dados da função filtragem via MAD demonstrado no Anexo A e aplica a regressão linear por mínimos quadrados ordinários (OLS).

```
def calibrar(self):
    # Verifica se o socket de conexao com o ESP32 esta ativo
    if not self.client_socket:
        QMessageBox.warning(self, "Erro", "Conecte primeiro.")
        return  # Interrompe a calibracao caso nao haja conexao com o
            servidor

    quilos = []  # Lista para armazenar os pesos reais aplicados
    medias = []  # Lista para armazenar as medias dos valores lidos do
        sensor

    # Loop para realizar a calibracao com 7 diferentes cargas
    for i in range(7):
        # Abre uma janela de entrada para o usuario digitar o valor da
            carga (em kg)
        peso, ok = QInputDialog.getDouble(
            self, "Calibracao", f"Digite o peso {i + 1} em kg:", decimals=4
        )
        if not ok:
            return  # Interrompe a calibracao se o usuario cancelar a
                entrada

        quilos.append(peso)  # Armazena o peso fornecido
        self.log.append(f"Coletando dados para {peso} kg...")  # Loga a
            etapa no painel

        # Coleta a media de 100 amostras de leitura do sensor para esse
            peso
        media = self.coletar_media(100)
        if media is None:
            QMessageBox.warning(self, "Erro", "Falha na coleta.")
            return  # Interrompe a calibracao se ocorrer falha na leitura

        medias.append(media)  # Armazena a media das leituras do sensor

    # Converte listas para arrays NumPy para facilitar o uso com regressao
    medias_np = np.array(medias).reshape(-1, 1)  # Formato de matriz coluna
        para regressao
    quilos_np = np.array(quilos)  # Pesos reais em formato de vetor

    # Cria e treina um modelo de regressao linear (OLS) com os dados
```

```
        coletados
35      model = LinearRegression()
36      model.fit(medias_np, quilos_np)  # Ajusta os coeficientes a e b com
            base nos dados
37
38      a = model.coef_[0]        # Coeficiente angular da reta (a)
39      b = model.intercept_      # Intercepto da reta (b)
40
41      # Refina o ajuste usando a funcao de minimizacao para minimizar o erro
            absoluto medio
42      resultado = minimize(
43          lambda p: np.mean(np.abs((p[0] * medias_np.flatten() + p[1]) -
                quilos_np)),
44          x0=[a, b],  # Chute inicial baseado na regressao linear OLS
45          method='Nelder-Mead'  # Metodo de otimizacao numerica robusto
46      )
47
48      self.a, self.b = resultado.x  # Armazena os coeficientes finais
            otimizados
49
50      # Atualiza a interface com a equacao calibrada
51      self.equation_label.setText(f"Equacao: {self.a:.8f} * sensor + {self.b
            :.8f}")
52
53      # Adiciona a mensagem de sucesso ao log
54      self.log.append(f"Calibracao concluida: {self.a:.8f} * sensor + {self.b
            :.8f}")
```

Listing B.1 – Trecho do código de calibração com regressão linear OLS.

APÊNDICE C – TRECHO DO CÓDIGO DO ESP32 PARA COLETA E TRANSMISSÃO DOS DADOS

Este anexo apresenta uma parte do código embarcado no ESP32, responsável por realizar a leitura dos dados do sensor HX711 e transmitir essas informações via Wi-Fi para o computador cliente. O sistema opera no formato cliente-servidor, em que o ESP32 atua como servidor TCP, e o computador como cliente de recepção dos dados. O trecho a seguir corresponde à lógica de leitura e envio periódico:

```
// Verifica se o HX711 esta pronto para realizar uma nova leitura
if (scale.is_ready()) {

  // Realiza a leitura bruta do sinal digital proveniente do HX711
  long leituraBruta = scale.read();

  // Captura o tempo, em milissegundos, desde que o ESP32 foi iniciado
  unsigned long tempoLeitura = millis();

  // Envia o valor bruto da leitura para o cliente conectado via TCP
  client.print(leituraBruta);
  client.print(",");  // Envia um separador entre os campos

  // Envia o tempo de leitura correspondente
  client.print(tempoLeitura);
  client.print(";");  // Marca o final do pacote de dados

  // Imprime no terminal serial os dados enviados, para depuracao
  Serial.print("Bruto: ");
  Serial.print(leituraBruta);
  Serial.print(" | Tempo: ");
  Serial.println(tempoLeitura);

} else {
  // Caso o HX711 nao esteja pronto, informa via Serial e aguarda
  Serial.println("HX711 nao esta pronto, pulando leitura...");
  delay(100); // Aguarda 100ms antes de tentar novamente
  continue;   // Pula o restante do loop e tenta novamente
}

// Aguarda 100ms antes de realizar a proxima leitura
delay(100);
```

Listing C.1 – Trecho do código do ESP32 para leitura e envio dos dados do HX711.

APÊNDICE D – TRECHO DO CÓDIGO RESPONSÁVEL PELA AQUISIÇÃO E EXPORTAÇÃO DOS DADOS

Este trecho apresenta a implementação da thread que realiza a aquisição contínua dos dados enviados pelo ESP32 via socket. O código recebe os valores brutos do sensor e o timestamp correspondente, aplica a equação de calibração para converter o valor bruto em torque, e grava os dados em um arquivo texto no formato tabulado para posterior análise. Além disso, a thread emite sinais para atualizar a interface gráfica com as leituras em tempo real e trata possíveis desconexões ou erros durante a coleta.

```
1
2  def run(self):
3      # Abre o arquivo no modo escrita e adiciona o cabecalho das colunas
4      with open(self.filepath, 'w') as file:
5          file.write("TORQUE(N.m)\tTEMPO(ms)\n")
6
7      try:
8          # Loop principal que roda enquanto o atributo running for True
9          while self.running:
10             # Recebe ate 1024 bytes de dados do socket
11             data = self.socket.recv(1024)
12             # Se nao receber dados, emite sinal de erro e sai do loop
13             if not data:
14                 self.error_signal.emit("Servidor desconectou.")
15                 break
16
17             # Decodifica os bytes recebidos para string e remove espacos no
                   inicio/fim
18             decoded = data.decode('utf-8').strip()
19             # Separa os pacotes usando ponto e virgula como delimitador
20             pacotes = decoded.split(';')
21
22             # Itera sobre cada pacote recebido
23             for pacote in pacotes:
24                 # Ignora pacotes vazios ou que contenham somente espacos
25                 if not pacote.strip():
26                     continue
27                 try:
28                     # Separa o pacote em valor bruto e timestamp pela
                           virgula
29                     valor_str, timestamp_str = pacote.strip().split(',')
30                     # Converte valor bruto para inteiro
31                     number = int(valor_str)
32                     # Converte timestamp enviado pelo esp para inteiro
33                     timestamp_esp = int(timestamp_str)
34
35                     # Se ainda nao foi definido o tempo inicial do esp,
```

```
                    define agora
36                  if self.tempo_inicial_esp is None:
37                      self.tempo_inicial_esp = timestamp_esp
38
39                  # Calcula o tempo relativo subtraindo tempo inicial do
                        timestamp atual
40                  tempo_relativo = timestamp_esp - self.tempo_inicial_esp
41
42                  # Calcula o torque usando a equacao calibrada (a *
                        valor + b) e multiplica por 10 para escala correta
43                  torque = (self.a * number + self.b) * 10
44
45                  # Abre o arquivo em modo append e escreve o torque e
                        tempo relativo separados por tab
46                  with open(self.filepath, 'a') as file:
47                      file.write(f"{torque:.2f}\t{tempo_relativo}\n")
48
49                  # Emite sinal para atualizar o log da interface com os
                        dados atuais
50                  self.data_signal.emit(f"Torque: {torque:.2f} N.m |
                        Tempo ESP: {tempo_relativo} ms")
51              except:
52                  # Ignora qualquer excecao que ocorrer dentro do try
                        acima e continua o loop
53                  continue
54
55      except Exception as e:
56          # Emite sinal de erro caso alguma excecao ocorra no loop principal
57          self.error_signal.emit(str(e))
```

Listing D.1 – Trecho do código responsável pela aquisição e exportação dos dados.

APÊNDICE E – TRECHO DO CÓDIGO ESP32 PARA TRANSMISSÃO DOS DADOS VIA WI-FI

Este anexo apresenta o código-fonte desenvolvido em C++ para o microcontrolador ESP32, utilizando a plataforma Arduino IDE. O código é responsável por estabelecer conexão Wi-Fi com uma rede local e transmitir continuamente, em intervalos regulares, o tempo interno de sistema (em milissegundos) via protocolo TCP/IP. Esse valor é utilizado no computador receptor para análise da estabilidade da comunicação e possíveis perdas de dados durante os testes em ambiente ruidoso.

```
1
2  // Bibliotecas necess rias para conex o Wi-Fi
3  #include <WiFi.h>
4
5  // Dados da rede Wi-Fi
6  const char* ssid = "iPhone";
7  const char* password = "12345678";
8
9  // Cria  o do servidor e cliente TCP
10 WiFiServer server(8080);
11 WiFiClient client;
12
13 // Intervalo entre envios (em milissegundos)
14 unsigned long tempo_envio = 100;
15
16 void setup() {
17   Serial.begin(115200);
18   WiFi.begin(ssid, password);  // Inicia conex o com a rede
19
20   Serial.print("Conectando ao Wi-Fi");
21   while (WiFi.status() != WL_CONNECTED) {
22     delay(500);
23     Serial.print(".");
24   }
25
26   Serial.println("\nConectado ao Wi-Fi.");
27   Serial.println(WiFi.localIP());
28
29   server.begin();  // Inicia o servidor TCP
30 }
31
32 void loop() {
33   // Aguarda cliente conectado
34   if (!client || !client.connected()) {
35     client = server.available();
36     return;
37   }
```

```
38
39    // Envia o tempo atual em milissegundos
40    unsigned long agora = millis();
41    client.print(agora);
42    client.print(",");  // Separador (pode ser \n se desejado)
43
44    delay(tempo_envio);  // Aguarda o pr ximo envio
45  }
```

Listing E.1 – Código C++ do ESP32 para envio contínuo de dados via Wi-Fi

APÊNDICE F – SCRIPT PYTHON PARA TESTE DE CONECTIVIDADE COM O ESP32

Este anexo apresenta o script Python utilizado para realizar o teste de conectividade entre o computador e o microcontrolador ESP32 via rede Wi-Fi. O código é responsável por detectar automaticamente o endereço IP do ESP32 na rede local, estabelecer a conexão via socket TCP e registrar os dados recebidos em um arquivo de texto, com marcação temporal, para posterior análise. O teste foi realizado por 30 segundos em cada configuração, possibilitando avaliar a estabilidade da comunicação sob diferentes condições de ruído eletromagnético.

```
import socket
import subprocess
import time
import os
import datetime
import numpy as np

# Gerar nome do arquivo com data/hora
def gerar_nome_arquivo():
    data_hora = datetime.datetime.now().strftime("%Y-%m-%d_%Hh%Mmin%Ss")
    return f"log_freq_esp32_{data_hora}.txt"

# Obter IP local da m quina
def get_ip():
    s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
    try:
        s.connect(("10.255.255.255", 1))
        ip = s.getsockname()[0]
    except Exception:
        ip = "127.0.0.1"
    finally:
        s.close()
    return ip

# Verificar IPs ativos na rede local por meio de ping
def ping_sweep(network, start=1, end=20):
    ativos = []
    for i in range(start, end + 1):
        ip = f"{network}.{i}"
        try:
            result = subprocess.run(["ping", "-n", "1", "-w", "100", ip],
                                    stdout=subprocess.DEVNULL, stderr=
                                        subprocess.DEVNULL)
            if result.returncode == 0:
                ativos.append(ip)
        except:
            continue
```

```
37      return ativos
38
39  # Estabelecer conex o com o ESP32 via socket
40  def conectar_esp(porta=8080):
41      local_ip = get_ip()
42      rede = ".".join(local_ip.split('.')[:-1])
43      print(f" Buscando ESP32 na rede {rede}.x...")
44
45      ativos = ping_sweep(rede, 1, 20)
46
47      for ip in ativos:
48          try:
49              sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
50              sock.settimeout(1)
51              sock.connect((ip, porta))
52              print(f"    Conectado ao ESP32 em {ip}:{porta}")
53              sock.settimeout(2.0)
54              return sock
55          except:
56              continue
57
58      print(" Nenhum ESP32 encontrado.")
59      return None
60
61  # Coletar e registrar os dados enviados pelo ESP32
62  def analisar_e_salvar(sock, duracao_segundos=30):
63      print(f"    Coletando dados por {duracao_segundos} segundos...")
64      inicio = time.time()
65      dados_recebidos = []
66
67      nome_arquivo = gerar_nome_arquivo()
68      with open(nome_arquivo, 'w') as f:
69          f.write("ESP32_millis\tPC_timestamp_ms\n")
70
71          while time.time() - inicio < duracao_segundos:
72              try:
73                  dados = sock.recv(1024)
74                  if not dados:
75                      break
76                  partes = dados.decode().split(',')
77                  for p in partes:
78                      try:
79                          valor = int(p.strip())
80                          timestamp_local = int((time.time() - inicio) *
                                1000)
81                          dados_recebidos.append(valor)
82                          f.write(f"{valor}\t{timestamp_local}\n")
```

```
                    except:
                        continue
            except socket.timeout:
                continue
            except Exception as e:
                print("Erro:", e)
                break

    print(f"\n Dados salvos em: {nome_arquivo}")
    print(f" Total de millis recebidos: {len(dados_recebidos)}")

    if len(dados_recebidos) < 2:
        print(" Dados insuficientes para an lise.")
        return

# Execu  o principal
if __name__ == "__main__":
    sock = conectar_esp()
    if sock:
        try:
            analisar_e_salvar(sock, duracao_segundos=30)
        finally:
            sock.close()
```

Listing F.1 – Script Python para detecção, conexão e registro dos dados enviados pelo ESP32

APÊNDICE G – CÓDIGO DE PROCESSAMENTO E FILTRAGEM DO SINAL DE TORQUE

O código a seguir realiza a leitura de um arquivo CSV contendo dados de torque, aplica um filtro para remoção de outliers isolados com base no método IQR (intervalo interquartílico) e gera um novo arquivo CSV com os dados processados, além de um gráfico comparativo.

```
import pandas as pd
import os
import matplotlib.pyplot as plt
import numpy as np

# Caminho original do arquivo
caminho_arquivo = r"C:\Users\joaop\tcc2\teste_pratico\teste_4_padrao.csv"

#  L   o CSV
df = pd.read_csv(caminho_arquivo)
df.columns = ['torque', 'tempo_ms']

# Etapa 1: Detecta outliers pelo IQR
def detectar_outliers_iqr(serie):
    Q1 = serie.quantile(0.25)
    Q3 = serie.quantile(0.75)
    IQR = Q3 - Q1
    limite_inferior = Q1 - 1.5 * IQR
    limite_superior = Q3 + 1.5 * IQR
    return (serie < limite_inferior) | (serie > limite_superior)

# Etapa 2: Mant m apenas outliers isolados (n o pertencentes a grupo com
    >=3 pontos)
def filtrar_outliers_isolados(mask_outliers, max_grupo=2):
    mask_final = mask_outliers.copy()
    grupo = []

    for i, is_out in enumerate(mask_outliers):
        if is_out:
            grupo.append(i)
        else:
            if 0 < len(grupo) <= max_grupo:
                pass
            else:
                for idx in grupo:
                    mask_final[idx] = False
            grupo = []

    if 0 < len(grupo) <= max_grupo:
        pass
```

```
    else:
        for idx in grupo:
            mask_final[idx] = False

    return mask_final

# Detecta outliers isolados
mask_outliers = detectar_outliers_iqr(df['torque'])
mask_outliers_isolados = filtrar_outliers_isolados(mask_outliers)

# Dados filtrados (sem outliers isolados)
df_filtrado = df[~mask_outliers_isolados].copy().reset_index(drop=True)


# Salva CSV dos dados filtrados
pasta, nome_arquivo = os.path.split(caminho_arquivo)
nome_base, extensao = os.path.splitext(nome_arquivo)
novo_nome = f"{nome_base}_filtrado{extensao}"
caminho_novo_arquivo = os.path.join(pasta, novo_nome)
df_filtrado.to_csv(caminho_novo_arquivo, index=False)
print(f"Arquivo salvo em: {caminho_novo_arquivo}")

# Dados dos outliers isolados (originais)
df_outliers = df[mask_outliers_isolados]

# ====== GR FICO ======
plt.figure(figsize=(12,6))
plt.plot(df_filtrado['tempo_ms'], df_filtrado['torque'], label='Filtrado',
    color='blue', linewidth=1.5, alpha=0.8)
plt.scatter(df_outliers['tempo_ms'], df_outliers['torque'], label='Outliers
     Isolados', color='red', s=40)
plt.xlabel('Tempo (ms)')
plt.ylabel('Torque (N m)')
plt.title('Torque Filtrado e com Outliers Isolados Destacados')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

Listing G.1 – Código Python para filtragem de outliers e inversão do sinal de torque

APÊNDICE H – CÓDIGO DE COMPARAÇÃO ENTRE O SENSOR SEM FIO (ESP32) E O SENSOR COM FIO (HBM)

Este código realiza a leitura dos arquivos CSV gerados pelos dois sistemas de aquisição (com fio e sem fio), aplica um atraso manual aos dados do sensor ESP32 para alinhar os sinais no tempo, e gera um gráfico comparativo que possibilita avaliar visualmente a similaridade entre os dois sensores.

```
import pandas as pd                          # Importa a biblioteca pandas para manipula  o de dados em tabelas (DataFrames)
import matplotlib.pyplot as plt          # Importa matplotlib.pyplot para plotar gr ficos

# Caminhos dos arquivos CSV contendo os dados dos sensores
caminho_fio = r"C:\Users\joaop\tcc2\teste_pratico\util\teste_3_em_ms.csv"
caminho_esp = r"C:\Users\joaop\tcc2\teste_pratico\teste_3_padrao_corrigido_filtrado.csv"

# L   os arquivos CSV e cria DataFrames com os dados
df_fio = pd.read_csv(caminho_fio)      # Dados do sensor com fio (HBM)
df_esp = pd.read_csv(caminho_esp)      # Dados do sensor sem fio (ESP32)

# Renomeia as colunas para nomes mais f ceis de manipular e entender
df_fio.columns = ['torque_fio', 'tempo_ms_fio']    # Colunas: torque e tempo em ms do sensor com fio
df_esp.columns = ['torque_esp', 'tempo_ms_esp']  # Colunas: torque e tempo em ms do sensor sem fio

# Define um valor manual de delay em milissegundos para ajustar o eixo X dos dados sem fio
delay_ms = 0  # Pode ser positivo (atrasar os dados) ou negativo (adiantar)

# Aplica o delay ao vetor de tempo do sensor ESP32, deslocando os tempos
tempo_esp_ajustado = df_esp['tempo_ms_esp'] + delay_ms

# Cria a figura do gr fico com tamanho personalizado (largura=12, altura=6 polegadas)
plt.figure(figsize=(12, 6))

# Plota os dados do sensor com fio, usando tempo no eixo X e torque no eixo Y
plt.plot(df_fio['tempo_ms_fio'], df_fio['torque_fio'], label='Sensor com fio (HBM)', linewidth=2)

# Plota os dados do sensor sem fio, usando o tempo ajustado no eixo X e torque no eixo Y
# Usa transpar ncia alpha=0.8 para diferencia  o visual
```

```
plt.plot(tempo_esp_ajustado, df_esp['torque_esp'], label=f'Sensor sem fio (
    ESP32)', linewidth=2, alpha=0.8)

# Define o t tulo do gr fico
plt.title('Compara  o de Leitura de Torque: Sensor com Fio vs ESP32')

# Define o r tulo do eixo X
plt.xlabel('Tempo (ms)')

# Define o r tulo do eixo Y
plt.ylabel('Torque')

# Exibe a legenda, mostrando os nomes das linhas
plt.legend()

# Ativa a grade no gr fico para melhor visualiza  o dos valores
plt.grid(True)

# Ajusta o layout para que nada fique cortado na figura
plt.tight_layout()

# Exibe o gr fico na tela
plt.show()
```

Listing H.1 – Código Python para comparação entre os sinais do ESP32 e do sistema HBM

APÊNDICE I – CÓDIGO PYTHON PARA COMPARAÇÃO DOS SINAIS DE TORQUE COM E SEM FIO

O código a seguir realiza a comparação entre o sinal de torque adquirido com fio (sensor HBM) e o sinal obtido via sistema sem fio (ESP32), aplicando interpolação, cálculo de erro absoluto médio, erro máximo e correlação de Pearson entre os sinais.

```
import pandas as pd
import numpy as np
from scipy.stats import pearsonr
from scipy.interpolate import interp1d

# === Caminhos dos arquivos ===
caminho_fio = r"C:\Users\joaop\tcc2\teste_pratico\teste_2_em_ms.csv"
caminho_esp = r"C:\Users\joaop\tcc2\teste_pratico\
    teste_2_padrao_corrigido_filtrado.csv"

# === Leitura dos arquivos ===
df_fio = pd.read_csv(caminho_fio)
df_esp = pd.read_csv(caminho_esp)

# === Normaliza  o dos nomes das colunas ===
df_fio.columns = df_fio.columns.str.strip().str.lower()
df_esp.columns = df_esp.columns.str.strip().str.lower()

# === Verifica  o b sica ===
print(f"Amostras com fio: {len(df_fio)}")
print(f"Amostras sem fio: {len(df_esp)}")

tempo_total_fio = (df_fio['tempo(ms)'].iloc[-1] - df_fio['tempo(ms)'].iloc
    [0]) / 1000
tempo_total_esp = (df_esp['tempo_ms'].iloc[-1] - df_esp['tempo_ms'].iloc
    [0]) / 1000

print(f"Dura  o com fio: {tempo_total_fio:.2f} s")
print(f"Dura  o sem fio: {tempo_total_esp:.2f} s")

# === Interpola  o do sinal sem fio para os tempos do com fio ===
interp_func = interp1d(df_esp['tempo_ms'], df_esp['torque'], kind='linear',
     bounds_error=False, fill_value="extrapolate")
torque_interp_esp = interp_func(df_fio['tempo(ms)'])

# === Vetores de torque ===
torque_fio = df_fio['torque(n.m)'].values
torque_esp = torque_interp_esp

# === Erro absoluto e m tricas ===
```

```
erro_absoluto = np.abs(torque_fio - torque_esp)
mae = np.mean(erro_absoluto)
erro_max = np.max(erro_absoluto)
correlacao, _ = pearsonr(torque_fio, torque_esp)

# ===  ndice  do erro m ximo e tempo correspondente ===
indice_erro_max = np.argmax(erro_absoluto)
tempo_erro_max = df_fio['tempo(ms)'].iloc[indice_erro_max]

print(f"Tempo do erro m ximo: {tempo_erro_max} ms")
print(f"\nErro absoluto m dio (MAE): {mae:.3f} N.m")
print(f"Erro m ximo observado: {erro_max:.3f} N.m")
print(f"Correla  o de Pearson: {correlacao:.3f}")
```

Listing I.1 – Código para interpolação e análise estatística entre os sinais com fio e sem fio.