Universidade Federal de Ouro Preto
Instituto de Ciências Exatas e Aplicadas
Departamento de Engenharia Elétrica

Trabalho de Conclusão de Curso

Projeto de um Escâner 3D Aliado a um Sistema de Processamento Digital de Sinais para Identificação de Falhas em Estruturas

Vitor Hugo Mendes Gomes

João Monlevade, MG
2022

Vitor Hugo Mendes Gomes

Projeto de um Escâner 3D Aliado a um Sistema de Processamento Digital de Sinais para Identificação de Falhas em Estruturas

Trabalho de Conclusão de curso apresentado à Universidade Federal de Ouro Preto como parte dos requisitos para obtenção do Título de Bacharel em Engenharia Elétrica pelo Instituto de Ciências Exatas e Aplicadas da Universidade Federal de Ouro Preto.
Orientador: Prof. Glauco Ferreira Gazel Yared
Coorientador: Prof. Marcelo Moreira Tiago

Universidade Federal de Ouro Preto
João Monlevade
2022

SISBIN - SISTEMA DE BIBLIOTECAS E INFORMAÇÃO

G633p Gomes, Vitor Hugo Mendes.
Projeto de um escâner 3D aliado a um sistema de processamento digital de sinais para identificação de falhas em estruturas. [manuscrito] / Vitor Hugo Mendes Gomes. - 2022.
114 f.: il.: color., gráf., tab..

Orientador: Prof. Dr. Glauco Ferreira Yared.
Coorientador: Prof. Dr. Marcelo Moreira Tiago.
Monografia (Bacharelado). Universidade Federal de Ouro Preto. Instituto de Ciências Exatas e Aplicadas. Graduação em Engenharia Elétrica .

1. Engenharia elétrica. 2. Identificação - Falhas estruturais - Peças. 3. Localização de falhas (Engenharia). 4. Processamento de sinais - Técnicas digitais. I. Tiago, Marcelo Moreira. II. Yared, Glauco Ferreira. III. Universidade Federal de Ouro Preto. IV. Título.

CDU 621.3

Bibliotecário(a) Responsável: Flavia Reis - CRB6-2431

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE OURO PRETO
REITORIA
INSTITUTO DE CIENCIAS EXATAS E APLICADAS
DEPARTAMENTO DE ENGENHARIA ELETRICA

FOLHA DE APROVAÇÃO

Vitor Hugo Mendes Gomes

Projeto de um Escâner 3D Aliado a um Sistema de Processamento Digital de Sinais para Identificação de Falhas em Estruturas

Monografia apresentada ao Curso de Engenharia Elétrica da Universidade Federal
de Ouro Preto como requisito parcial para obtenção do título de Engenheiro Eletricista

Aprovada em 13 de abril de 2022

Membros da banca

Doutor - Glauco Ferreira Gazel Yared - Orientador - Universidade Federal de Ouro Preto
Doutor - Marcelo Moreira Tiago - Co-Orientador - Universidade Federal de Ouro Preto
Doutor - Juan Carlos Galvis Manso - Universidade Federal de Ouro Preto
Doutor - Renan Fernandes Bastos - Universidade Federal de Ouro Preto

Glauco Ferreira Gazel Yared, orientador do trabalho, aprovou a versão final e autorizou seu depósito na Biblioteca Digital de Trabalhos de Conclusão de Curso da UFOP em 15/06/2022

Documento assinado eletronicamente por **Glauco Ferreira Gazel Yared**, **PROFESSOR DE MAGISTERIO SUPERIOR**, em 15/06/2022, às 20:24, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site http://sei.ufop.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0 , informando o código verificador **0346369** e o código CRC **EE8F94CB**.

R. Diogo de Vasconcelos, 122, - Bairro Pilar Ouro Preto/MG, CEP 35400-000
Telefone: (31)3808-0818 - www.ufop.br

Agradecimentos

Agradeço, em primeiro lugar, a Deus, depois aos meus pais, Wagdo Braga Gomes e Maria Conceição. Não poderia esquecer dos meus irmãos Ana Lara e Wagdo Jr.

Minha sincera gratidão a meus avós, José Geraldo e Maria Aparecida pelo acolhimento. Agradeço igualmente minha tia e madrinha Janice, além das minhas maravilhosas primas.

Agradeço em especial ao professor Glauco pela orientação, amizade e apoio para a realização deste projeto. Tenho uma gratidão enorme a este professor com o qual aprendi muito desde o início do curso. Agradeço da mesma maneira ao professor Marcelo por se dispor a coorientar este trabalho e por estar sempre disposto a tirar dúvidas e por ter dado tantas ideias em relação ao desenvolvimento deste projeto. Meus mais sinceros cumprimentos à todos os demais professores que me acompanharam durante a graduação, sem vocês não chegaria onde cheguei. Gostaria de deixar um cumprimento especial aos professores Juan Galvis e Renan Bastos por terem aceitado avaliar esse trabalho.

Não poderia esquecer da minha companheira, Júlia, e dos meus amigos Guilherme e Vitor Martins; e também das demais amizades que fiz durante a graduação.

Os nomes mencionados aqui tem, de certa forma, uma contribuição para a elaboração deste trabalho. É por isso que ficam os meus mais sinceros agradecimentos à todos.

"O homem não teria alcançado o possível se, repetidas vezes, não tivesse tentado o impossível."

– Max Weber

Resumo

O escâner 3D é um equipamento capaz de varrer uma estrutura tridimensional e gerar dados que contenham informações sobre a geometria espacial desse objeto, e uma aplicação comum para este equipamento é o desenvolvimento de sistemas de identificação de falhas estruturais em peças e objetos. Sendo assim, foi proposto neste trabalho o desenvolvimento do projeto de um escâner 3D e a utilização desse equipamento para o levantamento de uma base de dados contendo peças metálicas defeituosas e saudáveis. Portanto, o projeto consiste em duas grandes partes: na elaboração e execução do projeto do equipamento, e no desenvolvimento das técnicas para identificação de falhas nas estruturas metálicas propostas. O escâner 3D funcionou como esperado e o sistema de identificação de falhas desempenhou com taxas de acerto na localização de peças defeituosas de mais de 85%, com a aplicação de métodos de extração de características no domínio espacial e classificador linear simples. Tal resultado reforçou a justificativa da construção do equipamento e reiterou a sua aplicação em sistemas de identificação de falhas em estruturas.
Palavras-chave: escâner 3D, técnicas de identificação de falhas.

Abstract

The 3D scanner is an equipment that can scann an object or a three-dimensional structure, and a common application for this equipment is the development of structural failure identification systems. Therefore, it was proposed in this work the development of the project of a 3D scanner and the use of this equipment to survey a database containing deficient and healthy metal parts. Therefore, the project consists of two major parts: the elaboration and execution of the equipment project, and the development of techniques to identify failures in the proposed metallic structures. The 3D scanner worked as expected and the fault identification system performed with hit rates in locating defective parts of more than 85%, with the application of feature extraction methods in the spatial domain and simple linear classifier. This result reinforced the reason for the construction of the equipment and reiterated its application in failure identification systems in structures.
Keywords: 3D scanner, failure identification systems.

Lista de ilustrações

Lista de Siglas

CPU	Unidade Central de Processamento, do inglês *Central Processing Unit*
DMA	Acesso Direto à Memória, do inglês *Direct Memory Acess*
FIR	*Finite Impulse Reponse*
GPIO	I/O Digital de Uso Geral, do inglês *General-Purpose Digital I/O*
I2C	Circuito Inter Integrado, do inglês *Inter Integrated Circuit*
ISR	Rotina de Serviço de Interrupção, do inglês *Interrupt Service Routine*
LIDAR	*Light Detection and Ranging*
NDT	Ensaios Não Destrutivos, do inglês *Non-Destructive Tests*
PCI	Placa de Circuito Impresso
PWM	Modulação por Largura de Pulso, do inglês *Pulse Width Modulation*
SHM	Monitoramento de Integridade Estrutural, do inglês *Structure Health Monitoring*
SPI	Interface Periférica Serial, do inglês *Serial Peripheral Interface*
UART	Receptor e Transmissor Assíncrono Universal, do inglês *Universal Asynchronous Receiver and Transmiter*
UC	Unidade de Controle
ULA	Unidade Lógica Aritmética

Sumário

1 Introdução

1.1 Contexto

Diante do grande avanço da tecnologia vivenciado nos dias atuais, o crescimento e desenvolvimento de técnicas de inspeção e reconhecimento de falhas em estruturas mecânicas e civis através de sensores não invasivos têm crescido expressivamente (MARINDRA; TIAN, 2008).

Tais técnicas são tradicionalmente conhecidas como Ensaios Não Destrutivos, do inglês *Non-Destructive Tests* (NDT) e são muito utilizadas nos sistemas de Monitoramento de Integridade Estrutural, do inglês *Structure Health Monitoring* (SHM), aumentando a confiabilidade e segurança de estruturas, como no caso de aeronaves, por exemplo (BO-LIN; BI-FENG; FEI, 2007).

De acordo também com Bo-lin, Bi-feng e Fei (2007), dentre as técnicas não invasivas para monitoramento e identificação de falhas, é possível citar os métodos por meio de ondas de ultrassom e a utilização de sensores piezoelétricos, por exemplo.

Nesse contexto, o escâner 3D com aquisição a laser pode ser utilizado como uma técnica alternativa às citadas acima para monitoramento e identificação de falhas em estruturas. Isso porque é possível realizar o mapeamento de objetos tridimensionais com este equipamento.

Um exemplo da utilização do escâner 3D para tais fins foi desenvolvido por Collins e Sitar (2004), onde o objetivo é desenvolver um sistema para estudo da estabilidade de taludes e áreas de encosta. Na figura 1 é possível visualizar o escâner 3D utilizado no trabalho mencionado.

Figura 1 – Escâner 3D baseado em aquisição a laser fazendo o mapeamento de um talude.

Fonte: Retirado de Collins e Sitar (2004).

Nesse contexto, o escâner 3D pode ser definido como um dispositivo que faz a varredura de objetos por meio de um sensor, como o laser, e codifica suas caraterísticas geométricas em sinais digitais. As informações contidas nesses sinais podem ser utilizadas, portanto, para identificação de falhas e monitoramento de estruturas, como já foi mencionado.

Tal fato é reiterado pelo trabalho de He et al. (2017), onde os autores utilizam o escâner 3D para implementar um sistema de identificação de falhas na planicidade de peças manufaturadas.

Outros tipos de aplicação e utilização do escâner 3D serão discutidas com mais detalhes na seção 1.4, que trata de uma revisão do estado da arte do escâner 3D na literatura.

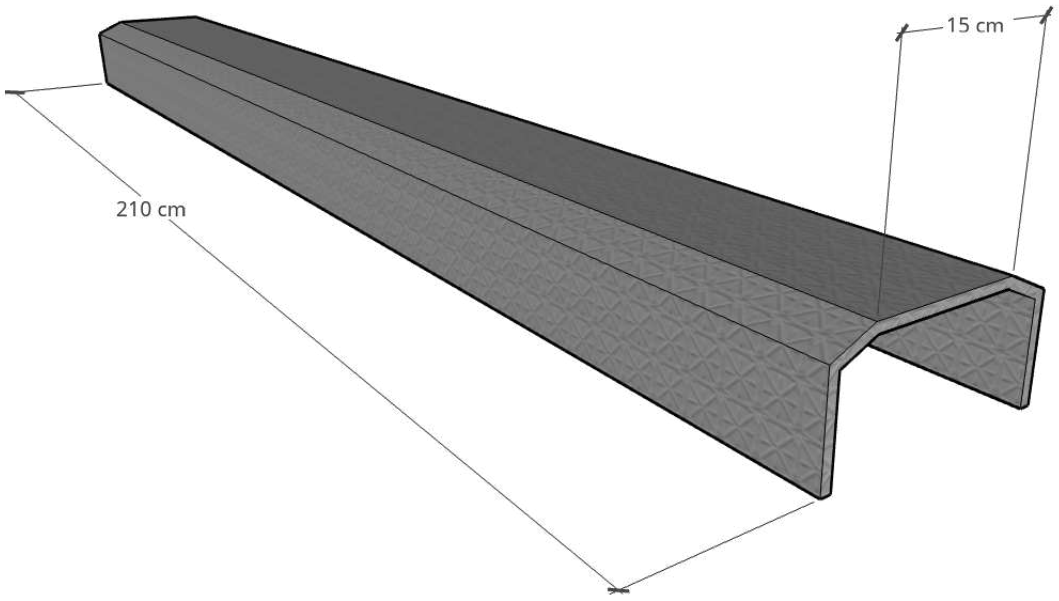
1.2 Justificativa

A partir do contexto mencionado, o desenvolvimento do projeto de um escâner 3D é o passo primordial para a criação de sistemas de identificação de falhas e monitoramento de estruturas e objetos.

A Figura 2 ilustra uma peça metálica que foi utilizada para estudo de caso deste trabalho. Nesse caso, o escâner 3D desenvolvido foi utilizado para identificação de falhas do tipo trinca em peças metálicas como da figura mencionada. Dessa forma, a correta identificação dessas falhas pode significar mais segurança e economia de recursos financeiros, devido ao contexto de aplicação dessas peças.

Dessa forma, a motivação para desenvolver o escâner 3D nesse trabalho foi conseguir projetar um sistema de aquisição de dados capaz de mapear objetos tridimensionais. Esses dados portam informações valiosas a respeito da condição estrutural de qualquer objeto escaneado, e, por isso outra motivação foi de utilizar esses dados dentro de um sistema de classificação capaz de dizer, por exemplo, se a estrutura escaneada tem defeito ou não. Em resumo, a justificativa para o desenvolvimento e projeto do escâner 3D, aliado à um sistema de processamento de dados, é de conseguir mapear essas peças metálicas mencionadas e identificar trincas nessas estruturas.

Figura 2 – Representação tridimensional de uma peça metálica que foi utilizada para estudo de caso do sistema proposto neste trabalho.



Fonte: Do Autor.

1.3 Objetivos

1.3.1 Objetivos Gerais

Este trabalho, portanto, tem como objetivo geral mostrar o desenvolvimento e projeto de um escâner 3D, com aquisição de distância por sensor laser, com o intuito de mapear e identificar falhas em peças metálicas. Para isso, foi montado uma base de dados com essas peças para identificar trincas nessas estruturas, se utilizando de algoritmos de classificação com extração de características espaciais e classificador linear.

1.3.2 Objetivos Específicos

Além disso, vale destacar que os objetivos específicos são:

- implementação do projeto elétrico e do sistema embarcado capaz de realizar todos os movimentos do escâner 3D e adquirir os dados captados pelo sensor laser;
- mostrar os testes realizados para o acerto do funcionamento do escâner;
- Levantamento de uma base de dados de peças metálicas, contendo objetos defeituosos e não defeituosos, através da varredura do escâner;
- realização do tratamento e pré-processamento dos dados adquiridos;

- implementação de classificação, com técnicas de treinamento supervisionado, e baseado num sistema de classificação linear e extração de características no domínio espacial, para a separação dos dados das peças defeituosas e saudáveis;
- apresentação da validação e desempenho do sistema de classificação;
- apresentação do teste de escaneamento, pelo escâner 3D, de uma estrutura humana.

Portanto, ao longo do texto será exposto toda a base teórica e as metodologias aplicadas para que os objetivos listados pudessem ser alcançados.

1.4 Revisão de Literatura

Nesta seção será feita uma revisão dos trabalhos já existentes na literatura que abordam sobre o tema escâner 3D com a intenção de revisar trabalhos que mostram diferentes tipos de aplicação deste dispositivo, desde aquelas para problemáticas gerais quanto aquelas aplicadas a problemas específicos. Além disso, foi feito também um apanhado sobre o mercado deste equipamento, e empresas que manufaturam o escâner 3D em maiores escalas.

1.4.1 Aplicações do Escâner 3D

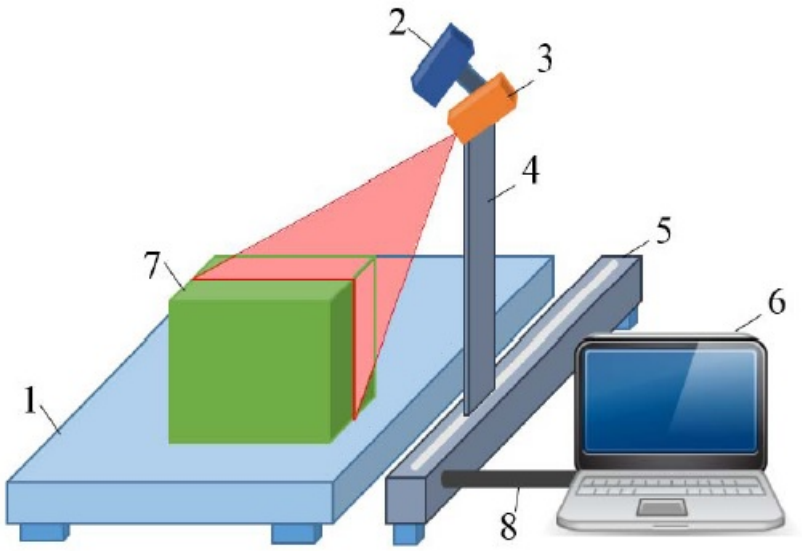
De acordo com Ghazali et al. (2011), um escâner 3D a laser pode ser definido como um dispositivo que é capaz de capturar dados tridimensionais de um objeto a partir da varredura do mesmo. Os escâneres utilizam estruturas mecânicas capazes de movimentar um sensor laser, por exemplo, para realizar as medições (MOLDER et al., 2014).

Dentre os tipos de escâneres, existem os que realizam o escaneamento por contato e os que não (MOSTAFA; EBRAHIM, 2015). No primeiro caso, geralmente são usados pontas que, em contato com a superfície, captam os valores dos pontos espaciais (x, y, z). Já no segundo caso, são utilizados instrumentos óticos capazes de realizar essas medições sem entrar em contato com a superfície (DANESHMAND et al., 2018).

O escâner a laser é um exemplo, onde o sensor laser pode ser projetado sobre o objeto de forma pontual, ou em linha, mas sem contato (PAWAR et al., 2017). De acordo com Abd-Raheem, AlDeiri e Alyaman (2018) uma possível configuração para o escâner 3D é que o sensor fique fixo em ponto enquanto o objeto passe através do sensor. Entretanto, as configurações em que o sensor se move e o objeto fica fixo são comuns na literatura (Yunardi; Imandiri, 2018; Lavelle; Schuet; Schuet, 2004; UKIDA et al., 2010). A Figura 3 ilustra um arranjo de escâner 3D com sensor em linha proposto por Andrushchak, Neznaradko e Hnatyuk (2017).

Sendo assim, em resumo o escâner 3D com aquisição de distância a laser é capaz de varrer um objeto e fazer medidas dos pontos espaciais (x, y, z) de onde o laser incide, e

Figura 3 – Esquemático de um escâner 3D: 1 é a plataforma de apoio do objeto, 2 é a câmera, 3 é o laser, 4 é o apoio da câmera e do laser, 5 é o mecanismo de escaneamento, 6 é o computador que faz o processamento das imagens, 7 é o objeto a ser escaneado e 8 são os cabos de conexão.



Fonte: Retirado de Andrushchak, Neznaradko e Hnatyuk (2017).

esse conjunto de pontos podem ser utilizados para fazer a reconstrução da imagem desse objeto, por exemplo (ABD-RAHEEM; ALDEIRI; ALYAMAN, 2018).

1.4.2 Aplicações do Escâner 3D

Já no início deste trabalho, na seção 1.1, foi apresentado o trabalho desenvolvido por Collins e Sitar (2004), onde o escâner 3D é utilizado para monitorar locais de encostas e taludes, a partir do mapeamento geográfico dessas estruturas. Outro exemplo similar é observado em Armesto et al. (2009), em que os autores propõem o escaneamento da geometria de rochas de granito para fins de análise de estabilidade.

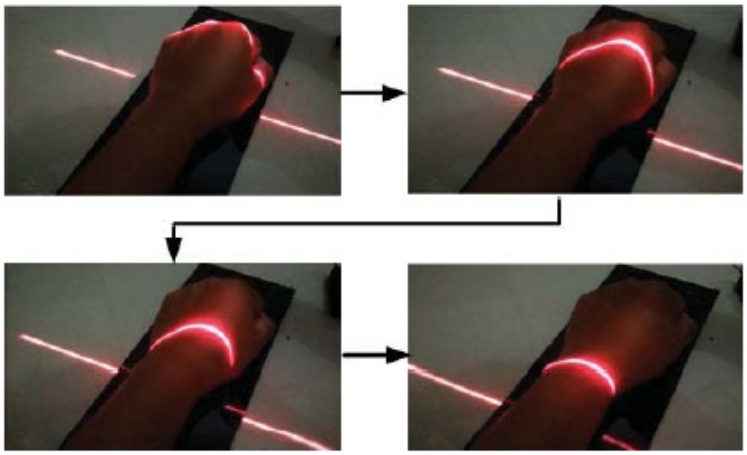
Na industria, o escâner 3D é utilizado para checar a qualidade do produto final e encontrar possíveis defeitos de fabricação em objetos. A tecnologia de escaneamento pode ser útil, portanto, para produtos feitos de materiais frágeis ou macios, tal como a borracha, onde visualmente é difícil encontrar defeitos (YAO, 2005). Mas o escâner 3D pode ser aplicado também a objetos e estruturas metálicas, embora seja menos comum.

Indo na mesma direção das aplicações em industrias, em He et al. (2017) os autores abordam técnicas de monitoramento da qualidade de produtos manufaturados a partir do escaneamento tridimensional. Um exemplo interessante e relevante pode ser observado em Lavelle, Schuet e Schuet (2004). A proposta do trabalho em questão é desenvolver um escâner capaz de realizar o escaneamento em alta velocidade e com alta acurácia. Este

trabalho foi apresentado para fazer parte do programa *NASA Mars roverprogram*, como parte de um sistema de inspeção de proteção térmica.

Dentro da área biomédica e biomecânica, no trabalho desenvolvido por Yunardi e Imandiri (2018), o escâner 3D é utilizado para mapear a região do pulso de um indivíduo pela técnica de triangulação a laser e criar uma imagem gráfica desta parte do corpo. Com essa imagem, o sistema proposto consegue imprimir uma espécie de cinta ortopédica para ajudar na recuperação de lesões no pulso. Os autores mostram através dos resultados obtidos que é possível utilizar o escâner para obter um modelo 3D e imprimir o protótipo desejado através de uma impressora 3D. A Figura 4 mostra como é feito o processo de escaneamento do pulso do indivíduo através do escâner 3D a laser.

Figura 4 – Modelo do pulso de um ser humano sendo escaneado através do método de triangulação a laser.



Fonte: Retirado de Yunardi e Imandiri (2018).

Dentro dessa área de biomédica e biomecânica, o escâner 3D pode ser utilizado para recriar imagens de partes do corpo humano com a intenção de analisar algum padrão de patologia ou condição. Ou mesmo criar moldes de talas para serem impressos em impressoras 3D, como já foi mostrado em Yunardi e Imandiri (2018).

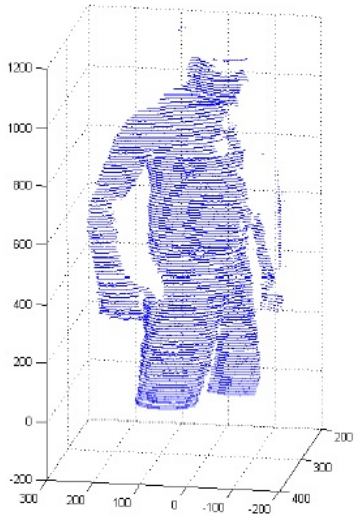
Outro exemplo interessante ainda dentro dessa área foi desenvolvido por Gazziro (2005). A partir da técnica de aquisição por triangulação a laser e técnicas de processamento de imagem, o autor propõe a criação de um escâner 3D antropométrico. A antropometria se ocupa em analisar medidas de partes do corpo humano e comparar com dados genéticos e biológicos desse mesmo indivíduo e compará-los entre si. O trabalho de Gazziro (2005) utiliza o escâner 3D para estudar os aspectos antropométricos do corpo humano. A Figura 5 mostra a cabine em que os sensores laser estão colocados e o indivíduo é posicionado para aquisição de dados.

Figura 5 – Exemplo da cabine de varredura para o escâner 3D desenvolvida por Gazziro (2005).

Fonte: Retirado de Gazziro (2005).

A Figura 6 mostra o resultado da varredura de um indivíduo feita na cabine mostrada na figura 5.

Figura 6 – Nuvem de pontos de um corpo real capturados pela estrutura mostrada na figura 5.



Fonte: Retirado de Gazziro (2005).

Outros trabalhos focam mais na montagem da estrutura do escâner, visando a economia e otimização dos componentes para apresentar um dispositivo mais barato e acessível, como é o caso de Sukvichai et al. (2018). O trabalho em questão pretende construir um escâner de baixo custo para ajudar na confecção de peças ortopédicas para pessoas mais pobres.

Dado o exposto, fica claro que as aplicações do escâner 3D são bem variadas e cada autor tem a liberdade e criatividade de utilizar esta ferramenta de acordo com o problema que deseja solucionar e a tecnologia que tem disponível em mãos.

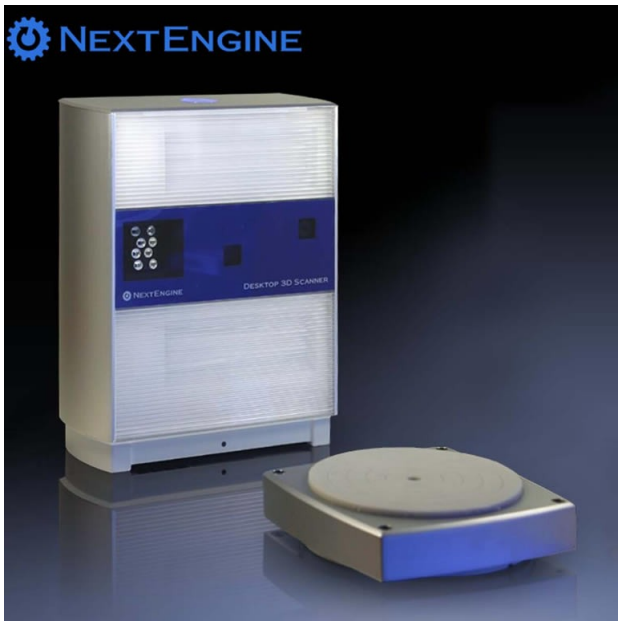
1.4.3 O Escâner 3D no Mercado

As tecnologias de escâner 3D são manufaturadas por algumas empresas especializadas e esses produtos são adquiridos para aplicação nas diversas áreas mencionadas na seção 1.4.2.

Existem no mercado empresas especializadas no comércio desse tipo de tecnologia, tanto em larga, média e pequena escala, como é o caso da *NextEngine Inc.* Trata-se de uma empresa de capital privada fundada no ano 2000, especializada no desenvolvimento de escâner 3D a laser. Essa empresa conta com clientes como *HP*, *Kodak*, *Seiko*, *Motorola* e *Symbol Technologies* (NEXTENGINE, 2015).

A Figura 7 mostra um escâner a laser da *NextEngine*, que chega a custar cerca de $3000.

Figura 7 – Escâner 3D da *NextEngine*, em que o objeto escaneado gira uma volta completa no objeto circular.



Fonte: Retirado de NextEngine (2015).

No trabalho desenvolvido por Mathys, Brecko e Semal (2013) é mostrada uma comparação entre as principais tecnologias já existentes no mercado. O objetivo do trabalho em questão é utilizar essas diferentes tecnologias para escanear um crânio humano pertencente a coleção do Instituto Real Belga de Ciências Naturais.

De acordo com os autores, o escâner 3D é bastante útil em museus pois a digitalização 3D permite gravar coleções, reduz a necessidade de manipulação física, especialmente para objetos frágeis e fornece um *backup* em caso de perda ou destruição. Além disso, é possível que espécimes digitalizados sejam compartilhadas com outros cientistas em todo o mundo. As técnicas de imagem também permitem a produção de modelos 3D texturizados ou não texturizados com detalhes finos da superfície.

O trabalho de Mathys, Brecko e Semal (2013) faz o comparativo das empresas que desenvolvem escâner 3D, baseado no tipo de tecnologia de aquisição utilizada, a precisão de medição, e o preço de comercialização. O intuito deles é conseguir escolher qual tecnologia de escâner traz o melhor custo benefício para aplicação no museu mencionado.

De acordo também com Mathys, Brecko e Semal (2013) a *NextEngine* e a *FARO Scan Arm* são as empresas de escâner a laser que são bem conhecidas no mercado.

A Figura 8 mostra um escâner da *FARO Scan Arm*, que escaneia o objeto através de um braço mecânico. Esse braço é movimentado manualmente. O preço varia de $40.000 a $85000 e esse escâner é bastante utilizado na industria.

Figura 8 – Escâner 3D da *FARO Scan Arm*.

Fonte: Retirado de FARO (2016).

1.4.4 Análise dos Dados Adquiridos pelo Escâner

Os dados de um objeto varrido pelo escâner 3D são sinais que contém informações uteis a respeito da geometria desse objeto.

Em Axelsson (1999), é proposto um estudo sobre técnicas e algoritmos capazes de processar esses dados escaneados. O objeto de estudo do trabalho é voltado para o processamento digital de imagens escaneadas de espaços geográficos.

Em certa parte, em Axelsson (1999) a proposta é parecida com a deste trabalho, uma vez que o objetivo é desenvolver o escâner para fazer o processamento e classificação dos dados adquiridos.

Já em Andrushchak, Vasylyshyn e Chornenkyy (2015) é feita uma análise comparativa de algoritmos para identificação de projeções de linhas de laser. Além do estudo de técnicas de reconhecimento de padrões aplicadas aos sinais adquiridos pelo escâner 3D.

Dessa forma, esses dois trabalhos mencionados podem ser uteis para o desenvolvimento deste projeto, uma vez que algumas das técnicas abordadas nesses trabalhos podem ser aproveitadas. Outras técnicas para tratamento e classificação dos dados do escâner 3D serão mostradas posteriormente.

1.5 Estrutura do Trabalho

Neste capítulo foi feita uma revisão de literatura a respeito do estado da arte do escâner 3D e suas principais aplicações, além de mostrar a justificativa e os objetivos que motivaram o desenvolvimento deste trabalho.

No capítulo 2 é feito uma apanhado geral de toda base teórica utilizada para a execução deste projeto. É mostrado os conceitos dos sistemas de aquisição de dados e das técnicas de pré-processamento. Além disso é exposto os métodos utilizados para implementar um sistema de classificação, e por fim, é apresentado o conceito e o design de um sistema embarcado.

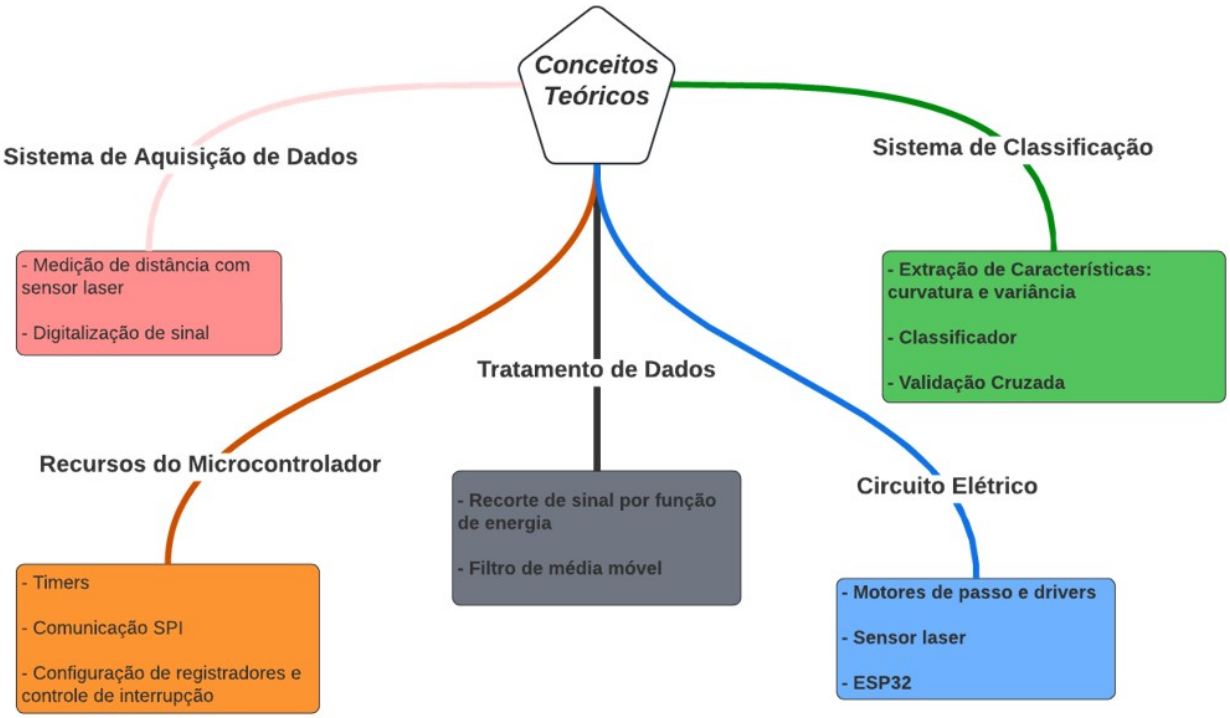
Se tratando da metodologia do trabalho, no capítulo 3 é mostrado todas as especificações do escâner, além dos detalhes do projeto elétrico e do sistema embarcado. Neste mesmo capítulo é abordado os testes feitos no escâner, bem como as metodologias utilizadas no levantamento da base de dados.

Por fim, no capitulo 4 é mostrado o desempenho do sistema de classificação proposto para a base de dados levantada e um teste feito para o escaneamento de um ser humano. Por fim no capítulo 5 de conclusões são feitas as considerações finais com uma reflexão sobre o cumprimento dos objetivos do trabalho e apresentação de ideias para trabalhos futuros.

2 Revisão Teórica

Neste capítulo é exposta toda a base teórica utilizada ao longo do desenvolvimento do projeto do escâner 3D e do sistema de identificação de falhas. São mostradas todas as técnicas de aquisição e processamento de dados utilizadas, bem como as técnicas de design de um sistema de classificação, além da teoria sobre sistemas embarcados e circuitos elétricos. A Figura 9 mostra um mapa conceitual de todas as bases teóricas abordadas neste capítulo.

Figura 9 – Mapa conceitual das teorias abordadas.

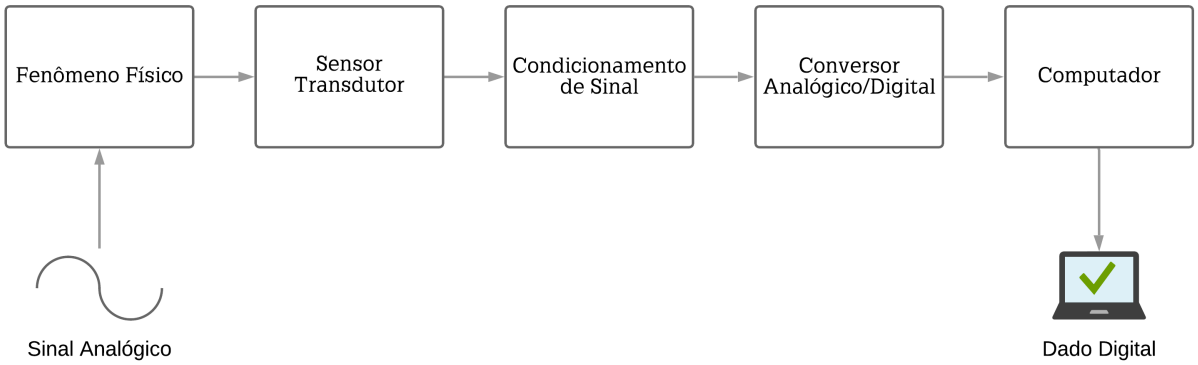


Fonte: Do autor.

2.1 Sistema de Aquisição de Dados

Um sistema de aquisição de dados é definido como uma ferramenta que consegue traduzir em medidas algum fenômeno físico qualquer, como uma medição de temperatura, por exemplo. Em geral, qualquer evento pode ser representado por um conjunto de dados, e a organização desses dados em um sistema computacional capaz de manipulá-los é conhecido como um sistema de aquisição (THEODORIDIS; KOUTROUMBAS, 2009). A Figura 10 mostra um diagrama típico de um sistema de aquisição de dados.

Figura 10 – Sistema de aquisição de dados.



Fonte: Do autor.

2.1.1 Técnica de Triangulação para Medição de Distância com Sensor Laser

Os sensores podem ser definidos como dispositivos que detectam e medem propriedades físicas, e fazem a indicação ou registro dessa medição. Em outras palavras, o mecanismo do sensor é projetado para responder a um estímulo físico ou químico de maneira que esse estímulo seja medido e monitorado. Por outro lado, os chamados transdutores transformam esse estímulo captado pelo sensor em um sinal elétrico (BEGA et al., 2011).

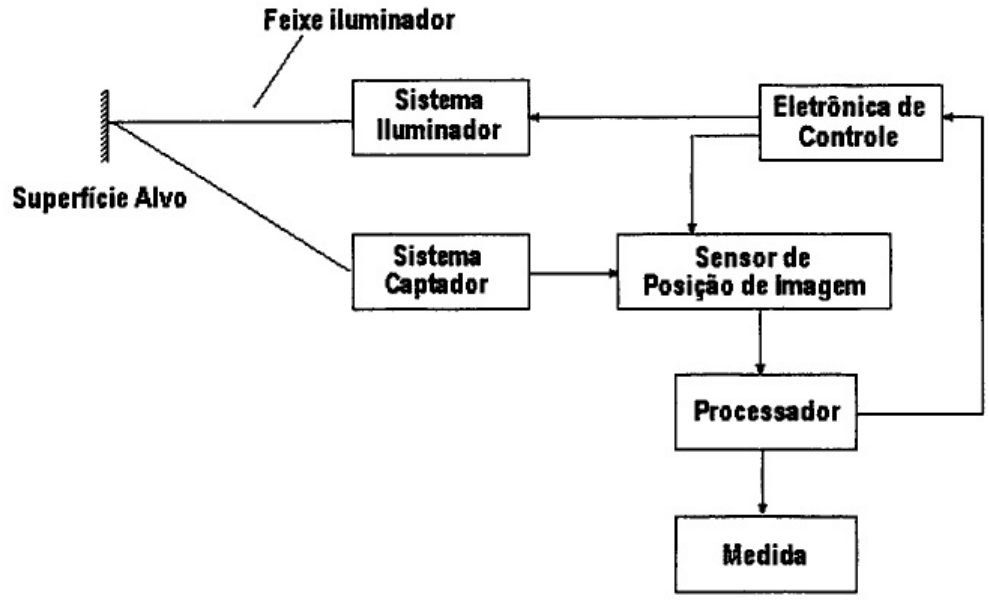
Os sensores e transdutores, portanto, são dispositivos de conversão de energia, convertem temperatura em corrente elétrica, por exemplo. Para os sistemas de aquisição de dados, independentemente do tipo de variável medida na entrada, a saída tem que ser necessariamente elétrica, por ser uma grandeza mais fácil de quantificar e de comunicar com um sistema computacional.

Nesse contexto, os sensores óticos se utilizam das propriedades da luz e da eletrônica para aferir medições em processos sem a necessidade de contato físico. Existem outros métodos de medição sem contato, mas os meios óticos apresentam certas vantagens interessantes em relação aos outros, como o índice de radiação baixo em relação a um raio-X, por exemplo (DOEBELIN, 1975).

As técnicas a laser permitem que as medições de distância sejam feitas diretamente entre o objeto e o instrumento através do feixe luz, garantindo assim uma diferente resolução de escaneamento e um maior ângulo de cobertura (GHAZALI et al., 2011).

A técnica de medição por triangulação a laser é baseada no princípio de reconhecer um feixe de luz laser que incide sobre um padrão de luz projetado sobre o objeto alvo da medição. Essa técnica é conhecida também como proxímetro. A Figura 11 mostra um diagrama de blocos esquemático dessa técnica.

Figura 11 – Esquemático de um sistema de medição de distância baseado no método de Triangulação a Laser.



Fonte: Retirado de Stefani (1995).

2.1.2 Digitalização do Sinal

A digitalização de um sinal tem por objetivo transformar uma medida analógica em uma medida correspondente, para que os sistemas computacionais sejam capazes de processar esses sinais. Os sensores captam grandezas analógicas e digitalizar significa transformar essa medida em um sinal digital (BEGA et al., 2011).

De acordo com Lee e Seshia (2013), um sinal analógico x é definido matematicamente como uma função que varia continuamente nos domínios do tempo e da amplitude, ou seja $x : R \rightarrow R$, onde o domínio da função x é o tempo e contradomínio é a amplitude.

O processo de quantização serve para trazer a amplitude para um contradomínio fechado binário, ou seja $q : R \rightarrow \{0, 1\}$, onde o valor da função quantizada q assume esses valores baseado em algum limiar. Em outras palavras, q é uma aproximação de x.

Obviamente que a aproximação de algum valor de x em apenas um bit 0 ou 1 é uma aproximação bem grosseira. Entretanto, como será visto posteriormente, se forem utilizados mais bits para fazer essa representação, a aproximação fica mais fiel ao valor analógico, e a quantização mais exata.

Ainda assim, a função q continua contínua no tempo. O processo de amostragem serve para trazer o domínio do tempo para valores discretos. Ou seja, o sinal amostrado e quantizado y pode ser definido por $y : Z \rightarrow \{0, 1\}$.

A equação (2.1) mostra a representação de um sinal y que é digital, por ser discreto tanto no tempo quanto na amplitude:

$$y(n) = q(nT), \tag{2.1}$$

onde todo $n \in Z$, e T é o período de amostragem.

De forma mais detalhada e com uma abordagem mais voltada para o sinal que para a matemática, Oppenheim, Willsky e Young (1983) mostram algumas particularidades do teorema de amostragem. Um sinal de tempo contínuo $x(t)$ de banda limitada pode ser determinado por suas amostras $x(nT)$, se

$$\omega_s > 2\omega_M, \tag{2.2}$$

em que,

$$\omega_s = \frac{2\pi}{T}. \tag{2.3}$$

A frequência ω_M é a maior componente de frequência presente no sinal a ser amostrado e ω_s é a frequência de amostragem. A equação (2.2) mostra que a frequência de amostragem deve ser pelo menos duas vezes maior que a maior componente de frequência do sinal, e o valor ω_M é conhecido como taxa de Nyquist.

Durante a amostragem é preciso tomar cuidado para evitar o *aliasing*, que é a sobreposição de réplicas do espetro do sinal amostrado quando a taxa de Nyquist não é respeitada (2.2). Assim, para evitar que o sinal se torne irrecuperável quando amostrado, ele deve ser necessariamente de banda limitada em frequências que estejam abaixo da metade da taxa de Nyquist. Por isso a importância da escolha de uma frequência de amostragem ω_s adequada (OPPENHEIM, 2011).

Já o Conversor Analógico-Digital é o dispositivo responsável por fazer a conversão do sinal e ele utiliza essas técnicas de quantização e amostragem. Existem dois parâmetros essenciais que caracterizam o conversor A/D, o período de amostragem T e o número de bits b.

São três etapas básicas de conversão: amostragem, quantização e codificação. As duas primeiras já foram explicadas anteriormente, e a codificação rotula os níveis de digitalização em um código binário correspondente.

Matematicamente, 2^b níveis de digitalização podem ser codificados em um código de b bits. Um conversor A/D de 3 bits, por exemplo, contém 8 níveis de digitalização.

Para um determinado valor de bits b, existem 2^b valores possíveis para representar o sinal analógico $x(t)$. Assim, ter um valor maior para b resulta em maior aproximação do sinal analógico. A resolução em bits de um conversor analógico, portanto, é dada por:

$$R_{bits} = 2^b - 1, \tag{2.4}$$

o que representa que o sinal analógico $x(t)$ é codificado por uma escala de números inteiros entre 0 e $2^b - 1$.

Além disso, à medida que T diminui, a quantidade de detalhe temporal do sinal que é preservado em sua representação digital aumenta. Na prática, quanto maior for b, mais difícil será tornar T pequeno. Assim, conversores de alta precisão (b alto) tendem a suportar taxas de amostragem mais lentas (T menor). A escolha de b e T representa uma compensação entre custo e precisão (LEE; SESHIA, 2013).

2.2 Técnicas de Pré-Processamento e Tratamento de Dados

Uma vez que os sinais estão adquiridos e digitalizados, é necessário aplicar técnicas de pré-processamento para poder eliminar erros e ruídos de aquisição que esses dados possam ter sofrido. Isso vai desde a eliminação de dados desnecessários, até a filtragem de componentes indesejadas dos sinais. Essa seção discute algumas dessas técnicas de pré-processamento.

2.2.1 Recorte de Sinal por Função de Energia

De acordo com Ciciora et al. (2004) tradicionalmente, no processamento de sinais, a energia de um sinal é definida como o quadrado da magnitude do sinal, ou a integral da magnitude do sinal ao quadrado. Matematicamente, no domínio de tempo discreto, a energia de um sinal é definida como:

$$E = \sum_{n=-\infty}^{\infty} |x(n)|^2. \tag{2.5}$$

Um sinal de energia só faz sentido se a energia total do sinal for finita, e a condição necessária para isso é que a amplitude do sinal se aproxime de zero à medida que o tempo se aproxima do infinito ($x(n) \to 0$ quando $n \to \pm\infty$). Para sinais que tenham energia infinita a potência média faz mais sentido (OPPENHEIM, 2011).

Então é possível reescrever a equação (2.5) para calcular a potência média do sinal dentro de um ciclo temporal. A potência média é calculada de acordo com a equação (2.6):

$$E(i) = \frac{1}{N} \sum_{n=0}^{N-1} |x_i(n)|^2. \tag{2.6}$$

Na equação (2.6) $x(n)$ é segmentado em uma janela temporal de tamanho N, e a potência média dentro dessa janela é equivalente a energia do sinal dentro dessa mesma janela temporal. Dessa forma, para cada valor de amplitude $x_i(n)$ há um valor de energia associado $E(i)$, que foi calculado dentro de uma janela temporal ao longo do sinal $x(n)$. Em outras palavras, dentro dessa janela, a potência média do sinal é equivalente à sua energia.

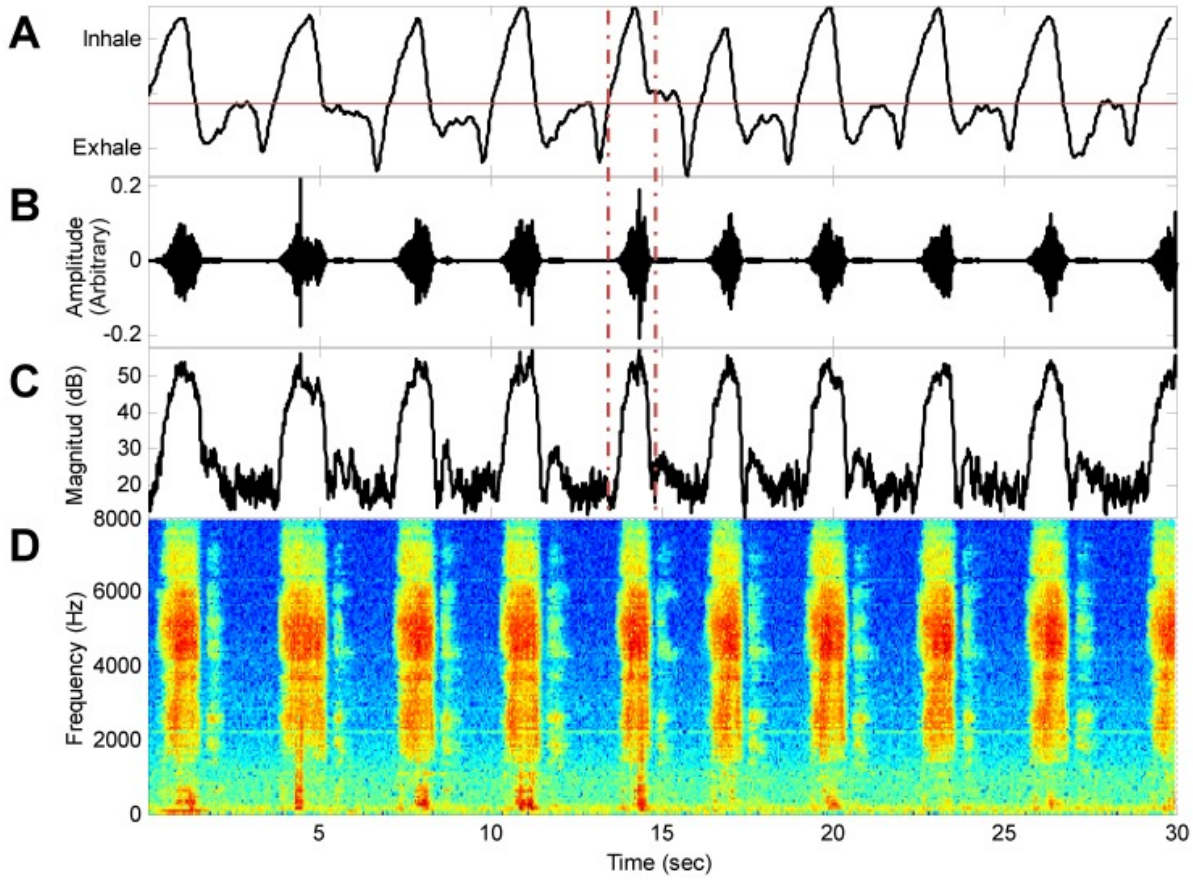
Segmentar o sinal em janelas temporais menores é uma técnica para garantir que as condições necessárias para que o cálculo seja feito sejam atendidas.

Portanto, a função de energia é uma ferramenta essencial para o pré-processamento dos dados, pois é através dela que é possível eliminar trechos do sinal que tenha energia nula e realizar recortes temporais.

Um bom exemplo da utilização das técnicas de energia em sinais são os sinais de áudio. Esses sinais têm um forte conteúdo de energia nas baixas frequências e um conteúdo de energia mais fraco nas altas frequências. Dessa forma a energia do sinal seria uma boa técnica para separar os diferentes espectros de frequência do sinal (AL-SHOSHAN, 2021).

A Figura 12 mostra um estudo em que os sons emitidos durante a respiração de um ser humano foram captados e foi utilizado o seu sinal de energia para detectar os momentos de inspiração. No caso desse estudo, os momentos de inspiração apresentam espectro de energia maiores pois se trata da respiração de indivíduos que têm problemas respiratórios de apneia obstrutiva do sono. Então o sinal de energia detecta os pontos da respiração onde a pessoa apresenta esses distúrbios.

Figura 12 – Em A - ciclo respiratório do indivíduo. B - sinal de áudio da respiração. C - sinal de energia. D - espectrograma.



Fonte: Retirado de Dafna, Tarasiuk e Zigel (2013).

2.2.2 Filtragem de Sinal com Filtro de Média Móvel

As mais diversas técnicas de filtragem de sinais pode ser feita através dos filtros digitais, pelo fato dos sinais fazerem parte de um sistema computacional. Esse filtros são implementados com o intuito de eliminar outros artefatos que possam interferir no sinal, como interferências de alta frequência ou componentes médias do sinal.

Para implementar tais filtros pode-se usar, por exemplo, uma série de ferramentas computacionais como o *Toolbox* do Matlab que trata de design de filtros digitais (MATHWORKS, 2019).

O filtro média móvel, por exemplo, é um filtro implementado através da convolução e é bastante utilizado quando se deseja suavizar o sinal em questão. De acordo com Oppenheim (2011), a equação geral de um filtro média móvel é dada por (2.7):

$$y[n] = \frac{1}{M_1 + M_2 + 1} \sum_{k=-M_1}^{M_2} x[n-k]. \tag{2.7}$$

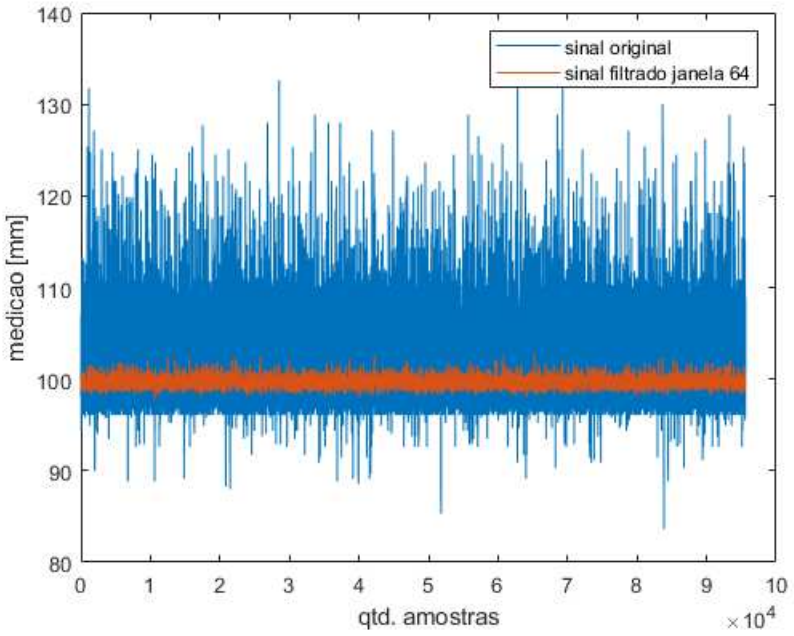
Nesse caso, a saída do filtro $y[n]$ é computada a partir do sinal de entrada $x[n]$. Esse filtro calcula a enésima amostra da sequência de saída como a média das $(M_1 + M_2 + 1)$ amostras da sequência de entrada em torno da enésima amostra da sequência de entrada. Em outras palavras $(M_1 + M_2 + 1)$ representa o tamanho da janela temporal em que a média é calculada, e que essa janela é deslizante ao longo do sinal.

O filtro de média não causal é um subtipo de um filtro de média móvel. Portanto, ele é não causal quando a média é feita com amostras passadas, presente e futuras; caso que ocorre bastante em sinais que dependem de coordenadas de posição, como imagens.

A Figura 13 mostra um sinal que deveria representar o perfil de uma estrutura plana, mas os ruídos de aquisição faz com que o sinal não seja coerente à sua representação. Dessa forma, ao aplicar um filtro de média não causal, é possível observar uma suavização no sinal, e que, com o ajuste correto do tamanho da janela espacial, o sinal se aproxima mais de uma curva plana. Na Figura 14 é possível observar a função de densidade de probabilidade do sinal original e do sinal filtrado. O sinal filtrado apresenta um desvio padrão muito menor justamente por conta da aplicação do filtro de média.

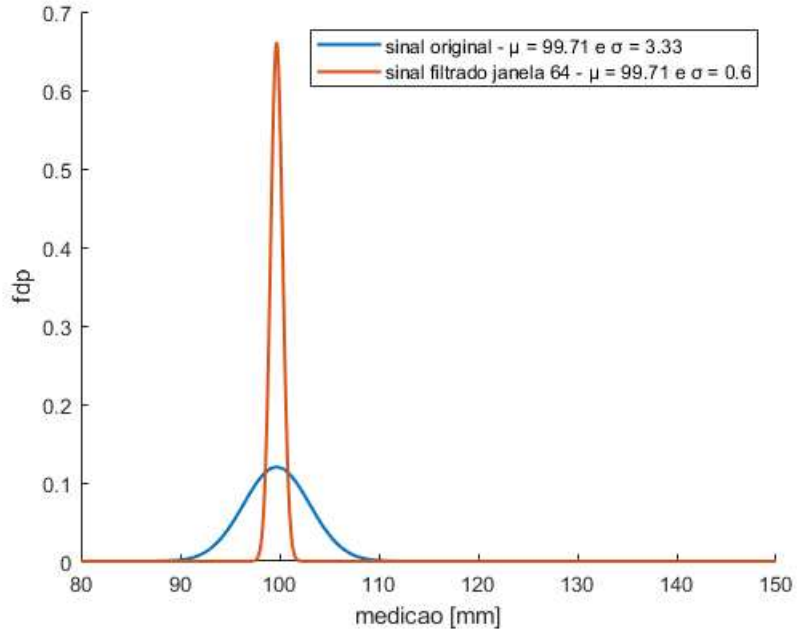
É perceptível que a aplicação de um filtro de média não causal consegue suavizar o sinal e concentrar as medições mais próximos de um valor médio. Considerando que se tratava de uma medição de uma superfície plana, era esperado que os valores se concentrassem mais próximos de um valor central.

Figura 13 – Sinal medido de uma curva plana e depois filtrado com um filtro de média não causal de janela 64 amostras.



Fonte: Do autor.

Figura 14 – Função de densidade de probabilidade para o sinal original e para o sinal filtrado.



Fonte: Do autor.

2.3 Sistema de Classificação

O objetivo desta seção é fazer um embasamento teórico de algumas das técnicas de reconhecimento de padrões utilizadas no *design* do sistema de classificação. A intenção é futuramente utilizar essas técnicas para fazer a identificação de falhas em objetos escaneados pelo escâner desenvolvido.

Exemplos de utilização de técnicas de reconhecimento de padrão podem ser observadas em Gutkin et al. (2011), Aretakis, Mathioudakis e Stamatis (2004). Em ambos os trabalhos os autores propõem diferentes tipos de técnicas para reconhecimento de falhas em equipamentos, algumas das quais serão citadas neste trabalho, como os classificadores lineares por exemplo.

Por trás de um sistema de classificação existem as técnicas de reconhecimento de padrões, que têm o objetivo de classificar objetos em uma quantidade de categorias ou classes. Dependendo das aplicações, esses objetos podem ser imagens, formas de onda, ou qualquer outro sinal que precisa ser classificado (THEODORIDIS; KOUTROUMBAS, 2009). As classes podem ser pré-definidas como um dado de entrada do sistema de classificação ou podem ser criadas e aprendidas pelos algoritmos por meio de similaridade entre os padrões.

De forma mais breve, o reconhecimento de padrões analisa um conjunto de dados de treinamento com o intuito de prever o comportamento dos dados de teste, e assim realizar uma tarefa de aprendizagem no sistema de classificação (RAMOS, 2014). Dessa maneira, com o sistema treinado, é possível classificar quaisquer dados de entrada através das técnicas de reconhecimento de padrões.

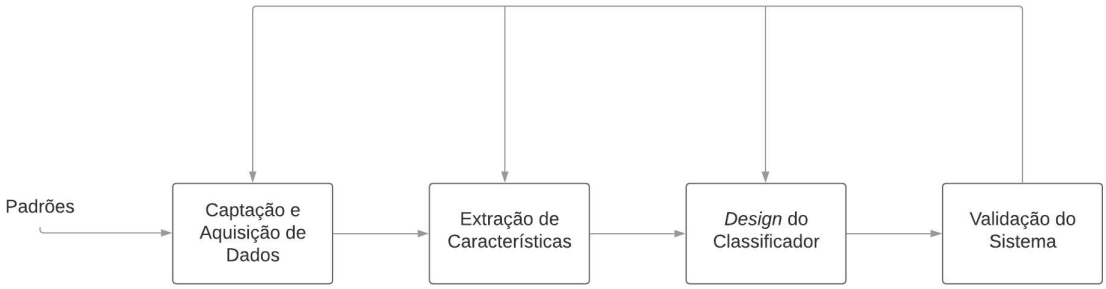
Diante do advento da automação na indústria e a evolução das tecnologias computacionais, observa-se também a necessidade de manipulação e classificação de dados. Essa tendência tornou o reconhecimento de padrões uma importante ferramenta para a realização de pesquisas em diversas áreas e para o desenvolvimento de técnicas que lidem com a classificação de dados. A Figura 15 mostra os estágios envolvidos no *design* de um sistema de classificação.

Dada a importância dos sistemas de classificação e a variada gama de aplicações das técnicas de reconhecimento de padrões, esta seção pretende explicar e explorar cada um dos estágios mostrados na Figura 15.

2.3.1 Técnicas de Extração de Características

Partindo da premissa que os dados foram obtidos e pré-processados da maneira correta, o sistema de classificação tem agora a tarefa de gerar e selecionar características desses dados. É admitido também, inicialmente, que alguns desses dados estejam rotulados, de maneira que seja possível dividi-los em classes.

Figura 15 – Os estágios básicos do design de um Sistema de Classificação. Da esquerda para a direita: entrada dos padrões, sensor, extração e seleção de características, design do classificador e avaliação do sistema.



Fonte: Adaptado de Theodoridis e Koutroumbas (2009).

Um exemplo interessante de como a extração de características funciona é mostrado por Theodoridis e Koutroumbas (2009). Sejam duas imagens rotuladas respectivamente em classe A e classe B, e que a primeira classe corresponde a células saudáveis e a segunda a células cancerígenas. Imagine também que o sistema tenha acesso a um banco de dados com várias dessas imagens rotuladas.

Primeiramente o sistema é projetado para tirar alguma medida ou calcular algum parâmetro a partir dessas imagens. A figura 16 mostra um gráfico do valor médio da intensidade da imagem em cada região de interesse versus o desvio padrão correspondente nessa mesma região.

Cada ponto no gráfico representa a característica de uma imagem, e é fácil perceber que os pontos da classe A e da classe B se organizam separadamente no espaço.

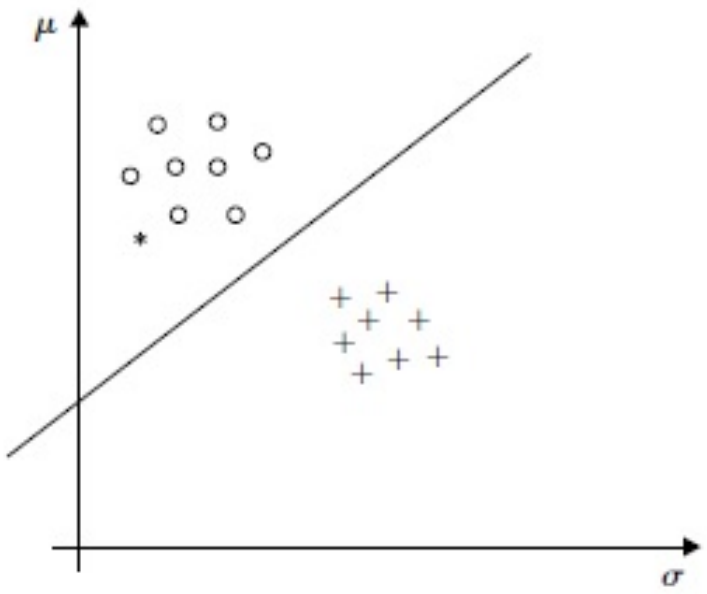
Dessa forma, se um novo dado de teste for inserido no sistema de classificação (nesse caso, o dado de teste é representado por $*$), as mesmas características serão extraídas desse dado de teste. Nesse caso, pelo gráfico, o dado de teste se aproxima mais da classe A, então é possível dizer que o dado $*$ corresponde a uma imagem que contém somente células saudáveis.

Esse exemplo mostra a importância da etapa de extração de características, pois com apenas duas características foi possível classificar as imagens em questão. Esse processo é essencial devido a inviabilidade de trabalhar com os dados puros; a extração de características permite trazer um conjunto imenso de dados para uma dimensão reduzida (RAMOS, 2014).

Portanto, um conjunto de dados referentes a um objeto pode gerar uma quantidade l de características, de forma que todas as características x_i, $i = 1, 2, 3...l$ referentes a esse dado formem um vetor de características.

$$x = [x_1, x_2, ..., x_l]^T. \tag{2.8}$$

Figura 16 – Gráfico que mostra a média de intensidade versus o desvio padrão para várias imagens do mesmo banco de dados. Nesse caso, a linha reta divide as classes A (◯) e B (+).



Fonte: Retirado de (THEODORIDIS; KOUTROUMBAS, 2009).

Cada vetor se refere unicamente a um conjunto de características pertencentes a um objeto. De acordo com Theodoridis e Koutroumbas (2009) o vetor de características x pode ser interpretado como uma variável aleatória.

2.3.1.1 Momentos da Variável Aleatória

Como foi mencionado anteriormente, os vetores de características podem ser interpretados como variáveis aleatórias. Em probabilidade, a variável aleatória tem o objetivo de transpor eventos do espaço de amostras para segmentos de retas no conjunto dos números reais (ALENCAR, 2009).

Em outras palavras, a variável aleatória X realiza o seguinte mapeamento $X : \Omega \rightarrow R$. O conjunto de dados amostrados Ω é mapeado para o domínio real R a partir da variável aleatória.

A ideia aqui, portanto, é gerar um conjunto de características relacionadas a algumas medidas estatísticas dessa variável aleatória. De acordo com Theodoridis e Koutroumbas (2009) essas técnicas são bastante utilizadas para reconhecimento de padrões em imagens.

Uma medida bastante utilizada são os momentos da variável aleatória, que podem revelar informações importantes, tais como tendências e desvios. A partir dos momentos,

por exemplo, é possível calcular a média, o desvio em torno da média, a tendência dos dados, simetria da distribuição dos dados, e etc (ALENCAR, 2009).

Seja $f(X)$ uma função de variável aleatória e $p(x)$ a função de densidade de probabilidade dessa variável aleatória, o valor esperado em relação a X da função $f(X)$, ou seja, a esperança matemática, é determinado por:

$$E[f(X)] = \int_{-\infty}^{\infty} f(x)p(x)dx. \tag{2.9}$$

Dessa forma, de acordo com ALENCAR (2009), os vários momentos de X são dados pela equação (2.10).

$$m_i = E[X^i] = \int_{-\infty}^{\infty} x^i p(x)dx. \tag{2.10}$$

Nesse caso $i = 1, 2, 3...$ e cada momento têm um significado ou uma interpretação física. Assim,

- $m_1 = E[X]$, é a média aritmética ou valor médio;
- $m_2 = E[X^2] = \sigma^2$, é a variância;
- $m_3 = E[X^3]$, mede a assimetria da função de densidade de probabilidade;
- $m_4 = E[X^4]$, mede o achatamento da função de densidade de probabilidade.

Esses momentos das variáveis aleatórias são os mais importantes e são utilizados para compor o vetor de características.

Pelo significado de cada momento mostrado na lista acima, é possível deduzir que cada um pode ser utilizado para caracterizar os sinais de entrada do sistema de classificação de alguma maneira.

A variância $m_2 = \sigma^2$, por exemplo, mede o quão um sinal está disperso do valor esperado ou do valor médio. O m_4, em aplicações de imagens, dependendo do seu valor pode indicar que o histograma resultante da imagem seja definido como platicúrtico para valores grandes, leptocúrtico para valores pequenos e mesocúrgico caso contrário.

Então é possível dizer que esses momentos podem trazer consigo informações valiosas a respeito dos dados de entrada do sistema de classificação.

2.3.1.2 Curvatura do Sinal

De acordo com Marshall (2021) a curvatura é uma medida do quanto a curva se desvia de uma linha reta. Em outras palavras, para cada ponto da curva existe uma curvatura que representa uma taxa de mudança da curva. Isso significa que, matematicamente, a curvatura é a magnitude da segunda derivada da curva em um determinado ponto.

A curvatura em um ponto tem um raio que é definido como o raio de um círculo que se ajusta melhor à curva naquele ponto. Esse círculo, por sua vez, tem um comprimento

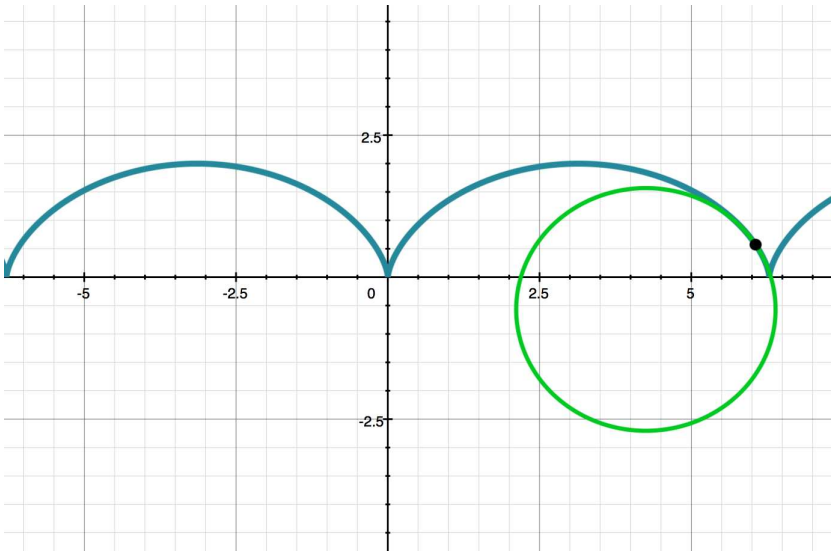
de arco, e em relação a esse comprimento que a derivada é calculada. A equação (2.11) traduz essas palavras:

$$\kappa = \left| \frac{dT}{ds} \right|, \tag{2.11}$$

em que κ é a curvatura, T é o vetor unitário tangente ao ponto de interesse e s é o comprimento do arco do círculo.

Portanto, é intuitivo pensar que a curvatura é o inverso do raio da curva $\kappa = 1/r$, e de fato, para curvas mais acentuadas, o raio é menor, enquanto para curvas mais planas o raio é maior, e, idealmente, para retas o raio é infinito, resultando em curvatura nula, o que é totalmente coerente. O círculo mostrado na Figura 17 é tangente ao ponto e o raio desse círculo que é utilizado no cálculo da curvatura.

Figura 17 – Círculo que melhor se ajusta à curva.

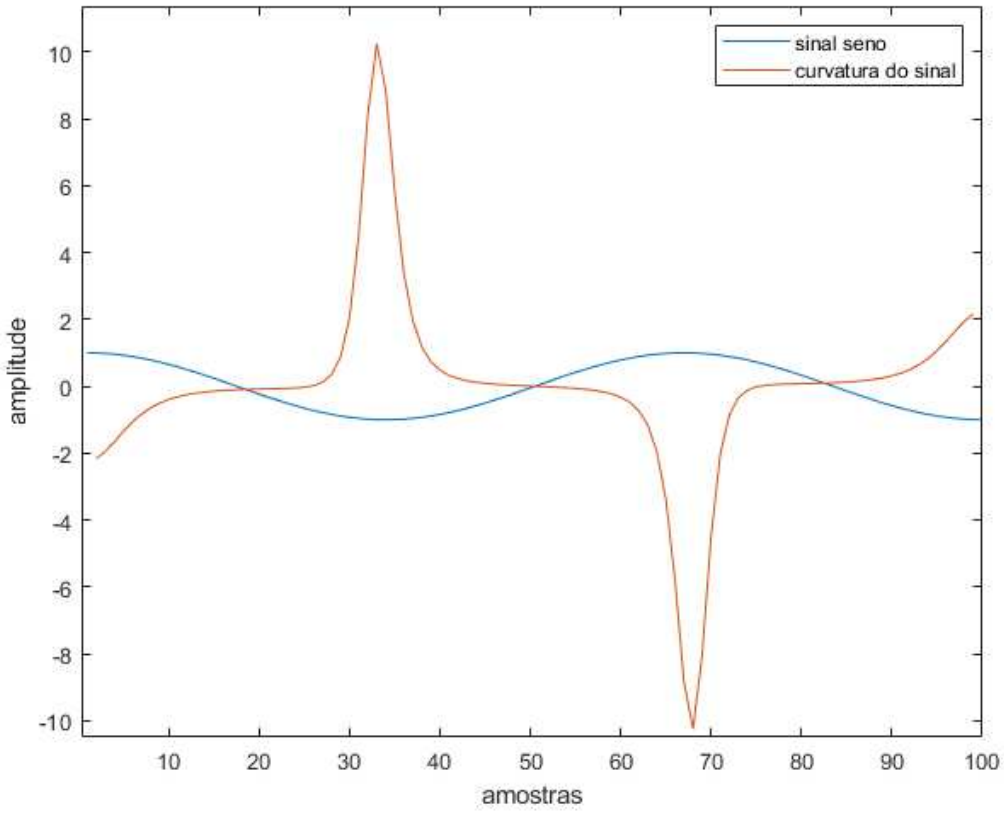


Fonte: Retirado de Marshall (2021).

Perceba que o cálculo mostrado na equação (2.11) envolve derivada de funções vetoriais, e que isso implica que T deve ser parametrizado com relação ao comprimento de arco da curva, e que os métodos de cálculo vetorial podem ser encontrados com mais detalhes em livros de cálculo. Aqui é mais interessante analisar este cálculo para sinais digitais, discretos no tempo.

Para esses sinais, basta fazer o cálculo da primeira e da segunda derivada, o que basicamente é a diferença da amplitude entre uma amostra atual e uma amostra anterior. A Figura 18 mostra a curvatura de um sinal senoidal calculada dessa forma, por se tratar de um sinal digital.

Figura 18 – Curvatura de um sinal digital.



Fonte: Do Autor.

Perceba que nos picos do sinal a curvatura é maior, enquanto que nas curvas mais suaves a curvatura se aproxima do zero. Portanto, essa medida de curvatura pode ser uma boa forma de caracterizar o sinal, visto que cada sinal pode ter uma curvatura diferente do outro.

2.3.2 Aprendizagem e Classificação

Como já foi discutido previamente no inicio desta seção, o reconhecimento de padrões tem o objetivo de extrair padrões e comportamentos comuns de um conjunto de dados. As técnicas de reconhecimento de padrão são uma área de estudo do aprendizado de máquina.

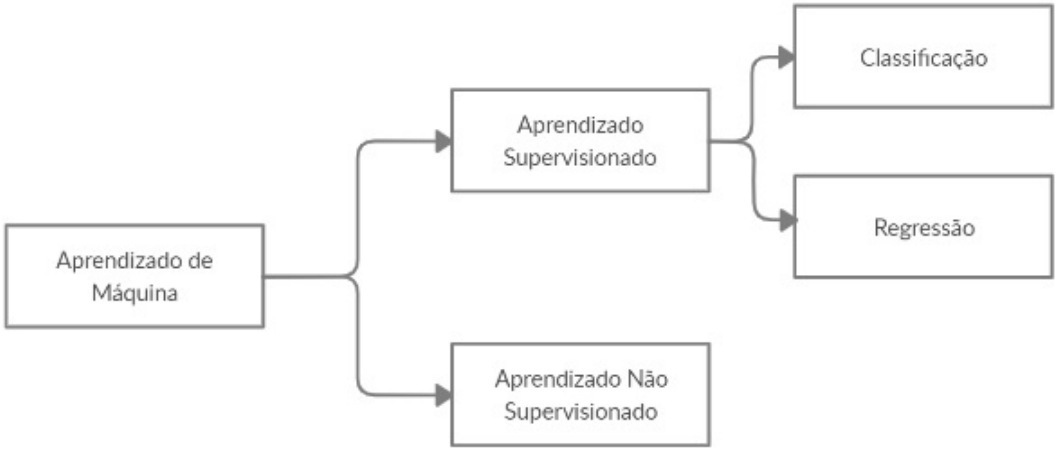
O reconhecimento de padrões está diretamente ligado à modelagem preditiva de um conjunto de dados, ou seja, a partir de um conjunto de treinamento, a intenção é prever o comportamento dos dados de teste. Esta tarefa é também chamada de aprendizado (RAMOS, 2014).

Dessa forma, é possível aplicar o aprendizado em um sistema de duas maneiras: treinamento supervisionado e não supervisionado (THEODORIDIS; KOUTROUMBAS,

2009). Existe também o treinamento semi-supervisionado, mas é menos utilizado.

De forma breve, o treinamento supervisionado assume que existe um conjunto de dados rotulados de acordo com as classes presentes no problema. Já no treinamento não supervisionado os dados de treinamento não são rotulados (JAIN, 2010). A Figura 19 ilustra a diferença entre os treinamentos.

Figura 19 – Hierarquia do aprendizado de máquina.



Fonte: Do Autor.

De acordo com Theodoridis e Koutroumbas (2009), no caso do treinamento não supervisionado, onde as classes não são conhecidas, os algoritmos de treinamento buscam similaridades entre os vetores de características e fazem o agrupamento desses dados. Tal técnica é conhecida como *clustering*.

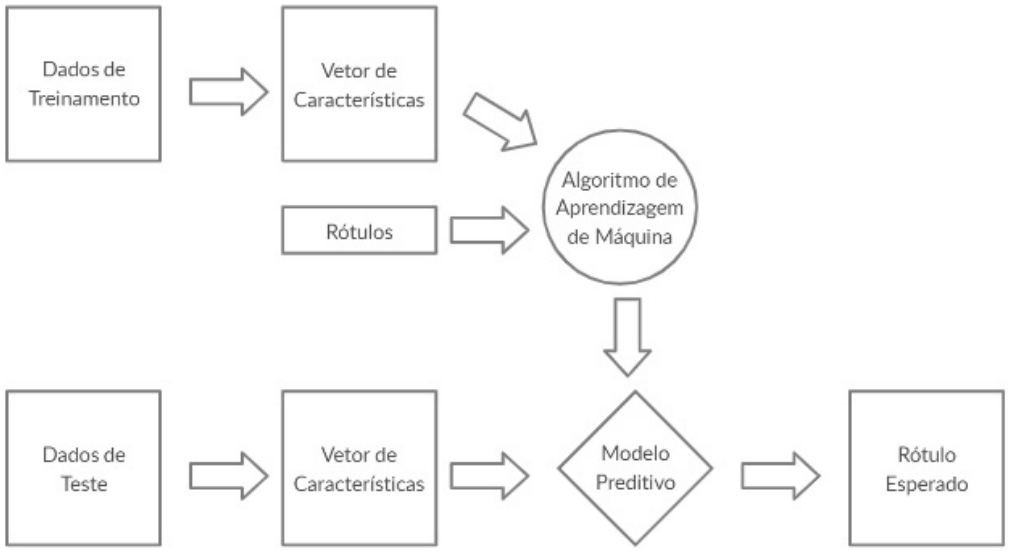
O grande problema do aprendizado não supervisionado é que diferentes algoritmos de *clustering* tendem a apresentar diferentes resultados e desempenho (THEODORIDIS; KOUTROUMBAS, 2009). O número ótimo de agrupamento desses algoritmos se apresenta também como uma variável a mais a ser determinada no problema.

Dessa forma, o aprendizado supervisionado é mais vantajoso quando se conhece inteiramente o conjunto de dados que são utilizados no sistema. A Figura 20 mostra o fluxograma de um aprendizado supervisionado.

A função do treinamento, portanto, é definir um modelo a partir do conjunto de características e esse modelo é treinado por um classificador. Os classificadores fazem a separação dos dados de teste baseado no modelo treinado e os caracteriza em classes, que são definidas conforme o tipo de treinamento.

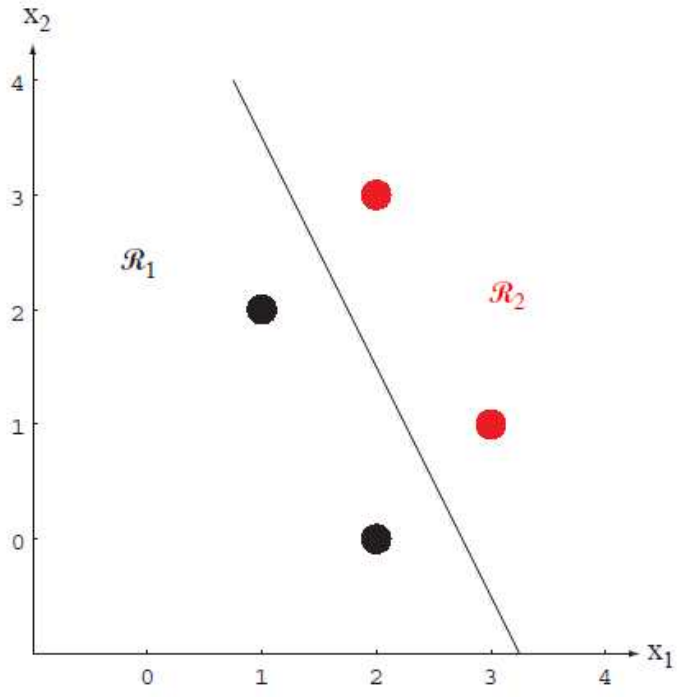
Na Figura 21 é mostrado um tipo de classificador linear em que uma reta, conhecida como hiperplano, divide os dados em duas classes. Mais detalhes sobre as técnicas de implementação desses classificadores podem ser encontradas em (THEODORIDIS; KOUTROUMBAS, 2009) e (DUDA; HART, 1973).

Figura 20 – Fluxograma do aprendizado supervisionado.



Fonte: Retirado de Ramos (2014)

Figura 21 – Exemplo de um classificador linear que separa as características de um conjunto de dados em duas classes.



Fonte: Retirado de Duda e Hart (1973).

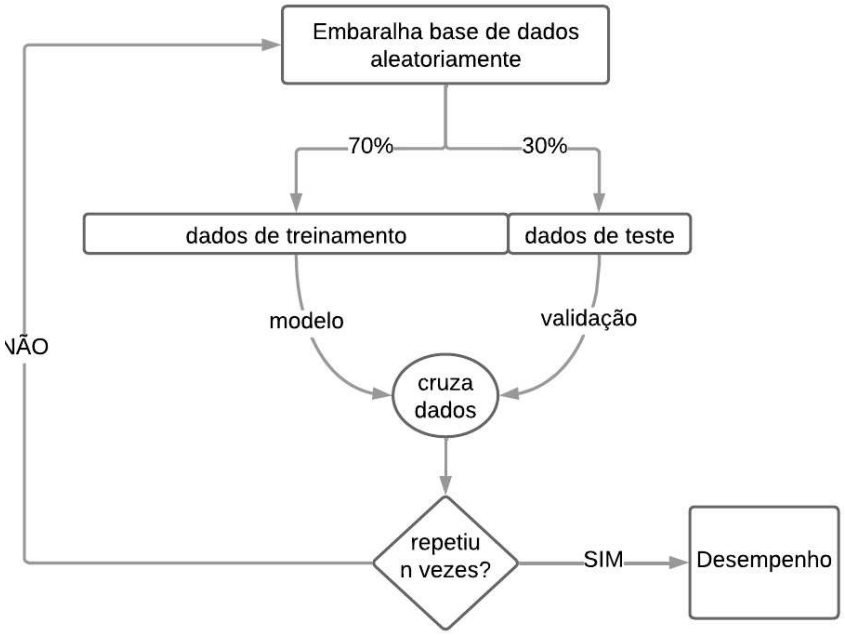
2.3.3 Validação Cruzada como Método de Análise

A acurácia do sistema de classificação depende de diversos fatores, tais como a escolha das técnicas de extração de características, desempenho dos classificadores e etc. Essa medida é importante para validar o sistema de classificação proposto. Dessa forma, aqui será abordado o método de análise e validação do sistema de classificação, por meio de validação cruzada. Em outras palavras, é mostrado como avaliar a habilidade de generalização de um modelo a partir de um conjunto de dados (KOHAVI; PROVOST, 1998).

A validação cruzada consegue generalizar um modelo a partir de um conjunto de dados. Ou seja, a partir de dados de testes é possível determinar o desempenho da classificação obtida pelo modelo criado. O princípio por trás da validação cruzada é dividir um conjunto de dados em dois subconjuntos exclusivos de dados de treinamento do modelo e os dados de teste e validação, e o método *hold out* costuma particionar os dados na proporção de 70% dos dados para estimar o modelo e 30% para valida-lo (KOHAVI, 1995).

Com os dados particionados, basta levantar o modelo de acordo com as técnicas citadas acima e utilizar os dados de teste para as validações, e assim observar o desempenho do sistema de classificação. Lembrando que antes de serem separados, os dados da base são embaralhados de forma aleatória, e então se obtém um desempenho, mas o desempenho final é obtido através da repetição desse processo diversas vezes, como pode ser observado na Figura 22.

Figura 22 – Validação cruzada pelo método *hold out*. Processo se repete por n vezes predefinida no início do algoritmo.

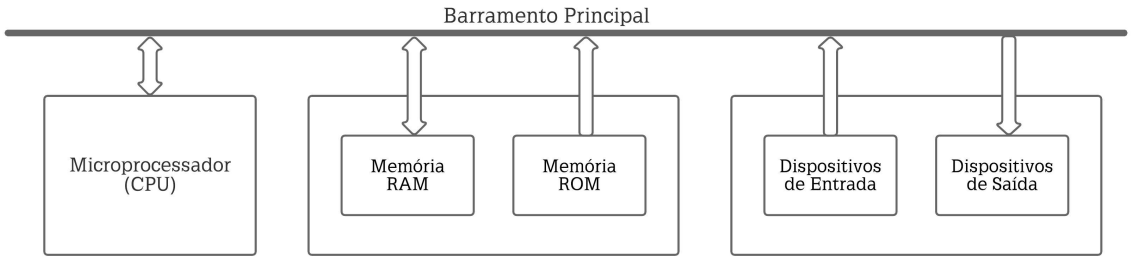


Fonte: Do autor.

2.4 Recursos dos Microcontroladores

É possível observar na Figura 23 os três blocos principais que compõem um microcontrolador: um bloco Unidade Central de Processamento, do inglês *Central Processing Unit* (CPU) ou microprocessador, um bloco de memória e um bloco de periféricos ou dispositivos I/O.

Figura 23 – Elementos básicos de um microcontrolador.



Fonte: Do autor.

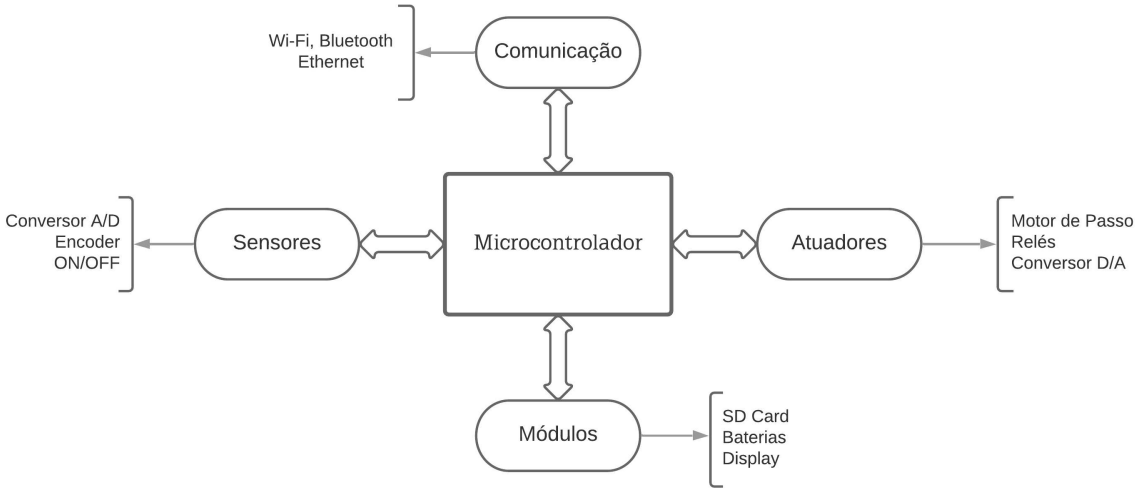
A função principal do barramento é garantir que os blocos se comuniquem, e as transferências de informações sejam realizadas, e ele é dividido em três segmentos: barramento de endereços, barramento de dados e barramento de controle.

Uma unidade de memória do microcontrolador armazena dados e instruções através de chips ROM e RAM. A memória RAM é uma memória volátil e é utilizada para armazenar programas e dados que são temporários e podem mudar durante a execução de um programa. Por convenção a memória RAM é chamada de memória principal, e a CPU só consegue executar instruções que estejam armazenadas nela.

A unidade de processamento CPU pode ser dividida em duas partes. A primeira é uma Unidade Lógica Aritmética (ULA), que é capaz de operar os dados binários, executando operações lógicas e aritméticas simples sobre um ou mais operandos (STALLINGS, 2002). Já a Unidade de Controle (UC) é utilizada para coordenar as operações entre as outras unidades, determinando o fluxo de operações do sistema (STALLINGS, 2002) (LEE; SESHIA, 2013).

Os periféricos, ou dispositivos I/O são unidades que transferem dados entre o microcontrolador e os dispositivos externos via portas de entrada/saída. A transferência envolve dados, status e sinais de controle. Esses dispositivos podem ser, por exemplo, conversores analógicos/digitais, *timers*, memórias externas e etc. A Figura 24 mostra a gama de periféricos possíveis para um microcontrolador, separados de acordo com suas áreas de atuação.

Figura 24 – Variedade de periféricos de um microcontrolador.



Fonte: Adaptado de Lee e Seshia (2013).

2.4.1 Tipos de Pinos e os *Timers*

A pinagem de um microcontrolador pode ser classificada de acordo com a sua utilidade e o tipo de dado que passa por cada pino, e basicamente são divididos em três tipos: os pinos I/O Digital de Uso Geral, do inglês *General-Purpose Digital I/O* (GPIO), os pinos de PWM e os pinos de entrada analógica (LEE; SESHIA, 2013).

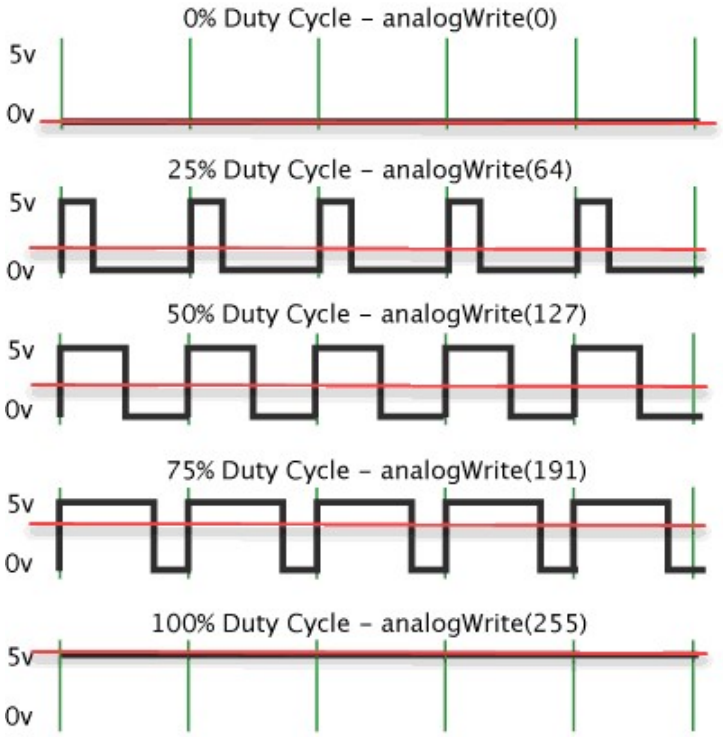
Os GPIO são pinos digitais que permitem ao *software* do microcontrolador ler ou gravar níveis de tensão que representam um nível lógico alto ou baixo, zero ou um. Se a tensão de alimentação do processador for V_{DD}, uma tensão de entrada próxima ao V_{DD} representa um nível alto, e uma tensão próxima de 0 representa um nível baixo.

Já os pinos PWM servem para fornecer uma quantidade variável de energia de forma eficiente para dispositivos de *hardware* externos. Pode ser usado para controlar, por exemplo, a velocidade de motores elétricos (LEE; SESHIA, 2013).

Um pino PWM é capaz de gerar um sinal que muda rapidamente entre alto e baixo em alguma frequência fixa, variando a quantidade de tempo que mantém o sinal alto. O ciclo de trabalho, ou *Duty Cycle* é a proporção de tempo que a tensão está alta. Se o ciclo de trabalho for 1, a tensão estará sempre alta. Se o ciclo de trabalho for 0, a tensão estará sempre baixa. Um ciclo de 0.8 significa que 80% do tempo o sinal é nível alto. A Figura 25 mostra um exemplo de um sinal PWM de 8 bits sendo modulado em uma saída analógica de um Arduíno. Veja que a linha vermelha representa a saída analógica, uma valor entre 0 e 5 V, e que nos extremos de *Duty Cycle* 0 e 1 a saída é 0 V e 5 V respectivamente.

Para utilização dos pinos PWM, o *software* deve definir o ciclo de trabalho e a frequência do sinal desejada. O dispositivo então fornece energia para algum módulo

Figura 25 – Saída analógica sendo modulada por PWM em um Arduíno.



Fonte: Retirado de Heath (2017).

externo em proporção ao ciclo de trabalho especificado (RAFIQUZZAMAN, 2000).

A respeito dos pinos de entrada e saída analógico, cada microcontrolador vai apresentar uma resolução diferente, pois esse parâmetro depende da quantidade de bits do conversor AD. Para efeito de comparação, abaixo é mostrado um cálculo que contrapõe a resolução do módulo mencionado com a resolução de um pino de 10 bits, que é o número mais comum nos microcontroladores encontrados no mercado. Utilizando da equação (2.4), em que n é o número de bits do conversor, para um conversor de 10 bits:

$$R_{bits} = 2^{10} = 1024, \tag{2.12}$$

enquanto que para 16 bits tem-se:

$$R_{bits} = 2^{16} = 65536. \tag{2.13}$$

Dessa forma, o conversor digitaliza um dado analógico em uma escala de números inteiros entre 0 e $2^{bits} - 1$. Ou seja, uma tensão de 0 a $10V$, por exemplo, seria digitalizada numa escala de 0 a 1023 para 10 bits de resolução e de 0 a 65536 para 16 bits. Veja que é uma conversão linear em que cada valor analógico representa um número inteiro dentro da escala, então mais bits implicam em mais resolução, ou seja, numa representação mais exata da variável física. Esse cálculos devem ser levados em consideração principalmente para a escrita do *software* que irá processar esses dados. É importante lembrar que esses

pinos podem servir para conversão digital em analógica também, fazendo o caminho inverso é possível controlar uma variável física a partir de uma escala digital definida pelo programa. Um exemplo seria o controle da intensidade luminosa de um LED pelo usuário do sistema computacional.

Por fim, os *timers* são uma aplicação imprescindível em sistemas embarcados, pois eles mantêm o *clock* de uma operação sincronizado com um *clock* do sistema ou externo. Os *timers* podem ser utilizados para geração de *delays*, para gerar taxas de transmissão, por exemplo. Partindo do principio que os sistemas embarcados utilizam funções rotineiras com certa recorrência, como verificar o estado de um pino ou registrar a leitura de um sensor, os *timers* são os responsáveis por sincronizar essas atividades. Isso porque algumas dessas ações são feitas pelo microcontrolador de forma periódica. É comum que os microcontroladores tenham 3 *timers*, como mostrado a seguir:

- ***Timer 0* e *Timer 1***: eles têm 16 bits de tamanho, e podem ser acessados com dois registradores de 8 bits, THx e TLx, representando um byte baixo e um byte alto do *timer* de 16 bits. Esses *timers* são configurados para zerar quando acontece o *overflow*, ou seja, comuta de $0x1FFF$ para $0x0000$ e um *flag* de sinalização é acionado.

- ***Timer 2***: diferentemente do 0 e 1, esse *timer* é acessado por 4 registradores, adicionalmente THx2 TLx2. Ele tem também registradores que podem fazer operações de *capture*, que registra o tempo em que um sinal de entrada é recebido em algum pino. A principal vantagem sobre os *timers* 0 e 1 é a simplicidade da programação das operações de varredura e leitura.

É possível configurar os *timers* em quatro modos de operação diferentes, o que vai depender da aplicação desejada e do tipo de operação a ser realizada. As especificações de cada modo são mostradas abaixo:

- *MODE* 0*, modo* 13 *bit*: neste modo, o THx atua como um temporizador de 8 bits e o TLx atua como um temporizador de 5 bits. O TLx conta de 0 até 2^5 e, em seguida, redefine para 0 e incrementa o THx em 1 bit, que vai até 2^8. O *overflow* acontece em 2^{13}.

- *MODE* 1*, modo* 16 *bit*: similar ao modo 0, neste caso o THx atua como um temporizador de 8 bits e o TLx atua como um temporizador de 8 bits. O TLx conta de 0 até $2^8 - 1$ e, em seguida, redefine para 0 e incrementa o THx em 1 bit, que vai até $2^8 - 1$ também. O *overflow* acontece em $2^{16} - 1$.

- *MODE* 2*, modo* 8 *bit*: neste modo, o TLx atua como o temporizador e o THx contém o valor de recarga, ou seja, o THx serve como um *backup* do TLx toda vez que ocorre o *overflow*. Quando o TLx atinge $2^8 - 1$ e é incrementado, em vez de

redefini-lo para 0, ele será redefinido para o valor armazenado em THx. Este modo é muito comumente usado para gerar a taxa de transmissão usada na comunicação serial (ATMEL CORPORATION, 2007).

- *MODE* 3*, modo split*: esse modo divide um *timer* de 16 bits em dois de 8 bits, e ambos os *timers* secundários são zerados quando atingem o *overflow* em $2^8 - 1$. Caso for configurado no *Timer 1* os *timers* secundários gerados param de contar quando chegam no *overflow*. Por isso é chamado de modo *split* pois divide o *timer* em dois.

Eis um exemplo de como calcular as especificações de um *timer* para configurá-lo da maneira como é desejado no projeto. Para efeito prático será utilizado o *Timer 0*. Imagine então que se deseja gerar um *delay* de $1ms$ e que o oscilador do microcontrolador é de $F_{osc} = 16MHz$. De acordo com Atmel Corporation (2007) o *Tick*, que é o tempo que o *timer* demora para ser incrementado, pode ser calculado de acordo com equação (2.14):

$$Tick = \frac{1}{\frac{F_{osc}}{12}} = \frac{12}{F_{osc}}. \tag{2.14}$$

Então, para $16MHz$ o *Tick* será de:

$$Tick = \frac{12}{16MHz} = 0,75\mu s.$$

Como,

$$Delay = TimerCount\ X\ Tick. \tag{2.15}$$

então, para o *delay* requerido tem-se:

$$TimerCount = \frac{Delay}{Tick} = \frac{1ms}{0,75\mu s} = 1333.$$

Ou seja, para gerar um *delay* de $1ms$ o *timer* precisa ser incrementado em 1333 vezes.

2.4.2 Comunicação Serial SPI

Uma maneira de usar os pinos do microcontrolador com eficiência é enviar informações sobre eles serialmente, em sequências de bits. Essa eficiência implica em menos gasto de energia e em um número menor de pinos, o que possibilita esses sistemas serem fisicamente mais compactos (LEE; SESHIA, 2013).

Essa interface é chamada de interface serial e uma série de padrões evoluíram para essas interfaces seriais para possibilitar que dispositivos de diferentes fabricantes pudessem ser conectados e comunicados entre si. Um microcontrolador normalmente usa protocolos para comunicação serial como, o Receptor e Transmissor Assíncrono Universal, do inglês *Universal Asynchronous Receiver and Transmiter* (UART), o Circuito Inter Integrado,

do inglês *Inter Integrated Circuit* (I2C), e o SPI, por exemplo. Esses protocolos ditam as regras para a conversão e transmissão dessas palavras binárias entre um dispositivo periférico e o microcontrolador.

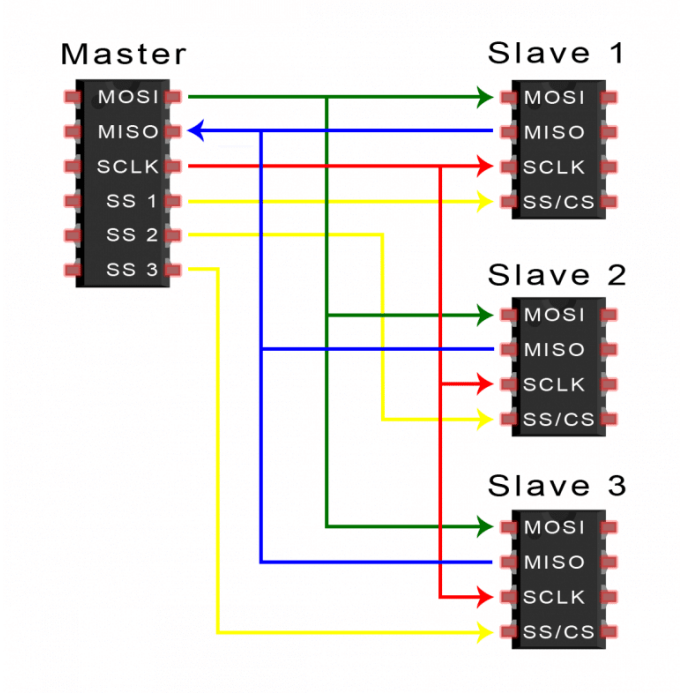
O protocolo SPI é vastamente utilizado para comunicar microcontroladores com módulos de cartão SD, por exemplo. Um benefício desse tipo de comunicação é que os dados transmitidos não precisam ser comprimidos em pacotes, o que resulta em uma transmissão sem interrupção, ou seja, não há limitação de quantidade de bits de dados que podem ser transmitidos continuamente. Esses dispositivos são síncronos e podem ter uma configuração de único mestre e único escravo, mas um mestre pode controlar mais de um escravo, diferentemente do UART. Um dispositivo SPI tem a seguinte pinagem:

- MOSI: saída do mestre e entrada do escravo, é o pino que transmite os dados do mestre para o escravo.
- MISO: entrada do mestre e saída do escravo, é o que transmite os dados do escravo para o mestre.
- SCLK: pino de *clock*;
- SS/CS: pino que seleciona para qual escravo os dados vão ser enviados.

O mestre envia dados ao escravo bit a bit através da linha MOSI e o escravo recebe os dados enviados do mestre no pino MISO. Os dados enviados do mestre para o escravo geralmente são enviados com o bit mais significativo primeiro. O escravo também pode enviar dados de volta ao mestre por meio da linha MISO em série. Os dados enviados do escravo de volta ao mestre geralmente são enviados com o bit menos significativo primeiro.

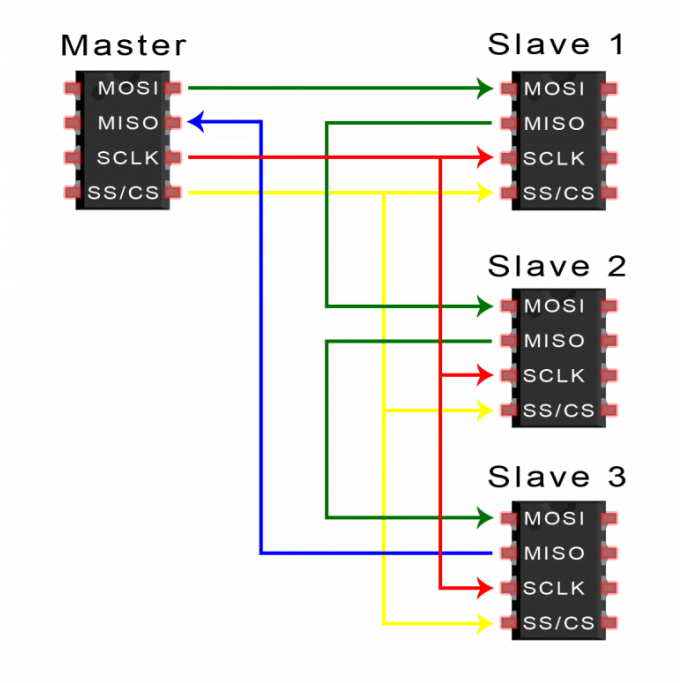
A Figura 26 mostra como um mestre pode controlar vários escravos. Veja que o sinal de *clock* e de *chip select* são compartilhados e conectados pelo mesmo ponto. Nesse caso os pinos MOSI e MISO são compartilhados pelo mesmo ponto também mas é possível configurá-los da forma mostrada na Figura 27.

Figura 26 – Configuração para múltiplos escravos SPI.



Fonte: Retirado de Campbell (2017).

Figura 27 – Outro tipo de configuração para múltiplos escravos SPI.



Fonte: Retirado de Campbell (2017).

2.4.3 Configuração de Registradores e Controle de Interrupção

Configurar os registradores é manipular os bits armazenados neles a fim de executar alguma tarefa específica. Na seção anterior, por exemplo, foi mostrado que é possível selecionar o modo de operação de um *timer*. Isso é feito através da manipulação do registrador de 8 bits TMOD.

É por essa razão que é importante que o programador domine as técnicas de manipulação e configuração de registradores, mesmo que seja para tarefas mais simples, como atribuir um nível lógico a pino de saída, por exemplo. Uma configuração pode envolver a manipulação de apenas um bit ou de um conjunto de bits.

Seguindo o exemplo do registrador TMOD, imagine que se deseja atribuir o modo 2 para o *Timer 0*. Lembrando que os 4 primeiros bits são para o *Timer 1* e os 4 últimos para o *Timer 0*. E também que está sendo considerado a programação em linguagem C.

Seria intuitivo pensar que a configuração seria feito assim:

$$TMOD = 00000010_2.$$

Mas existe uma maneira mais prática de fazer essa operação, utilizando uma máscara («):

$$TMOD = (1 << 2).$$

Esse operador máscara desloca o bit 1 em duas casas, e então se chegaria no valor desejado. Mas existe um problema com esse método, pois deslocar os bits modificaria todo o conteúdo do restante do registrador. Em alguns casos isso não é prático. Imagine que se o *Timer 1* estivesse configurado para o modo 1 essa operação modificaria o modo desse *timer*.

Uma maneira simples de resolver isso é utilizando um operador OR, e então a operação seria:

$$TMOD = TMOD \mid (1 << 2).$$

Dessa forma se o TMOD estivesse em 0001000, ao realizar a operação o resultado da operação acima seria:

$$TMOD = 00100010.$$

Assim é possível configurar o *Timer 0* sem modificar a configuração do *Timer 1*.

Outros operandos como AND, NOT, XOR e etc podem ser utilizados também, a depender do tipo de manipulação que se deseja realizar. Em geral a maneira de configurar um registrador em C deve ser com a seguinte sintaxe:

$$REG = REG \ \textbf{OPERADOR} \ MÁSCARA. \tag{2.16}$$

A respeito das interrupções, elas são mecanismos que servem para pausar a execução de tudo o que um processador está atualmente fazendo para executar uma sequência de código predefinida chamada de ISR.

Três tipos de eventos podem disparar uma interrupção. A primeira é uma interrupção de *hardware*, onde algum *hardware* externo altera o nível de tensão em algum pino e isso causa a interrupção. A segunda é uma interrupção de *software*, onde o programa que está sendo executado dispara a interrupção através de alguma instrução específica. A terceira é chamada de interrupção por exceção, onde a interrupção é acionada por um *hardware* que detecta uma falha, como uma falha de segmentação.

Para os dois primeiros casos, uma vez que o ISR é concluído, o programa que foi interrompido continua de onde parou. No caso de uma exceção, depois que o ISR for concluído, o programa que acionou a exceção normalmente não é reiniciado. Em vez disso, o contador do programa é definido para algum local fixo onde o sistema operacional pode encerrar o programa.

Após a ocorrência de uma interrupção, o *hardware* deve primeiro decidir se deve responder. Se as interrupções estiverem desabilitadas, ele não responderá. O mecanismo para habilitar ou desativar as interrupções varia de acordo com o processador. Além disso, pode ser que algumas interrupções sejam habilitadas e outras não. As interrupções e exceções geralmente têm prioridades e uma interrupção será atendida apenas se o processador já estiver finalizado o atendimento de uma interrupção com uma prioridade mais alta. Normalmente, as exceções têm a prioridade mais alta e são sempre atendidas logo que ocorrem.

Quando o processador decide atender a uma interrupção, ele move o conteúdo do contador de programa atual e dos registradores de estado do processador para uma pilha, e então executa um desvio para um endereço que contém o salto para o ISR. O ISR deve armazenar na pilha os valores atuais em qualquer registrador que irá usar, e restaurar o valor desses registradores antes de retornar da interrupção, para que o programa interrompido possa retomar de onde parou. As demais interrupções são reativadas ao final desse processo também.

Quando uma interrupção é acionada pela alteração da tensão em um pino, a resposta pode ser disparada por nível ou disparada por borda. Para interrupções disparadas por nível a tensão será mantida no pino até que a interrupção seja executada. Através do controle de interrupção é possível também determinar qual dispositivo externo acionou uma ISR, em casos onde uma mesma ISR é compartilhada por diferentes dispositivos externos.

Agora imagine que se queira utilizar essas técnicas de controle para criar um programa que gere um *delay* de $1s$ em um pino de saída do microcontrolador. Utilizando das técnicas mostradas na seção 2.4.1, imagine, hipoteticamente, que o *Tick* de um *timer* seja de $1ms$, então para gerar um *delay* de $1s$, de acordo com equação (2.15) a variável

TimerCount deve ser de:

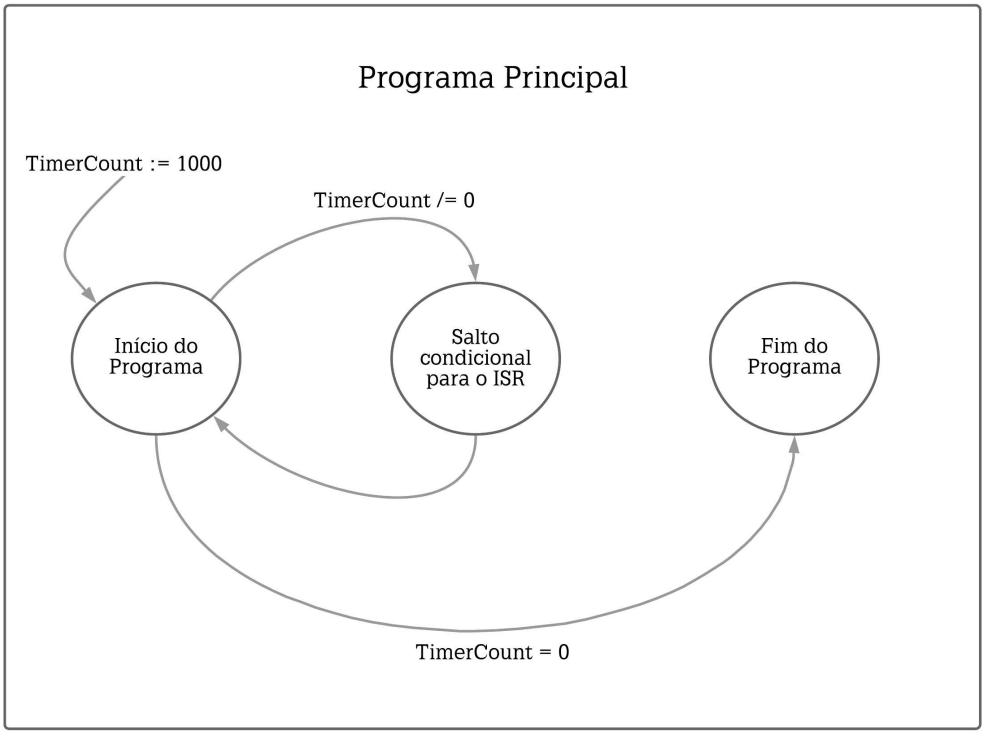
$$TimerCount = \frac{1s}{1ms} = 1000,$$

onde é possível modelar o programa exemplo de acordo com o código mostrado abaixo.

```
1  int main(void){
2     SysTickIntRegister(&ISR);
3     TimerCount = 1000;
4     while(TimerCount != 0)
5        ISR;
6  }
7
8  void ISR(void){
9      if(TimerCount != 0)
10        TimerCount--;
11 }
```

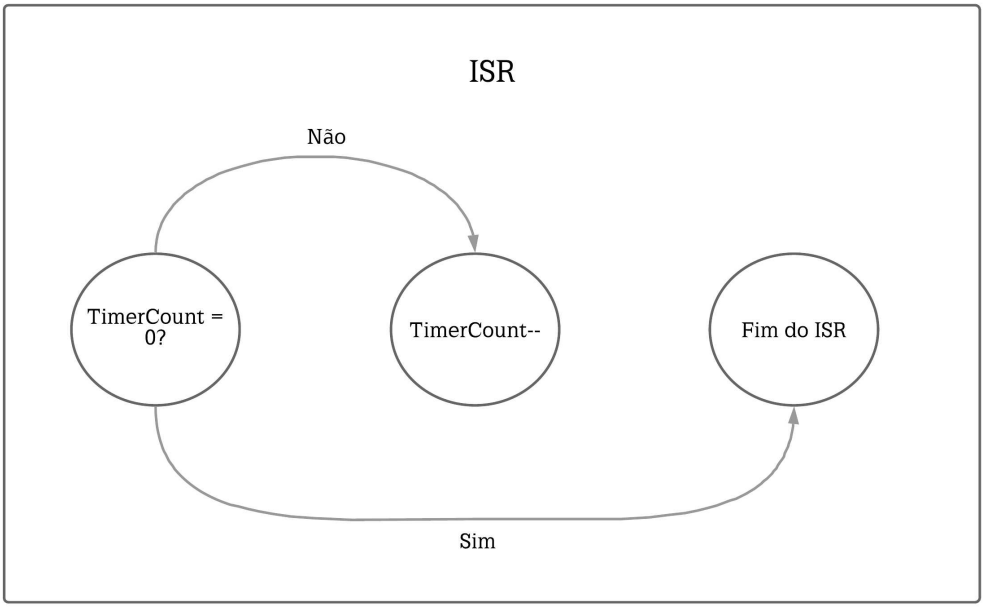
Os fluxogramas mostrados nas Figuras 28 e 29 ilustram o código do exemplo dado. Neste caso, a interrupção é gerada por *software*, já que é o próprio programa que gera o pedido e não um pino externo.

Figura 28 – Fluxograma do programa principal.



Fonte: Do autor.

Figura 29 – Fluxograma do ISR.



Fonte: Do autor.

3 Metodologia

A partir da utilização das ferramentas mencionadas no capítulo anterior foi possível conceber o projeto do escâner e fazer funcionar o sistema de classificação que identifica falhas nas peças metálicas mencionadas. Sendo assim, esse capítulo expõe os métodos empregados ao longo do desenvolvimento do trabalho, e estes serão explicados de forma que as técnicas referentes ao projeto do equipamento e os procedimentos para a identificação de falhas sejam expostas separadamente.

3.1 Projeto do Escâner 3D

O desenvolvimento do equipamento escâner 3D é dividido em três partes principais: o projeto mecânico, o projeto elétrico e o desenvolvimento do sistema embarcado. O projeto mecânico envolve o desenho e a fabricação das peças que compõem a estrutura principal do escâner, e toda essa etapa foi desenvolvida em Lima (2020). O projeto elétrico é a união de todos os circuitos e equipamentos responsáveis por fazer o acionamento dos motores e o funcionamento do sensor. O sistema embarcado é o cérebro do escâner, pois ele comanda a lógica de movimentação do equipamento, a aquisição e armazenamento dos dados e a interface com o usuário.

As especificações de projeto foram estabelecidas em concordância com o cumprimento dos objetivos estabelecidos, então todas as decisões sobre dimensão, geometria, área de varredura do equipamento foram tomadas baseadas na aplicação para a identificação de falhas nas estruturas metálicas mostradas, e em Lima (2020) é mostrado como foram estabelecidas essas especificações. Em resumo elas são colocadas na Tabela 1.

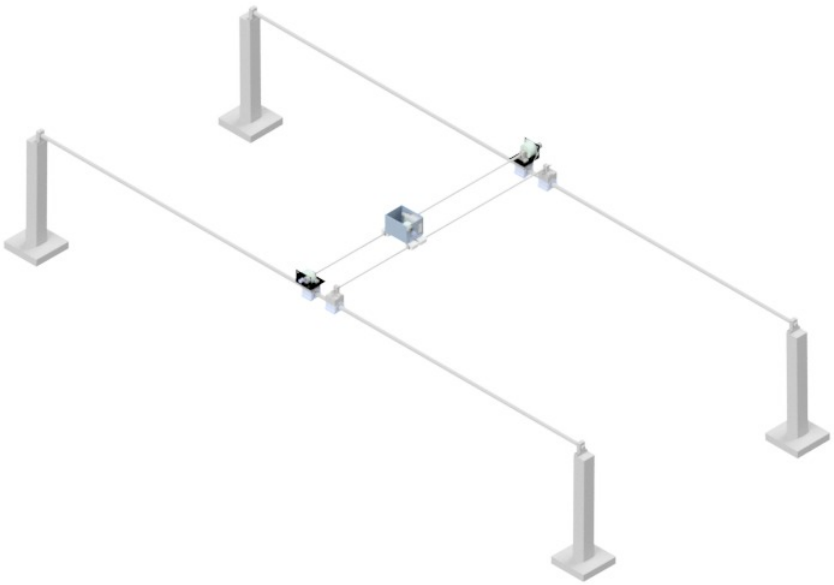
Novamente, é importante reforçar que para mais detalhes do desenvolvimento do projeto mecânico, bem como o desenho das peças e as especificações de cada uma delas podem ser encontradas em Lima (2020). No trabalho mencionado, o autor mostra a função que cada peça desempenha na movimentação do escâner e apresenta o projeto mecânico completo, e é mostrado na Figura 30 a estrutura mecânica do escâner 3D. Em seguida na Figura 31 é mostrado uma fotografia da base montada sobre o local de funcionamento do equipamento.

Tabela 1 – Especificações técnicas do escâner 3D.

comprimento $\times largura$	2500 $\times$1000mm
distância até a base	430 mm
material da estrutura	aço 1045
tipo de movimento do manipulador	na direção X (transversal) e Y (longitudinal)
motores utilizados para movimentação	motores de passo bipolares
tipo de sensor laser	sensor laser por triangulação Leuze ODSL8
modelo do microcontrolador	ESP32
forma de armazenamento de dados	cartão SD e servidor em nuvem

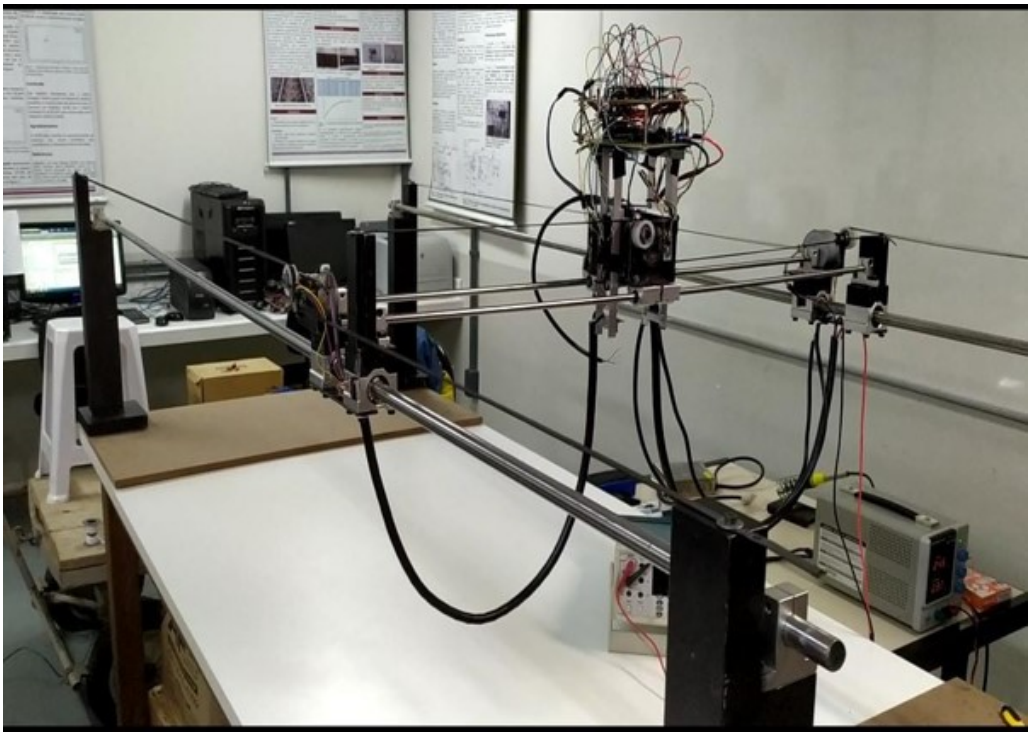
Fonte: Do Autor.

Figura 30 – Base mecânica do escâner 3D.



Fonte: Retirado de Lima (2020).

Figura 31 – Fotografia do escâner montado sobre sua base de funcionamento.



Fonte: Do autor.

Portanto, o funcionamento do escâner é dado por dois motores de passo que movimentam o sensor na direção longitudinal, e um outro motor que movimenta o sensor na direção transversal, e como o sensor mede a amplitude em Z (profundidade), todo objeto que estiver dentro da estrutura mostrada na Figura 30 consegue ser escaneado.

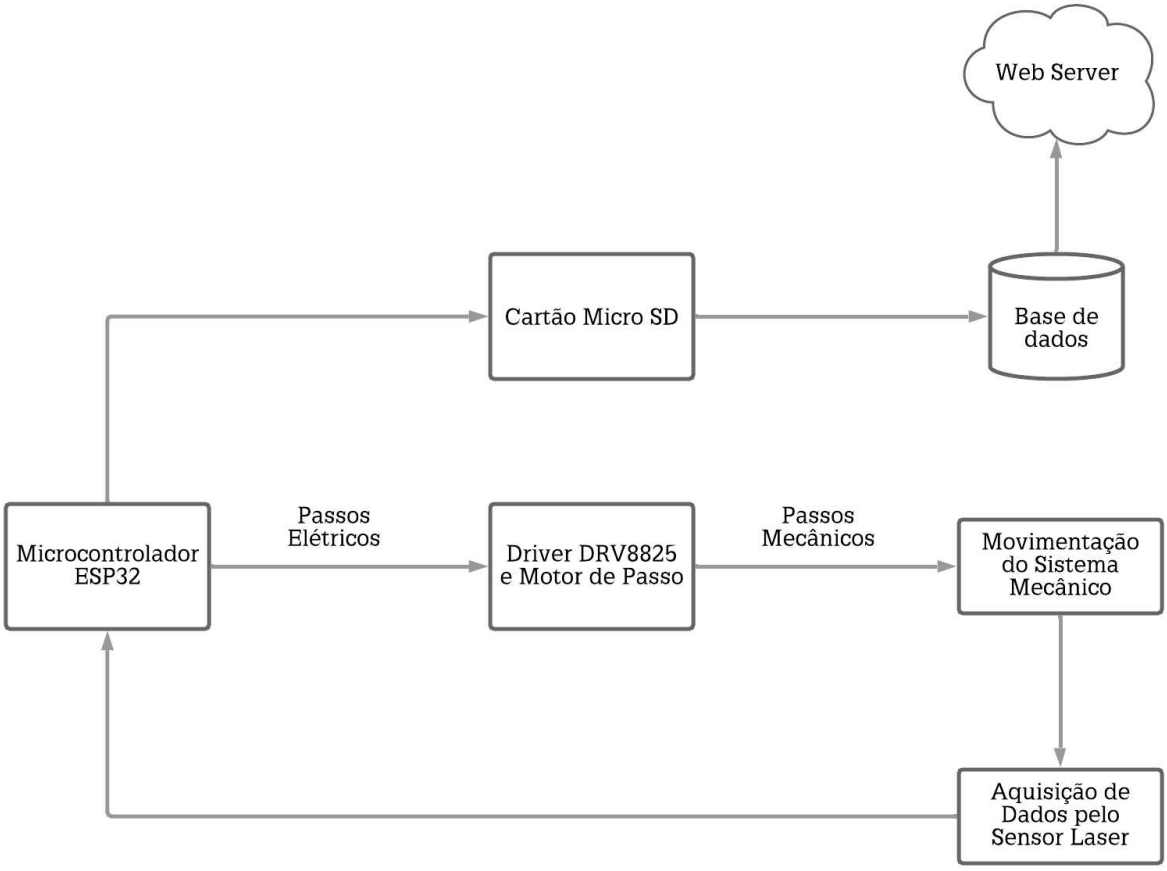
3.1.1 Projeto Elétrico

O projeto elétrico envolve o acionamento dos motores de passo para movimentar o suporte do sensor, além do circuito de aquisição e condicionamento de sinal. A figura 32 mostra um diagrama da configuração funcional do escâner, e o projeto elétrico tem o objetivo de unir fisicamente os blocos mostrados no diagrama através de um circuito elétrico que comunique todos os equipamentos. Sendo assim, os equipamentos utilizados para a realização do projeto elétrico foram:

- 1 Microcontrolador $ESP32$;
- 1 Sensor laser modelo Leuze $ODSL8$;
- 2 motores de passo de 48 pulsos por volta de 12 V;
- 1 motor de passo de 48 pulsos por volta de 24 V;
- 3 *drivers* DRV8825 para controle dos motores;

- 2 capacitores eletrolíticos de 100 μF e 50 V;
- 3 resistores de precisão de 250 Ω e 1%;
- 1 fonte chaveada estabilizada de 24V e 240 W;
- 1 conversor *buck step down* $LM2596$, de 24/12 V;
- 1 conversor *buck step down* $LM2596$, de 24/3,3 V;
- 1 módulo de cartão SD;
- 4 chaves fim de curso de alavanca.

Figura 32 – Diagrama do sistema proposto nesse trabalho.



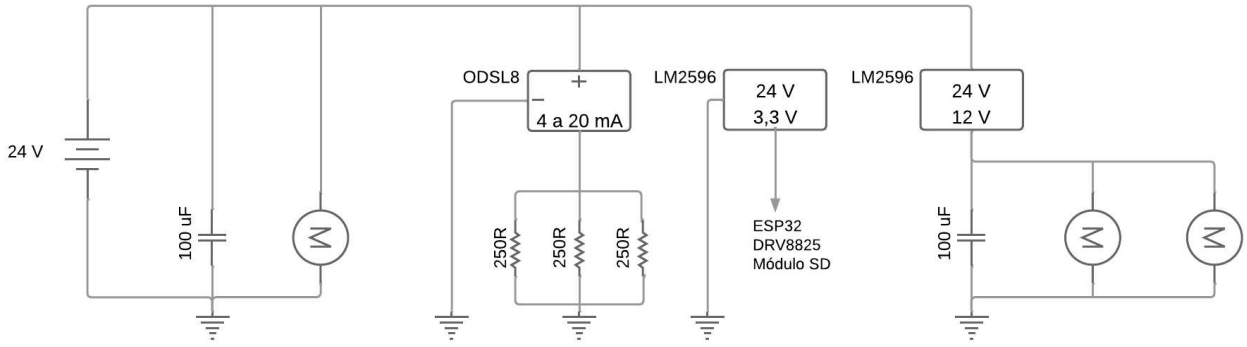
Fonte: Do Autor

Na Figura 33 é mostrado o circuito elétrico do escâner, desenvolvido com o intuito de unir eletricamente todos os equipamentos. A respeito desse circuito é possível fazer as seguintes observações:

- Tensão de 24 V: sensor $ODSL8$ e um motor de passo;

- Tensão de 12 V: Dois motores de passo;
- 3,3 V: Nível lógico do $ESP32$, do $DRV8825$ e do módulo do cartão SD;
- De acordo com o fabricante do $DRV8825$ Texas Instruments (2010), os capacitores são para diminuir *ripple* de tensão dos motores e para compensar indutâncias parasitas provocadas pelo acionamento dos motores;
- Os conversores $LM2596$ são do tipo Buck, operando a uma frequência de chaveamento de 150 kHz, e valor de tensão de saída é regulável de acordo com um potenciômetro (TEXAS INSTRUMENTS, 2013).

Figura 33 – Circuito elétrico do escâner.

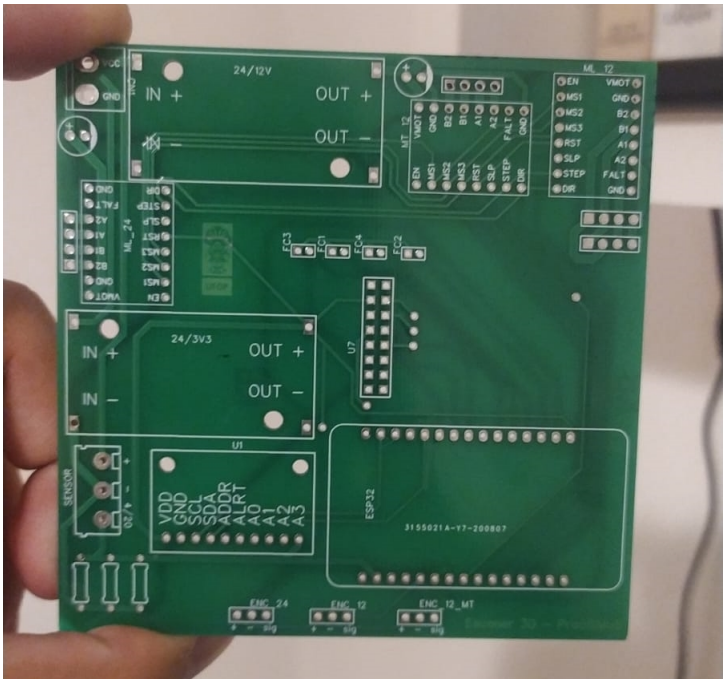


Fonte: Do autor.

Como foi mostrado na Figura 32, as funcionalidades de cada equipamento eletrônico devem ser integradas para que o escâner funcione da maneira como proposto. Por exemplo, para que o microcontrolador consiga salvar os dados no cartão SD, é necessária que haja uma conexão física entre os pinos de cada componente para que esta tarefa seja cumprida da maneira correta. Desa forma, foi necessário desenvolver uma PCI para que todas as conexões elétricas entre os pinos de cada elemento se conectassem de maneira correta, e que cada equipamento fosse alimentado com o nível de tensão correto para o seu funcionamento, de acordo com o circuito da Figura 33.

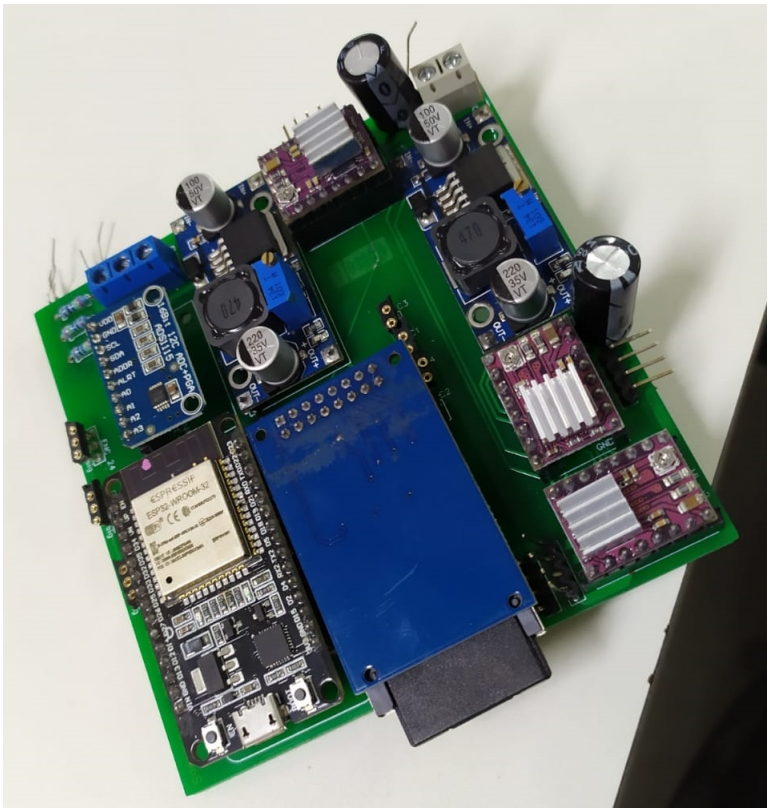
A Figura 34 mostra a PCI que foi desenvolvida para o projeto, e fabricada por uma empresa especializada, e o desenho e *layout* da placa, como conexões e posicionamento de cada equipamento, foi desenvolvido no *software* Altium, e na Figura 35 é possível observar a PCI com os dispositivos soldados sobre ela.

Figura 34 – PCI desenvolvida para o projeto.



Fonte: Do autor.

Figura 35 – PCI desenvolvida para o projeto com os equipamentos integrados.



Fonte: Do autor.

3.1.2 Projeto do Sistema Embarcado

Como foi dito na seção 2.4 o microcontrolador é o elemento capaz de coordenar as ações de um sistema embarcado, e o $ESP32$ foi escolhido para ser a parte central do sistema embarcado deste trabalho, pois apresenta uma série de funcionalidades que se encaixam com o projeto do equipamento proposto neste trabalho. Dentre elas, vale citar as mais pertinentes ao escâner:

- alimentação: 3,3 V e 5 V;
- conversor analógico-digital de 12 bits;
- quatro *timers* de 64 bits com oscilador de 8 MHz;
- periféricos compatíveis: Cartão SD, UART, SPI, I2C, LED PWM, Motor PWM, I2S, IR, pulse counter, GPIO, sensor de toque, conversor AD e DA;
- módulo Wi-Fi e conexão *bluetooth*.

As aplicabilidades desse micro que mais interessam são: os pinos PWM, para gerar a modulação necessária para o controle do movimento dos motores; o conversor analógico-digital de 12 bits, pois garante uma boa resolução; e a comunicação SPI, para garantir a compatibilidade com os periféricos, principalmente o cartão SD onde fica salva a base de dados.

Além disso, um grande diferencial do $ESP32$ é a presença de conectividade *bluetooth* e do módulo Wi-Fi. Este fato possibilitou a criação de um *Web Server* para interação com o usuário, em que ele consegue definir a área de escaneamento e baixar os dados medidos em arquivos. Tudo isso via tecnologia sem fio presente no micro.

No *datasheet*, disponível em Espressif Systems (2016), é possível verificar a pinagem do $ESP32$ utilizado no projeto e os detalhes sobre a funcionalidade de cada pino deste microcontrolador.

Para programar o $ESP32$, basta escrever um programa de instruções e passar para ele, que as funções desejadas são executadas. No caso do escâner, foi escrito um programa de instruções para o $ESP32$ que realizasse quatro tarefas principais, que são:

1. Acionamento dos motores de passo;
2. Conversão analógico-digital das medições do sensor laser;
3. Armazenamento das leituras em um dispositivo de memória externo;
4. Hospedar um *Web Server* para interface com usuário.

Dessa forma, nesta seção serão discutidos, além dos detalhes do projeto elétrico que envolve cada tarefa, os detalhes da programação de cada uma dessas funções no microcontrolador. Lembrando que o programa de instruções foi escrito em linguagem C e desenvolvido na plataforma do Arduino, e o programa completo pode ser encontrado no Apêndice A.

3.1.2.1 Controle de Movimentação dos Motores

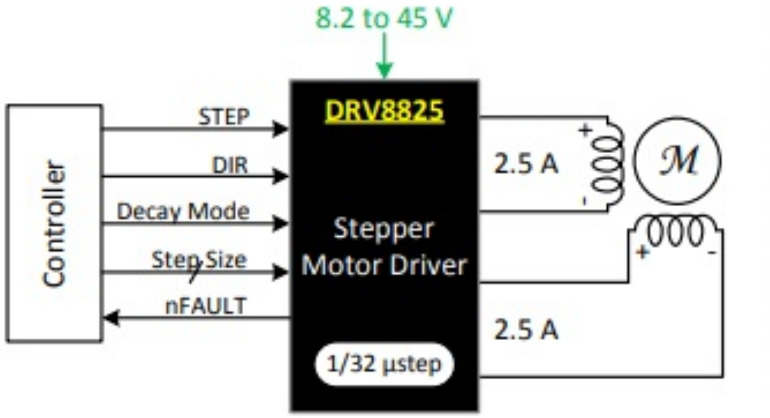
Os motores de passo são responsáveis por movimentar o suporte do sensor nas direções X e Y ao longo da estrutura do escâner. Foi escolhido o motor de passo pois com esse tipo de motor é possível ter um movimento mais preciso e controlado, para assim medir vários pontos da peça escaneada, trazendo uma melhor resolução.

De acordo com Sen (2013), um motor de passo é um motor que rotaciona a um específico número de graus mecânicos dada entrada de um número específico de pulsos elétricos, e é justamente por poder ajustar com precisão o passo mecânico do motor, que esses dispositivos são indicados para aplicações em impressoras, escâner e demais equipamentos de robótica, por exemplo.

Comumente, os motores de passo podem ser ajustados para passos mecânicos de 2°, 2.5°, 5°, 7.5° e 15° para cada pulso elétrico, variando de acordo com modelo e fabricante. Comercialmente é possível encontrar motores com 400 passos por revolução ou mais.

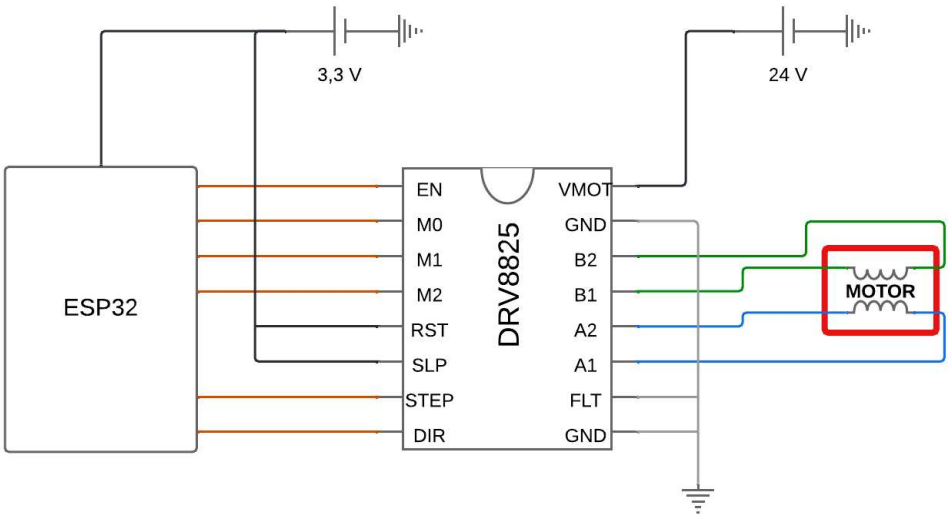
Esse motores de passo necessitam de um circuito de acionamento para controlar os passos, torque e direção de movimento do motor, e para este trabalho foi utilizado o driver $DRV8825$. De acordo com o fabricante Texas Instruments (2010), esse driver possui duas pontes H para acionamento e é indicado para motores de passo bipolares, e a Figura 36 mostra como é o circuito simplificado de ligação entre o driver e o motor de passo bipolar.

Figura 36 – Circuito simplificado da ligação do *driver* e os motores de passo.



Fonte: Retirado de Texas Instruments (2010).

Figura 37 – Circuito de ligação para controle de movimento dos motores.



Fonte: Do autor.

É mostrado na Figura 37 o esquema de ligação entre o motor, o *driver* e o microcontrolador, onde é possível fazer as seguintes observações:

- foi atribuído aos pinos $MODE0$, $MODE1$ e $MODE2$ nível lógico baixo LOW, para configurar o modo de operação em *full step*, de acordo com a tabela da Figura 38;
- o microcontrolador informa a direção de movimentação do motor pelo pino DIR, sendo esse nível lógico alto ou baixo, horário ou anti horário;
- o pino STEP é o pino que recebe a onda PWM correspondente aos passos elétricos.

A tabela da Figura 38 mostra que é possível configurar o modo de operação do motor, de forma que o passo mecânico é um múltiplo da quantidade de passos elétricos. Esse *driver* permite dividir 1 passo mecânico em até 32 passos elétricos. Para o escâner foi até testado outros modos de operação, mas não foi observado nenhuma vantagem operacional, por isso foi optado pelo modo de *full step*, ou seja, 1 passo elétrico gera um passo mecânico que gera 7,5 ° de giro, para esses motores.

Figura 38 – Tabela para configuração dos pinos do *driver* para os modos de operação do motor.

MODE2	MODE1	MODE0	STEP MODE
0	0	0	Full step (2-phase excitation) with 71% current
0	0	1	1/2 step (1-2 phase excitation)
0	1	0	1/4 step (W1-2 phase excitation)
0	1	1	8 microsteps/step
1	0	0	16 microsteps/step
1	0	1	32 microsteps/step
1	1	0	32 microsteps/step
1	1	1	32 microsteps/step

Fonte: Retirado de Texas Instruments (2010).

Para gerar a onda PWM no pino STEP para movimentar o motor foi utilizado um *timer* do $ESP32$. A PWM em questão se trata de uma onda de 50 % de *duty cycle* e o algoritmo para gerá-la, que foi embarcado no microcontrolador, é descrito abaixo:

1. primeiro configura o período do *timer* em μs;
2. depois informa a quantidade de pulsos para movimentar a distância desejada;
3. depois informa ao pino DIR a direção de movimentação;
4. seta a variável que dispara o *timer*;
5. *Timer* começa a contar;
6. cada vez que o programa entrar no *timer*, o estado lógico do pino $STEP$ é comutado, e uma variável contadora é incrementada;
7. assim que a variável contadora for igual ao número de pulsos desejados, parar o *timer*.

Dessa forma o *timer* trabalha como um gerador de onda PWM, pois o pino STEP muda de estado lógico com a frequência dada pelo período escolhido.

Em Lima (2020), o autor mostra as relações de equivalência de movimentação do sistema mecânico do escâner, que acontece de acordo com a seguinte lógica: a cada 3 passos mecânicos a correia que faz o movimento acontecer se desloca em 2 mm lineares. Então, com o motor girando 7,5 ° por passo mecânico e configurado em *full step*, existe a seguinte relação de igualdade entre as unidades:

$$3PE = 3PM = 22,5^{\circ} = 1PC = 2mm, \tag{3.1}$$

em que,

- PE: pulsos elétricos;
- PM: pulsos mecânicos;
- PC: passo da correia;
- 2 mm é de deslocamento linear.

Então, por exemplo, para movimentar o motor a 500 RPM, de acordo com a equação (3.1), o cálculo é o seguinte:

$$500RPM = 8,33RPS = 3000graus/s = 400PE = 400PM.$$

Dessa forma, para movimentar o motor com 500 RPM, é necessário gerar 400 pulsos elétricos por segundo, e para isso, basta configurar o *timer* que controla a geração PWM do motor para trabalhar com $T = 1/400s$ de período. Ainda segundo a equação (3.1), 500 RPM representa um deslocamento linear com velocidade de 266,67 mm/s.

3.1.2.2 Lógica de Interrupção de Movimento

O movimento dos motores é interrompido em duas situações: quando o número de pulsos desejados é atingido, ou quando há o contato entre o suporte do sensor e barra lateral do escâner. Nesse segundo caso, o sensor chegou ao máximo do movimento e o contato aciona uma chave fim de curso, que coordena a ação de desligar os motores e avisar ao microcontrolador que o sensor chegou ao fim do percurso.

Ao ser acionada, a chave fim de curso gera uma interrupção, como foi explicado na seção 2.4, e foi utilizada a seguinte lógica:

1. chave fim de curso ligada em normal fechado;
2. quando o motor chega ao limite físico de movimento, a chave é acionada por toque;
3. a ISR é programada para acionar por borda de descida;
4. ao comutar de fechada para aberta, a mudança de tensão da chave é detectada pela ISR e o motor é desabilitado.

Com a biblioteca do $ESP32$ é possível programar a interrupção utilizando a função "attachInterrupt"que tem como parâmetros, o número do pino onde ocorre a mudança de tensão, e o endereço de onde está declarado a ISR.

Por fim, essa mesma função que cria interrupção foi utilizada para gerar *delays* e coordenar a ação de acionar os motores. Dessa forma, a execução de um movimento longitudinal, por exemplo, só ocorre depois de algum tempo do término da execução de um movimento transversal.

Esse *delay* ajuda a suavizar o movimento do escâner como todo, e para conseguir gerar esses atrasos de tempo foi utilizada a seguinte lógica de interrupção:

1. ao detectar o fim do movimento de um motor, uma ISR é disparada;
2. dentro dessa ISR é acionada uma variável para disparar um *timer*;
3. assim que o *timer* atingir o *delay* desejado, uma variável dentro da ISR permite que o motor faça outro movimento ou que movimente outro motor, de acordo com a lógica.

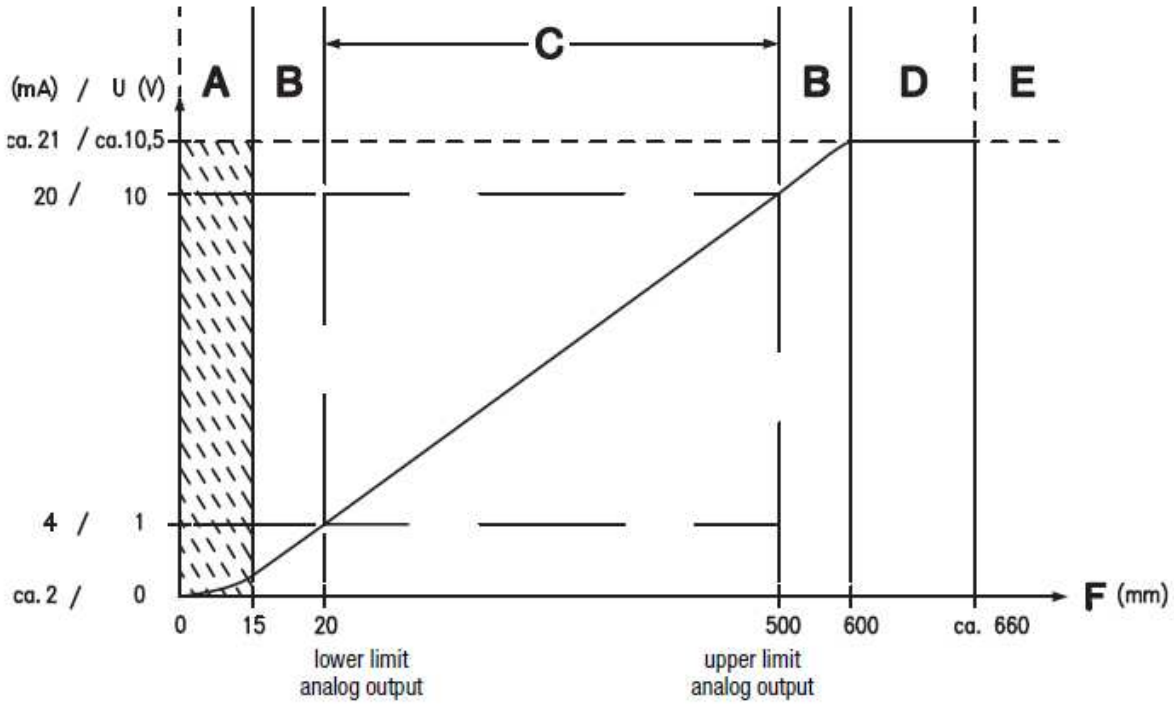
3.1.2.3 Circuito de Aquisição do Sensor Laser

O sensor laser $ODSL8$ foi escolhido para o projeto devido as suas características que possibilitam boa precisão de medição, e ele faz a medição de distância pelo método da triangulação, como foi discutido na seção 2.1.1. As especificações mais importantes desse sensor são:

- tipo de feixe de laser: luz vermelha visível;
- alimentação: 18 a 30 Vcc;
- taxa de medição: 2 a 7 ms;
- range de medição: 20 a 500 mm;
- resolução: 0,1 a 0,5 mm;
- saídas: 2 digitais de 1 a 10 V, e 1 analógica de 4 a 20 mA.

Um fator que pode afetar a precisão da medição é a distância em que o objeto está do sensor, e a curva de medição do sensor em função da distância do objeto a ser medido é mostrada na Figura 39. Analisando a curva, é possível afirmar que o objeto a ser escaneado neste projeto deve ficar a, no máximo, 500 mm e, no mínimo, 20 mm de distância do sensor, assim a medição é feita dentro da faixa linear do sensor, onde as regiões A e B representam faixas não lineares da curva de medição. Nas regiões D e E a medição é também imprecisa ou impossível de ser feita, pois o alcance máximo do feixe é de 500 mm. Como foi mostrado na Tabela 1, a máxima distância entre entre o sensor e a base do escâner é de 430 mm, garantindo que qualquer objeto fique dentro da faixa de medição linear do sensor.

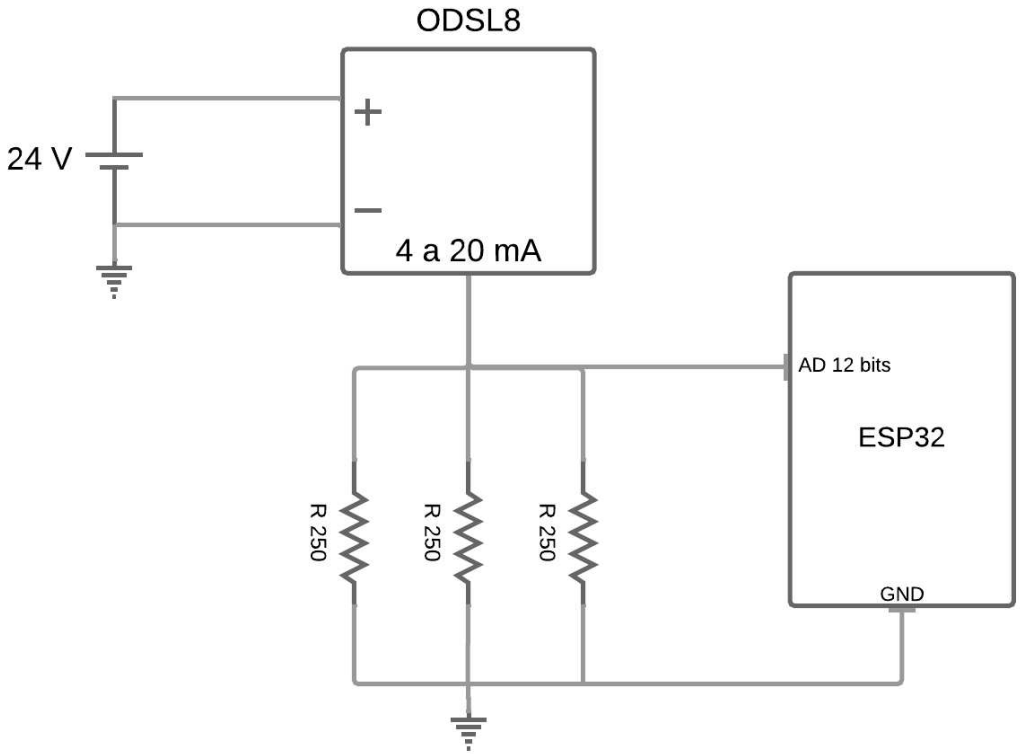
Figura 39 – Curva de medição do sensor laser ODSL8 em função da distância até o objeto medido. Escala vertical do gráfico representa as saídas em corrente ou tensão do sensor e a escala horizontal e medição em mm.



Fonte: Retirado de Leuze Eletronics (2015).

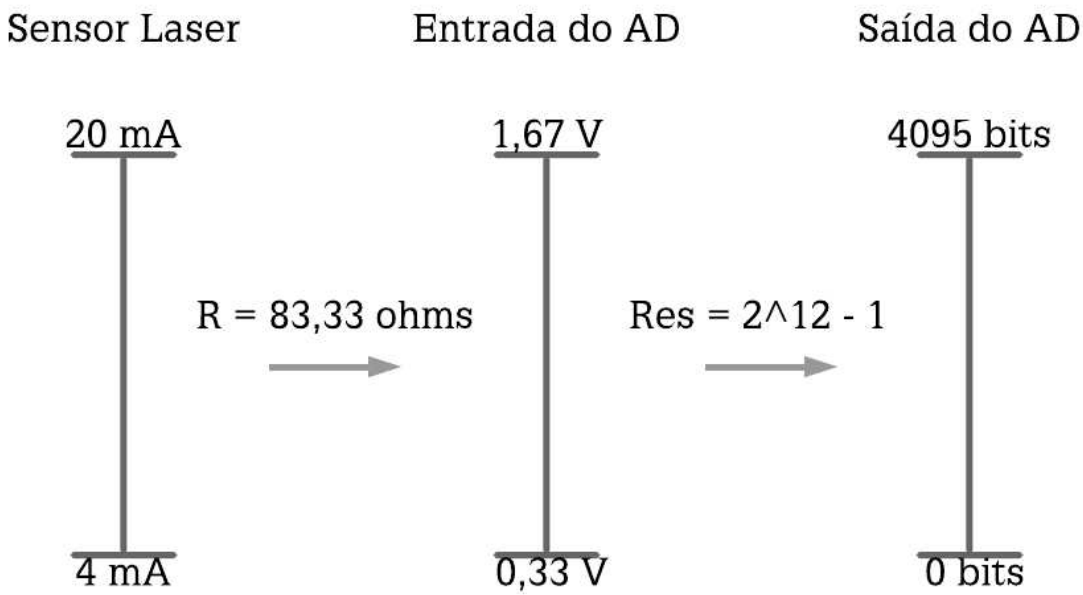
Dentro do circuito da placa PCI o sensor é conectado ao conversor analógico digital do $ESP32$ de acordo com circuito mostrado na Figura 40. Como já foi mencionado, o conversor AD do $ESP32$ é de 12 bits e de acordo com a equação (2.4) isso significa que cada valor de corrente de 4 a 20 mA é convertido em uma escala de inteiros de 0 a 4095. No circuito da Figura 40 foi utilizado uma resistência equivalente de $R = 83,33\Omega$ para converter o sinal de corrente do sensor de 4 a 20 mA para um sinal de tensão de 0,33 a 1,67 V, pois o pino de conversor AD do $ESP32$ aceita somente sinal de tensão, e por isso foi utilizado resistores de precisão de 1 % para não perder acurácia na leitura. A escala de equivalência da medida desde o sensor laser até o conversor AD é mostrado na Figura 41.

Figura 40 – Esquema de conexão do sensor laser e do conversor analógico do $ESP32$.



Fonte: Do autor.

Figura 41 – Equivalência de escalas.



Fonte: Do autor.

Depois de estabelecida a conexão física entre o sensor e o conversor analógico digital do $ESP32$, foi implementado um algoritmo para fazer a aquisição dos dados, e para isso foi utilizado também um *timer* do $ESP32$, e ele funciona da seguinte forma:

1. configura o período do *timer*, que é o período de amostragem do conversor analógico digital;
2. depois um *flag* é iniciado e é disparado o início da leitura;
3. cada vez que o programa entra no *timer*, armazena em um *buffer* as leituras obtidas no conversor AD;
4. por fim, um *flag* para finalizar o *timer* é iniciado.

Assim, as leituras só são coletadas no pino do conversor AD quando o *timer* é setado, garantindo que cada leitura seja feita com a mesma frequência de amostragem. Os *flags* que setam e resetam o *timer* são baseados na movimentação do motor que move o suporte do sensor. Ou seja, os dados só são coletados quando o sensor se movimenta, caso contrário, a leitura fica aguardando em *stand by*.

3.1.2.4 Definição da Frequência de Amostragem do Conversor AD

O próximo passo foi determinar a frequência de amostragem do conversor AD. O fabricante do sensor ótico $ODSL8$ indica que a taxa de medição dele é entre 2 a 7 ms. Dessa forma, para respeitar as taxas de Nyquist e evitar o *aliasing*, para o caso em que o sensor medir mais rápido (em 2 ms), a frequência de amostragem do sensor é:

$$F_{sensor} = \frac{1}{T_{sensor}} = \frac{1}{2ms} = 500Hz.$$

De acordo com a equação (2.2), a frequência de amostragem deveria ser de:

$$F_s = 2F_{sensor} = 1kHz.$$

Foi analisado no *datasheet* do $ESP32$ e foi verificado que a taxa de amostragem de $1KHz$ está dentro da faixa de frequência suportada pelo conversor AD do microcontrolador. Então foi decidido que a frequência de amostragem padrão do escâner seria de $F_s = 1kHz$.

Para configurar a frequência de amostragem basta configurar o período do *timer* responsável por disparar a leitura do conversor AD. Foi realizado o seguinte teste para validar a correta aquisição de dados na frequência escolhida:

1. o sensor foi colocado para medir, de forma estática, sem se movimentar, a distância até um mesmo ponto de uma superfície plana (base do escâner), na frequência de 1 kHz.

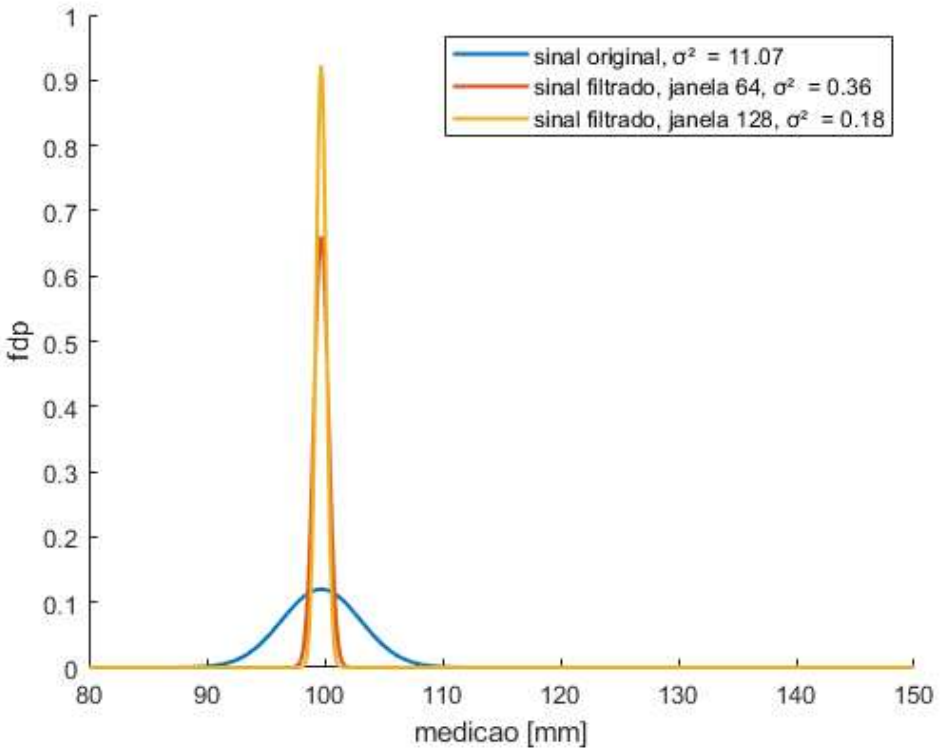
2. depois foi analisada a média e a variância da medição (através da função de densidade de probabilidade), assim como a diferença entre os valores máximos e mínimos medidos (range de medição). Idealmente esses valores deveriam ser próximo de zero.

3. para as duas análises foi aplicado um filtro de média não causal de diferentes tamanhos de janela para experimentar o efeito da filtragem sobre o sinal amostrado.

As Figuras 42 e 43 mostram o resultado dessa análise, provando que a frequência de amostragem de 1 kHz escolhida atende, e que um filtro de média não causal resolve facilmente alguns ruídos de medição no conversor AD. Isso porque é possível observar que as medições convergem para uma variância e diferença nula (que é o ideal) quando o sinal é filtrado.

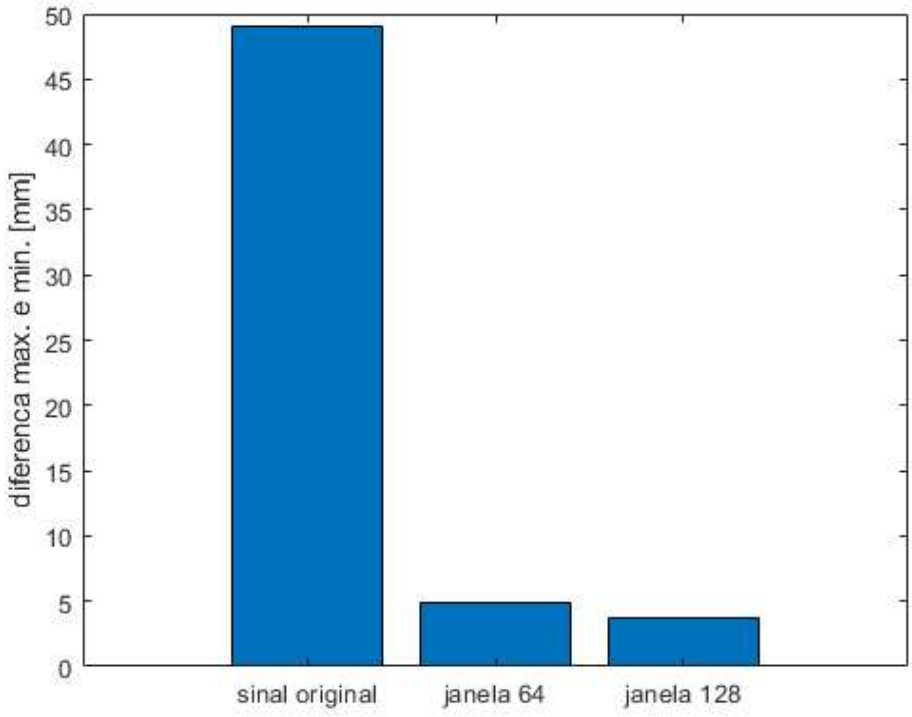
Vale fazer uma ressalva aqui, dizendo que o filtro de média é uma boa solução neste caso pois para a aplicação proposta essa técnica de filtragem não afeta o desempenho do sistema de classificação. Em alguns casos acontece do sinal adquirido ser suavizado pelo filtro a ponto de retirar dele características específicas essenciais para a sua caracterização, como transições bruscas de amplitude, por exemplo.

Figura 42 – Analise da função de densidade de probabilidade das medição com o conversor amostrando a $1kHz$. Análise também para o sinal filtrado.



Fonte: Do autor.

Figura 43 – Analise da diferença de amplitude máxima e mínima da medição feita com o conversor amostrando a $1kHz$. Análise também para o sinal filtrado.



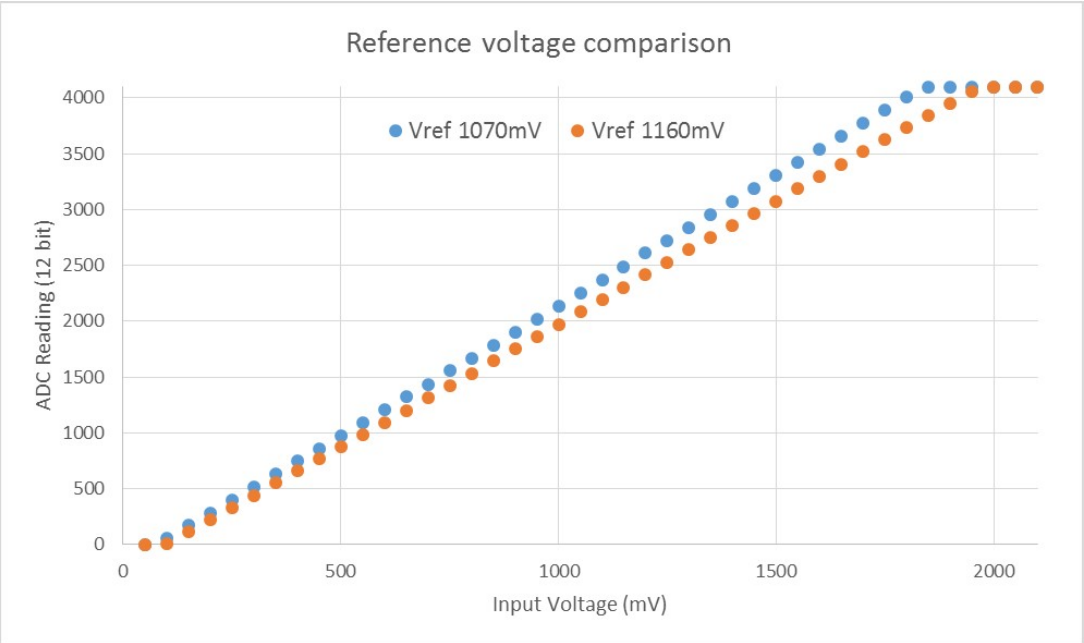
Fonte: Do autor.

Um dos motivos desses erros de medição é devido ao conversor AD do $ESP32$, onde o fabricante alerta o usuário para um possível erro de medição do conversor AD causado por uma não linearidade da tensão V_{ref} do microcontrolador (ESPRESSIF SYSTEMS, 2018).

Na folha de dados do conversor do $ESP32$ é apresentado um estudo que mostra a discrepância de medição entre dois modelos diferentes de $ESP32$, e as curvas de medição de cada modelo é mostrada na Figura 44, onde é possível observar o efeito da tensão V_{ref} na diferença de medição.

A resistência equivalente de entrada do pino do conversor AD de $83,33\Omega$ foi escolhida para que as tensões medidas estivessem dentro da faixa de 333 e 1667 mV, indicada pelo fabricante. Mas, ainda assim, para contornar o problema, o fabricante indica utilizar a biblioteca "esp_adc_cal.h", que faz uma calibração da tensão do conversor AD, trazendo a medição para uma faixa linear, além da utilização da função "ADC_ATTEN_DB_6"que ajusta a atenuação em dB do conversor para medir tensões dentro dos valores da faixa.

Figura 44 – Gráfico ilustrando o efeito de diferentes tensões de referência na curva de tensão do conversor AD.



Fonte: Retirado de Espressif Systems (2018).

3.1.3 Armazenamento dos Dados

Para fazer o armazenamento de dados foi utilizado um cartão SD, que dispositivos de memória *flash* removíveis de tamanho reduzido, capazes de armazenar uma quantidade relativamente grande de dados dependendo da aplicação embarcada. Sua integração a um circuito eletrônico é relativamente simples, e a pinagem é facilmente identificada.

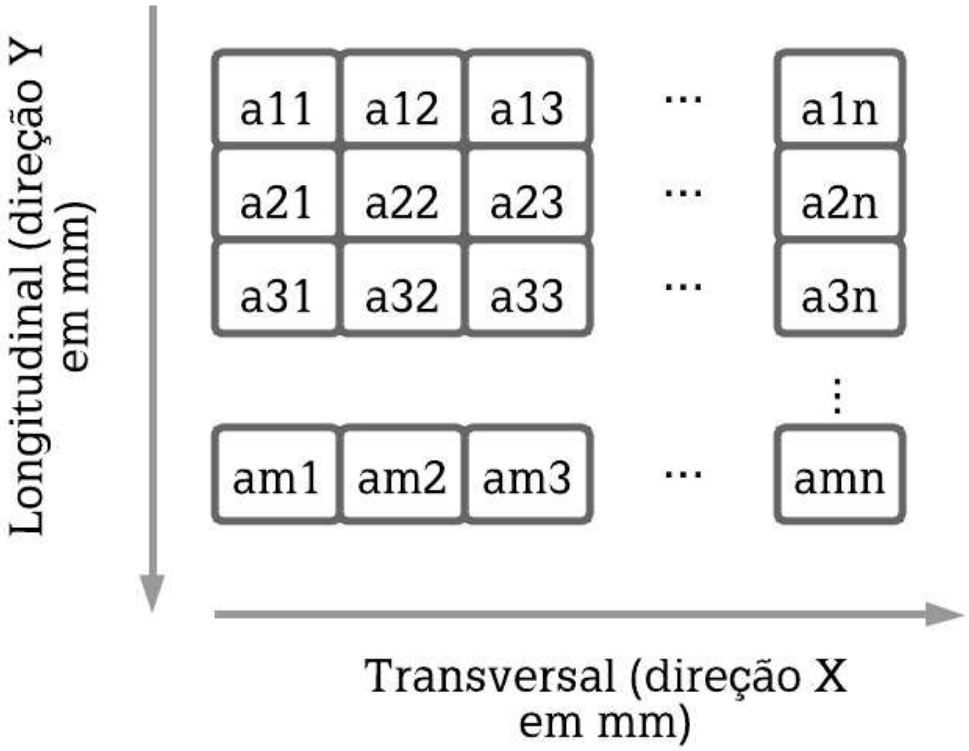
Neste projeto, a função desse cartão é armazenar os dados das leituras dos sensores em arquivos. A grande vantagem é que o acesso aos dados armazenados nessa memória pode ser feita tanto de forma manual, ou seja, retirando o cartão da placa e passando para outro dispositivo, quanto de forma sem contato, uma vez que é possível que o $ESP32$ acesse o cartão de memória e envie os dados via *bluetooth* ou por Wi-Fi. Para salvar os dados o cartão SD comunica com o microcontrolador através do protocolo serial SPI, que já foi apresentado na seção 2.4.2, e os dados podem trafegar tanto do microcontrolador para o cartão quanto pelo caminho inverso.

Uma vez que a lógica de leitura já está configurada no $ESP32$, e as correções das medições já foram aplicadas no momento da aquisição, basta transferir os dados que ficaram temporariamente salvos no *buffer* de memória do microcontrolador e salvá-los no cartão SD.

Para salvar no cartão foi escolhido o formato de arquivo ".txt". Esse arquivo contém as medições de distância captadas pelo sensor laser, e foi montado de acordo com a lógica de movimentação do escâner. Como já foi explicado, o suporte do sensor se movimenta nas

direções X e Y, transversal e longitudinal. Dessa forma, ele varre uma região retangular, conhecida como *grid*. A Figura 45 mostra como é montado esse *grid*.

Figura 45 – *Grid* de movimentação do sensor laser.



Fonte: Do autor.

Para ficar mais claro, a lógica de movimentação do sensor é a seguinte:

1. o sensor é posicionado no ponto a_{11} do *grid*;
2. o sensor começa a se movimentar na direção X e a leitura é disparada até o ponto a_{1n}, em que n é o número de pontos amostrados;
3. a leitura é finalizada;
4. depois o sensor se movimenta na direção Y e em seguida se posiciona no ponto a_{21};
5. a leitura é feita novamente na direção X;
6. processo se repete até completar as m leituras definidas pelo usuário.

Portanto, o número de leituras m é um parâmetro definido pelo usuário, enquanto o número de pontos coletados n depende da frequência de amostragem do conversor AD e da velocidade de movimentação dos motores, que é definido em código. Posteriormente,

na seção 4.1.1, será explicado como foi encontrado o melhor valor de velocidade de movimentação dos motores, e a frequência de amostragem já foi definida como 1 kHz.

Dessa forma, a varredura do sensor corresponde a um *grid*, e é uma matriz $A_{m\,X\,n}$, e que o arquivo ".txt" é montado como uma matriz também $A_{m\,X\,n}$. Ou seja, cada linha do arquivo é uma linha da matriz, e cada elemento da matriz é um ponto de leitura.

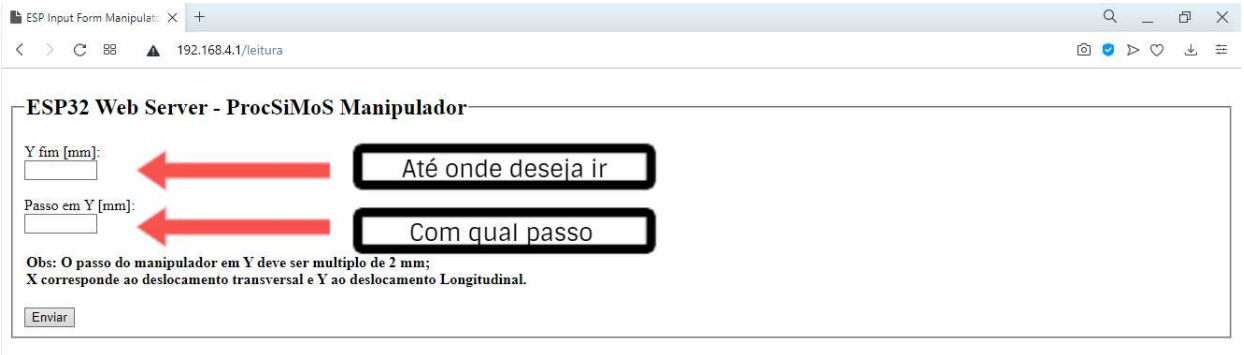
Quando cada linha de leitura é finalizada, os valores armazenados no *buffer* da memória interna do $ESP32$ são salvos em uma linha do arquivo texto. Esses valores são acessados por técnicas de Acesso Direto à Memória, do inglês *Direct Memory Acess* (DMA) e são transferidos ao cartão SD pelo protocolo serial SPI. Um arquivo texto é pré alocado dentro do cartão SD toda vez que uma nova leitura é iniciada pelo escâner, e tudo isso é programado com as bibliotecas "SD.h"e "SPI.h"compatíveis com o $ESP32$, e os detalhes do código são mostrados no Apêndice A.

Outro detalhe é que antes de salvar os dados, eles são convertidos para distância em mm, pois interessa mais ao usuário saber de fato a distância medida pelo sensor do que a quantidade de bits do conversor AD. Para isso, foi definido como referência de medição a base do escâner, ou seja, o chão. Depois, foi medido um outro objeto qualquer para fazer uma relação de transformação. Com dois pontos de referência, o chão e um objeto qualquer, é possível transformar uma escala de bits em mm. Dessa forma, foi padronizado que todos os arquivo texto salvos no cartão SD estejam na escala de distância em mm.

3.1.3.1 *Web Server*

Para conseguir definir o *grid* de leitura do escâner e acessar os dados salvos no cartão de memória, foi implementado um *Web Server* que faz a interação com o usuário através de uma rede Wi-Fi, e a Figura 46 mostra os parâmetros que o usuário deve inserir no *Web Server* para definir a movimentação do escâner.

Figura 46 – Página do *Web Server* em que o usuário define a lógica de movimentação longitudinal do escâner.



Fonte: Do autor.

Como foi mencionado, o usuário controla apenas a definição da direção longitudinal, enquanto a transversal é padrão de código.

3.2 Sistema de Identificação de Falhas

Tendo como ponto de partida que todas as funcionalidades do escâner 3D foi implementada, como foi mostrado na seção anterior, agora chega a etapa de expor as técnicas de processamento de dados utilizadas para compor o sistema de identificação de falhas nas peças metálicas, como proposto. E é sobre isso que se trata esta seção.

3.2.1 Levantamento da Base de Dados

A base de dados foi levantada com o escâner configurado com as especificações mostradas na Tabela 2.

Tabela 2 – Configurações do escâner 3D após testes.

Frequência de amostragem do conversor AD	1,0 kHz
Velocidade de rotação dos motores (transversal e longitudinal	500 RPM
Velocidade de deslocamento linear do suporte do sensor	266,67 mm/s
Resolução do sistema de aquisição	entre 4 e 8 mm

Fonte: Do Autor.

Partindo desse ponto, foi montado uma base de dados contendo peças metálicas, como a ilustrada na Figura 2. Foram escolhidas 10 peças com defeito de trinca e 10 peças saudáveis, aproximadamente planas, para compor a base.

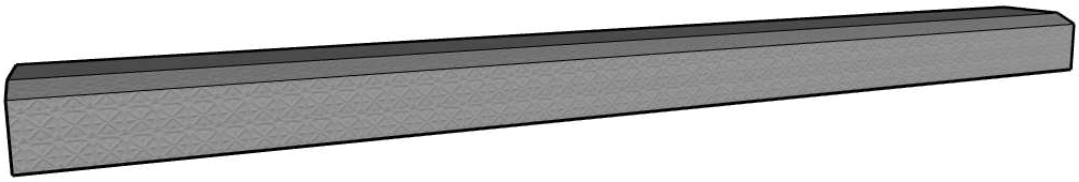
O primeiro passo foi definir qual seria o *grid* de leitura do escâner, justamente para padronizar a área de leitura para todas as peças. Essas peças metálicas têm 2100 mm de comprimento e cerca de 150 mm de largura. Valores dentro da faixa de varredura do escâner. Então o *grid* foi definido da seguinte forma:

- na direção X (transversal): de 0 a 640 mm;
- na direção Y (longitudinal): de 0 a 2150 mm.

Com a frequência de amostragem do sistema de aquisição de $1kHz$ e o suporte do sensor se movimentando a 266,67 mm/s, cada amostra em X é separada de 0,267 mm, o que é padrão de código, como já foi discutido. O que foi definido foi a quantidade de leituras feitas: na direção Y foi definido uma varredura até 2150 mm com um passo de 50 mm, totalizando 43 leituras. Então para cada peça da base de dados, se tem um arquivo texto correspondente. Cada arquivo é uma matriz de 43 linhas e 4800 colunas. É com essa matriz de dados e com esses 20 arquivos que as análises seguintes serão feitas.

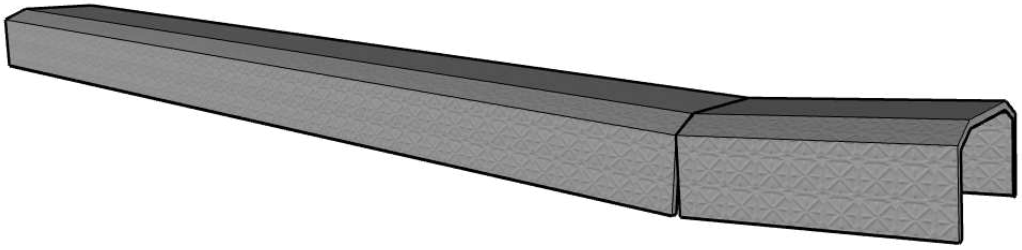
Dentre as 20 peças metálicas escolhidas, é possível fazer uma comparação entre o perfil de uma peça saudável e de uma peça com defeito, através das Figuras 47 e 48, onde fica perceptível que a peça que apresenta o defeito de trinca tem uma curva de perfil não plana.

Figura 47 – Representação tridimensional do perfil de uma peça metálica saudável.



Fonte: Do autor.

Figura 48 – Representação tridimensional do perfil de uma peça metálica defeituosa.



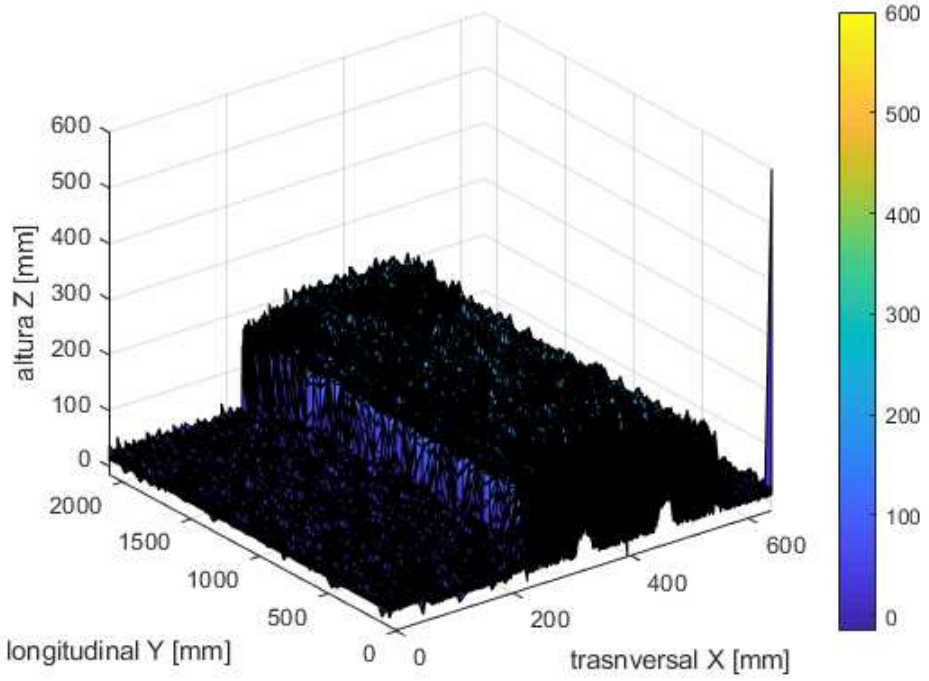
Fonte: Do autor.

3.2.2 Tratamento dos Dados

Como primeiro passo para o tratamento da base de dados, foi aplicado um filtro de média não causal de janela de 128 amostras para suavizar os ruídos de alta frequência de aquisição do sinal, para todas os dados das 20 peças escaneadas.

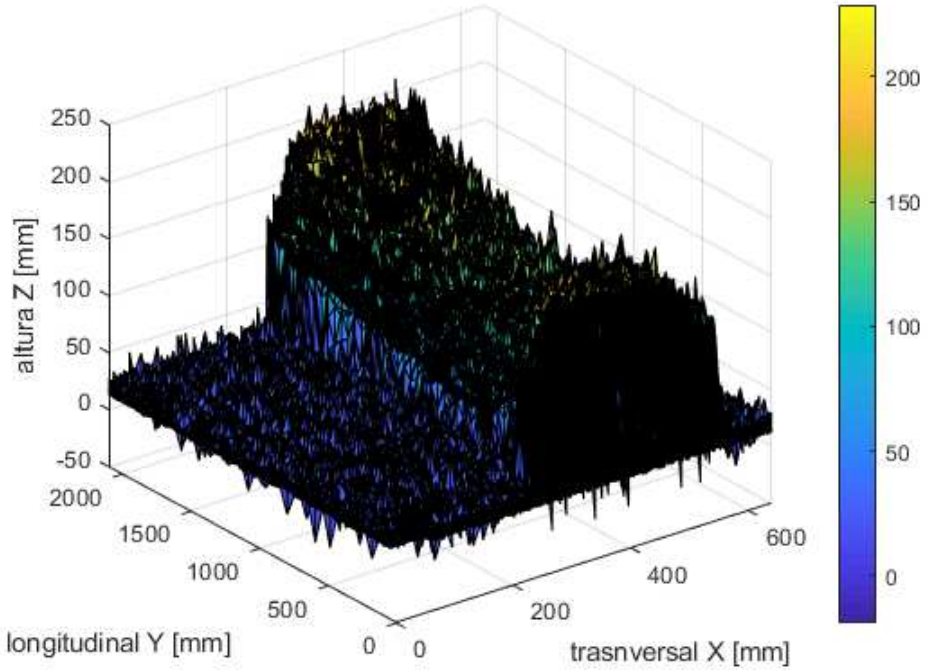
As Figuras 49, 51, 50 e 52 fazem um comparativo dos sinais originais e filtrados para os dois tipos de peças presentes na base de dados. É possível observar uma suavização considerável nos ruídos de aquisição, e como já foi discutido na seção anterior, o filtro de média é uma boa solução para esse problema.

Figura 49 – Peça de aço saudável, sinal original.



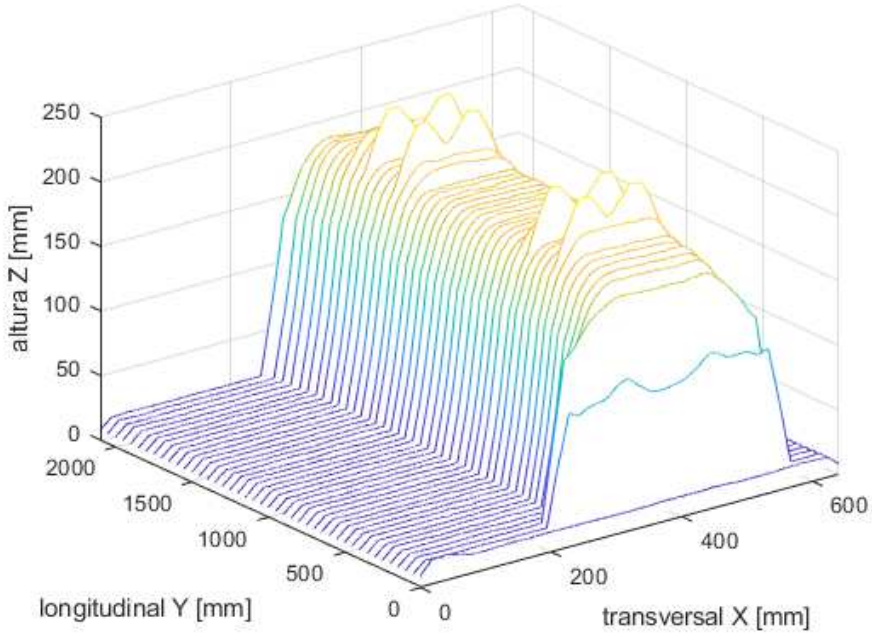
Fonte: Do autor.

Figura 50 – Peça de aço defeituosa, sinal original.



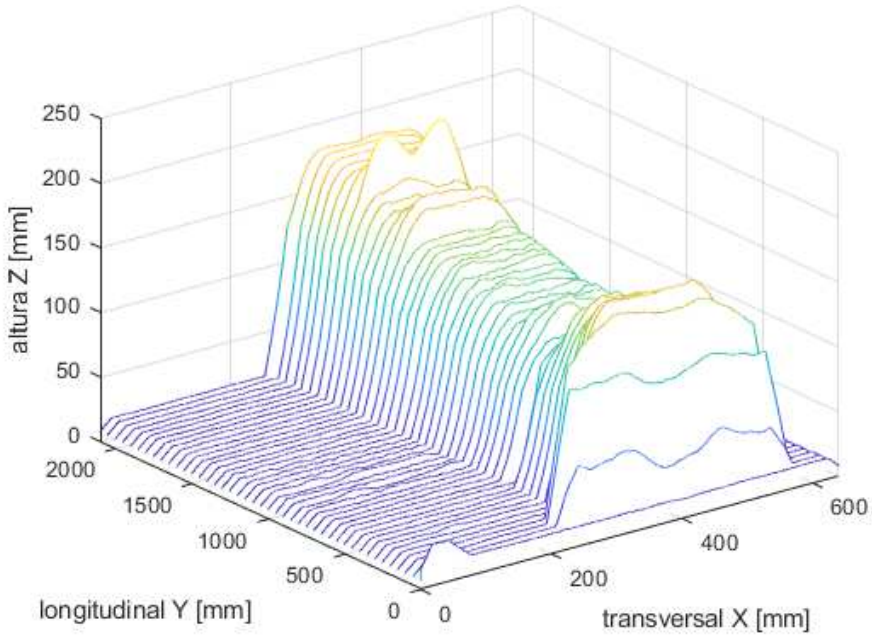
Fonte: Do autor.

Figura 51 – Peça de aço saudável, sinal tratado com filtro de média não causal de janela de tamanho 128.



Fonte: Do autor.

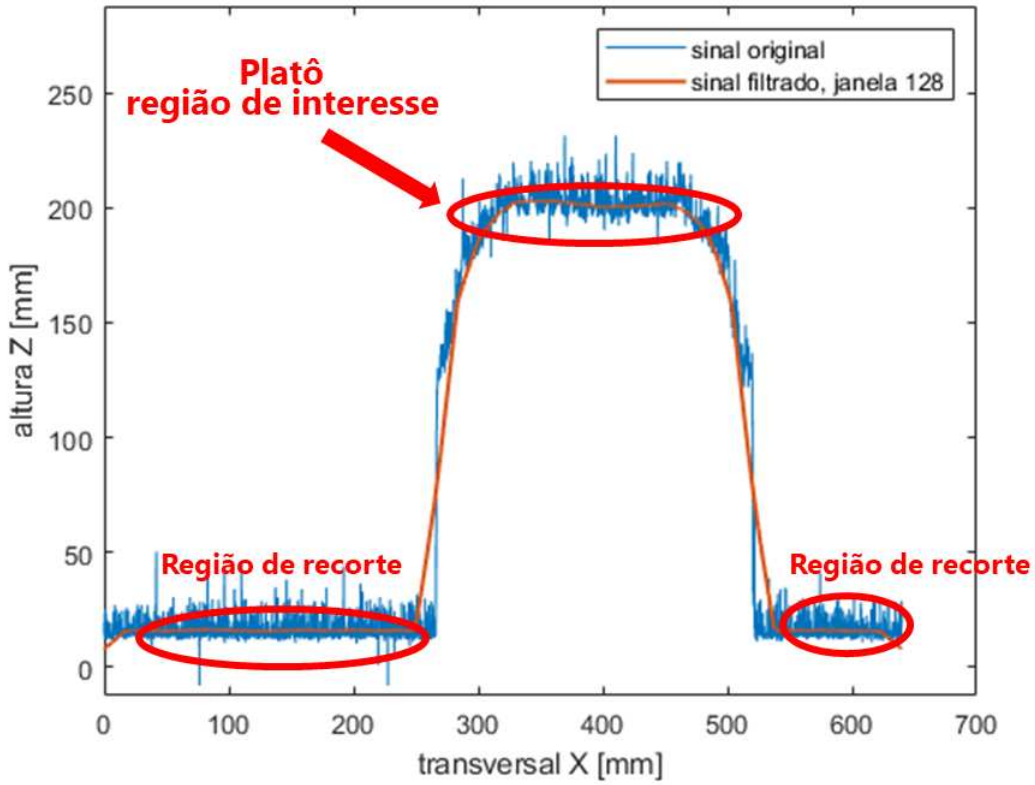
Figura 52 – Peça de aço defeituosa, sinal tratado com filtro de média não causal de janela de tamanho 128.



Fonte: Do autor.

A Figura 53 mostra um corte transversal da leitura feita em uma dessas peças, onde os dados de cada peça, é composta por 43 desses cortes transversais. É colocado em destaque nessa Figura as regiões de interesse do sinal e as regiões que podem ser eliminadas.

Figura 53 – Corte transversal da leitura da peça metálica.



Fonte: Do autor.

Como indicado na Figura 53, a única região de interesse do sinal é o platô superior entre 300 e 500 mm da direção transversal. Esse região é que melhor caracteriza a peça e é o local onde que caracteriza a presença ou não de defeito. Para eliminar essas regiões de não interesse, foi utilizado um código para identificação desse platô por meio da função de energia do sinal, como já foi discutido na seção 2.5.

Após isso, foi feito uma média dos pontos do platô para cada corte transversal, resultando, para cada peça, em um vetor de 43 pontos longitudinais por 1 ponto transversal. Em outras palavras, toda aquela região de interesse no corte transversal foi reduzida a um único ponto por média. Dessa forma, foi possível traçar um perfil longitudinal das peças metálicas como é mostrado na Figura 54. Lembrando que o perfil longitudinal deveria se aproximar de uma reta plana.

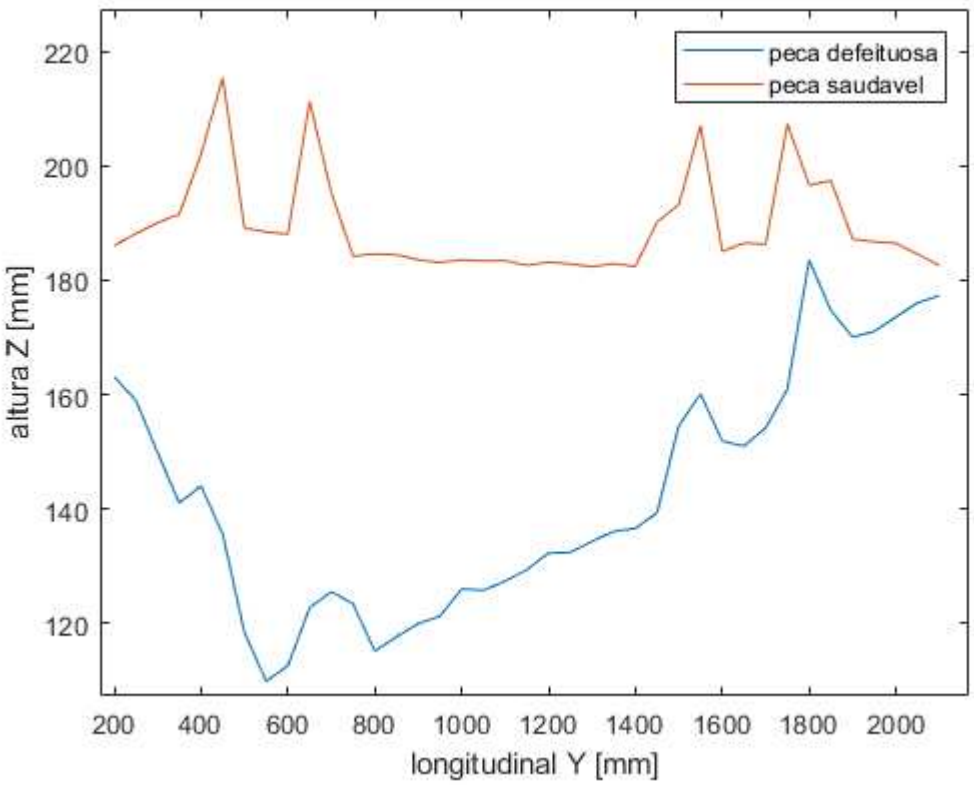
Analisando o perfil das peças, é possível visualizar um problema para a caracterização dos dados. Essas espécies de "chifres"são pontos da estrutura da peça metálica que

não trazem informações sobre a presença de defeito ou não. Dessa forma, foi utilizado um algoritmo para identificar esses pontos em todas as peças da base, e a interpolação foi necessária para substituir os dados correspondente a esses "chifres"que foram eliminados.

Então para eliminar esses "chifres"do perfil da curva foi utilizado um algoritmo de identificação de picos baseado na função de energia do sinal. Uma vez identificado, esses pontos foram retirados da matriz de dados, e foi utilizado uma interpolação do tipo *spline* para reconstruir a curva. Mais detalhes sobre esse tipo de interpolação pode ser encontrado em Ruggiero e Lopes (1996) e Barroso (1987).

Foi definido um novo período espacial, e a partir de então o perfil da curva teria 180 pontos separados por 10 mm cada, ao invés de 43 pontos separados por 50 mm. Com isso, o novo perfil da peça passar a ter 1800 mm de distância total ao invés de 2150 originais. Com a eliminação dos "chifres"a curva de interesse passou a ter um tamanho menor.

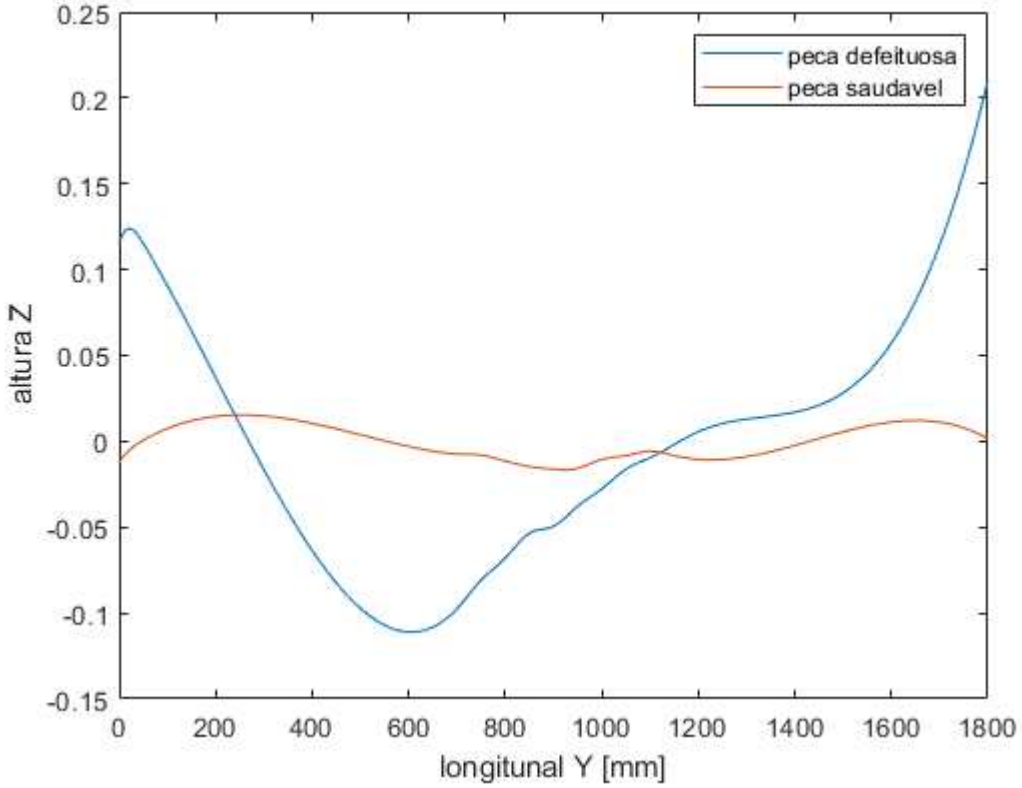
Figura 54 – Comparativo do perfil longitudinal de uma peça saudável e de uma peça defeituosa. Obtido fazendo a média do platô nos 43 cortes transversais. Sem eliminação dos "chifres".



Fonte: Do autor.

Na Figura 55 é mostrado o comparativo do perfil longitudinal de uma peça saudável e de uma peça defeituosa após eliminação dos "chifres". É possível observar, se comparar com a Figura 54, a eliminação dos picos.

Figura 55 – Comparação do perfil longitudinal de uma peça metálica saudável e uma defeituosa após eliminação dos "chifres"e interpolação dos dados.



Fonte: Do autor.

Portanto, em resumo, os dados coletados das 20 peças passaram pelo seguinte processo:

1. foram coletados pelo escâner em um grid de 2100 mm por 150 mm com $1kHz$ de amostragem;
2. filtrados por um filtro de média não causal de tamanho de janela de 256 amostras;
3. foram eliminados os dados que não estavam dentro da região do platô da peça;
4. foram eliminados os dados correspondentes aos "chifres", como mostra a Figura 54;
5. os dados dos "chifres"eliminados foram interpolados pela técnica *spline* e ficaram como mostra a Figura 55.

Dito isso, a partir de agora, a nova matriz de dados é da forma como mostrado na Figura 55, em que cada peça é uma curva longitudinal de 180 pontos. Foi a partir dessas curvas que foi aplicado o algoritmo de classificação, como será explicado a seguir.

3.2.3 Configuração do Sistema de Classificação

Com os dados das 20 peças tratados como mostrado anteriormente, o *design* do sistema de classificação foi configurado como mostra a Tabela 3.

Tabela 3 – *Design* do sistema de classificação.

tipo de treinamento	supervisionado
extração de características	desvio padrão do sinal curvatura do sinal amplitude máx. menos mín.
método de validação	validação cruzada 70 % dados de treinamento 30 % dados de teste
classificador	linear

Fonte: Do Autor.

O treinamento é dito supervisionado pois a base de dados foi rotulada, ou seja, cada peça escaneada tem uma identificação se é saudável ou defeituosa.

A validação cruzada quer dizer que os dados de 14 peças foram utilizadas para treinar o modelo de classificação e os dados de 6 peças foram utilizados para testar o modelo gerado e obter o desempenho.

Os modelos foram levantados baseado em três características dos sinais:

- Parâmetros estatísticos: média e desvio padrão do sinal;
- Curvatura: computada a curvatura média do sinal;
- Diferença de amplitude: a diferença entre o maior e o menor valor de amplitude do sinal.

Para o classificador foi utilizado um classificador linear em que a reta que separa o plano de *clusters* é um valor constante no eixo das ordenadas. Esse valor que separa os *clusters* de peça defeituosa e peça saudável foi calculado da seguinte forma:

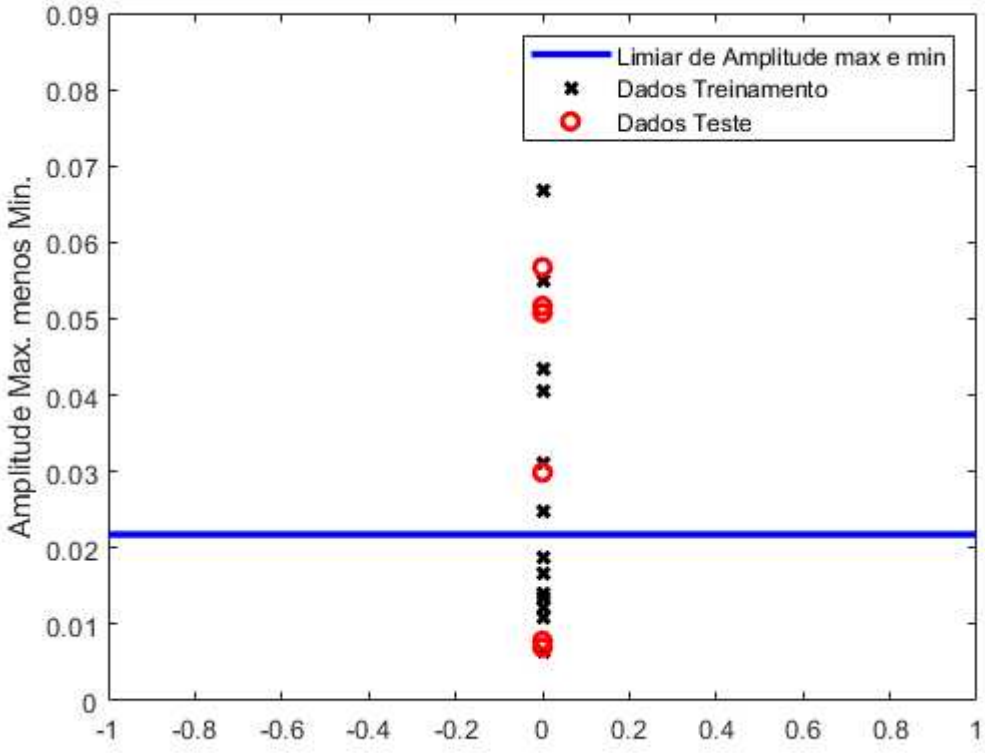
- as características dos dados das 14 peças de treinamento foram separadas;
- dentre as 14 características é feito uma média da maior e da menor, em amplitude.

Então o classificador é basicamente um limiar que separa os dados, e todos os dados que estão de acima do limiar são classificados na classe de defeituosos e os que estão abaixo na classe de saudáveis.

A Figura 56 mostra o limiar calculado para a característica de diferença de amplitude máxima e mínima, assim como as Figuras 57 e 58 mostram o limiar para desvio padrão e curvatura, respectivamente. Perceba que o limiar é a curva que separa as classes,

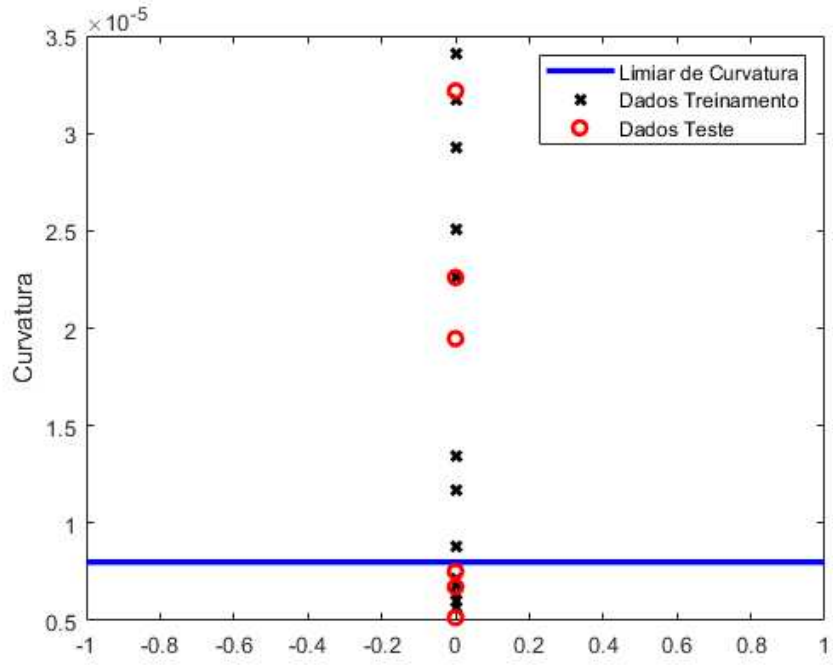
e que os dados de teste inseridos se encaixam em um classe ou outra, abaixo ou acima do limiar. Logo, a intenção da apresentação destas figuras é de mostrar os dados sendo separados pelo limiar.

Figura 56 – Limiar de classificação para o modelo baseado na diferença de amplitude do sinal.



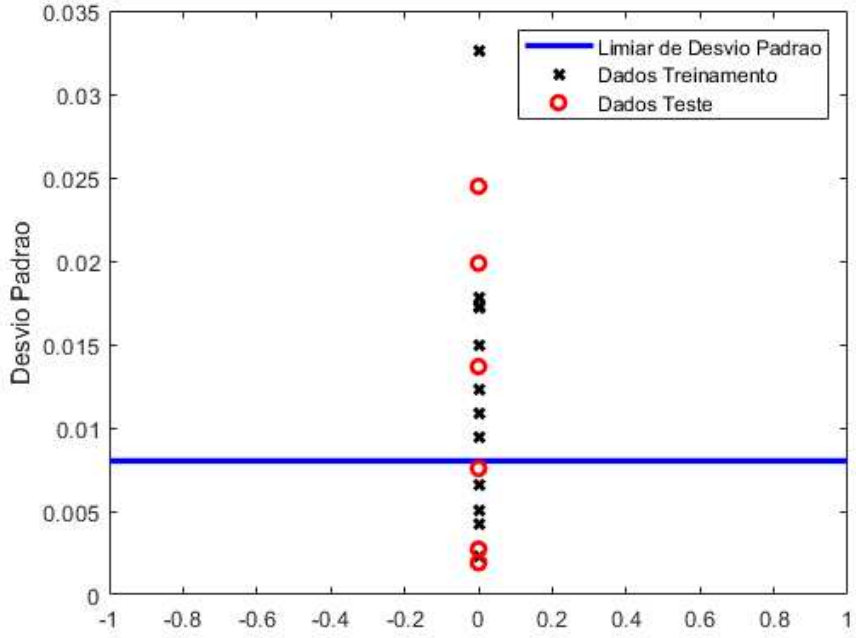
Fonte: Do autor.

Figura 58 – Limiar de classificação para o modelo baseado na curvatura do sinal.



Fonte: Do autor.

Figura 57 – Limiar de classificação para o modelo baseado no desvio padrão do sinal.



Fonte: Do autor.

4 Resultados

Será apresentado aqui neste capítulo os resultados obtidos pelas duas abordagens mostradas na metodologia: os resultados referentes ao projeto do escâner e suas implicações, além do resultado do desempenho do sistema de identificação de falhas.

Assim, será exposto os testes realizados para a validação do escâner, bem como as premissas e configurações adotadas, além da metodologia de identificação das peças defeituosas presentes na base de dados levantada.

4.1 Testes de Validação do Escâner

4.1.1 Teste de Velocidade de Movimentação dos Motores

Esse teste foi feito para buscar padronizar a velocidade de movimentação dos motores do escâner. Esse parâmetro é ajustado por código, pois basta mudar a frequência do gerador PWM do *driver*, como é mostrado na seção 3.1.2.1.

A importância de padronizar a velocidade é porque a quantidade de pontos amostrados pelo sistema de aquisição depende desse valor, e é ideal que esse valor seja o mesmo para todos os experimentos realizados.

Apesar da frequência de amostragem do conversor já ter sido definida, uma outra questão ainda havia de ser resolvida: no funcionamento do escâner, o sensor faz a aquisição dos dados enquanto está em movimento retilíneo e constante, dessa forma, até que ponto a velocidade de movimentação do motor que movimenta o suporte do sensor afeta a medição?

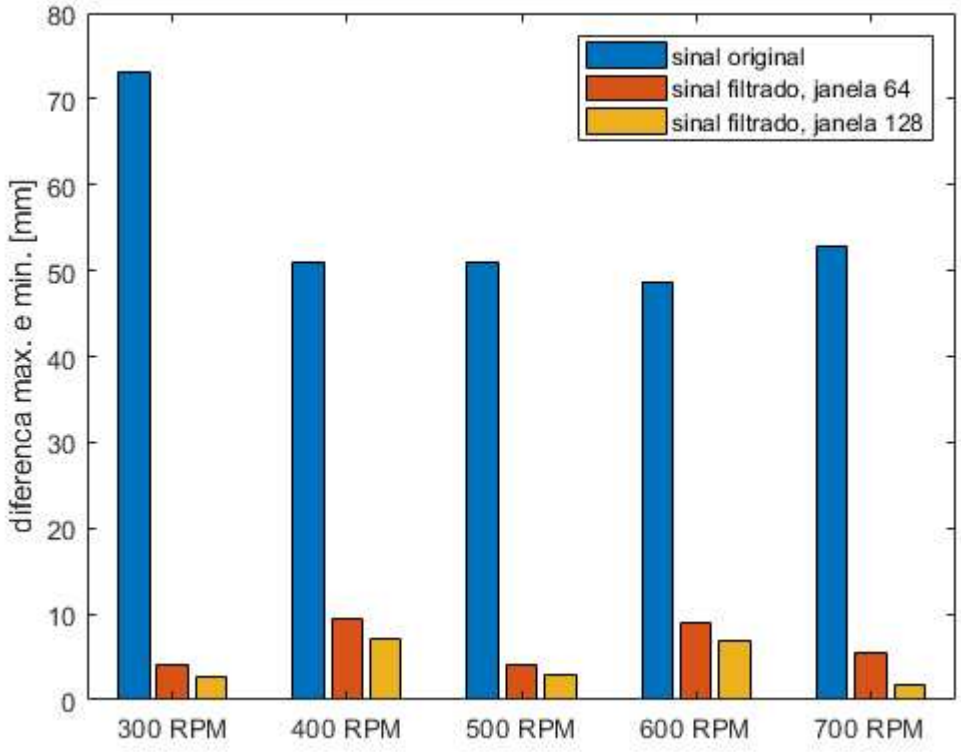
Para responder essa pergunta foi feito o seguinte teste:

1. O sensor foi colocado para se movimentar a diferentes velocidades.
2. Foi feito a leitura de pontos de uma superfície plana (base do escâner) com a taxa de amostragem de $1kHz$. Idealmente deveria ser uma uma reta plana.
3. Depois foi feito a análise do range de medição, da média e da variância para as diferentes velocidades testadas.
4. Foi aplicado também um filtro de média não causal aos sinais amostrados com diferentes tamanhos de janela.

As Figuras 59 e 60 mostram o resultado desse teste. É fácil perceber que a velocidade não afeta a medição do sensor laser e nem a aquisição do sistema conversor analógico digital. É possível notar também que um simples filtro de média não causal

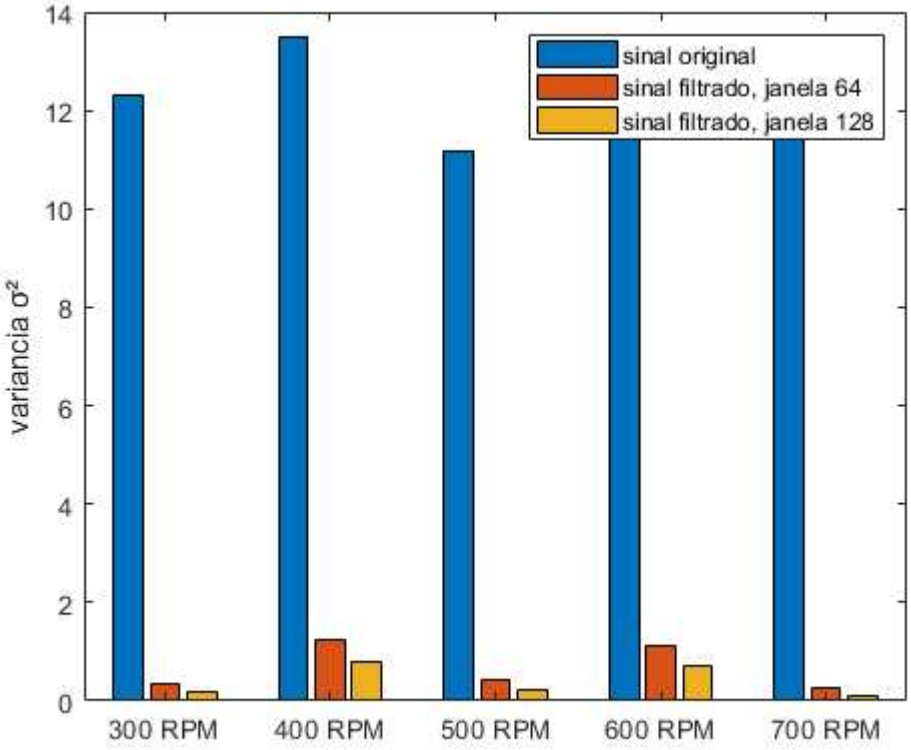
consegue diminuir bastante o range de medição. Para uma medição sobre uma superfície plana, o ideal seria um range nulo, mas nem com o sensor parado nem com o sensor em movimento foi possível atingir tal fato.

Figura 59 – Diferença de medição máxima e mínima da medição para o sensor se movimentando a diferentes velocidades. Análise feita para o sinal filtrado também.



Fonte: Do autor.

Figura 60 – Análise da variância da medição para o sensor se movimentando a diferentes velocidades. Análise feita para o sinal filtrado também.



Fonte: Do autor.

O range de velocidades escolhidas foi definida de acordo com a folha de dados dos motores, fornecido pelo fabricante, e foi utilizado como critério de escolha, uma vez que não afeta o sensor, a suavidade e continuidade do movimento, que foi observada de forma empírica em testes de bancada. Dessa forma, foi escolhido que os motores iriam se movimentar a uma velocidade de 500 RPM, pois é uma velocidade mais suave para o sistema mecânico e que não afeta a medição do conversor analógico digital.

4.1.2 Teste de Resolução do Sistema de Aquisição

O fabricante do sensor laser $ODSL8$ garante uma resolução de 0,1 a 0,5 mm na medição, o que significa que qualquer variação de distância nessa unidade de grandeza deveria ser detectada pelo sensor. Entretanto, como existe uma limitação do conversor AD e pode haver ruídos na medição, pode ser que a resolução real do sistema de aquisição seja diferente.

Dessa forma, se viu necessário encontrar a resolução real do escâner, justamente para que os erros de medição pudessem ser considerados no momento de aplicar as técnicas de processamento de dados. Para isso, foi feito o teste da seguinte forma:

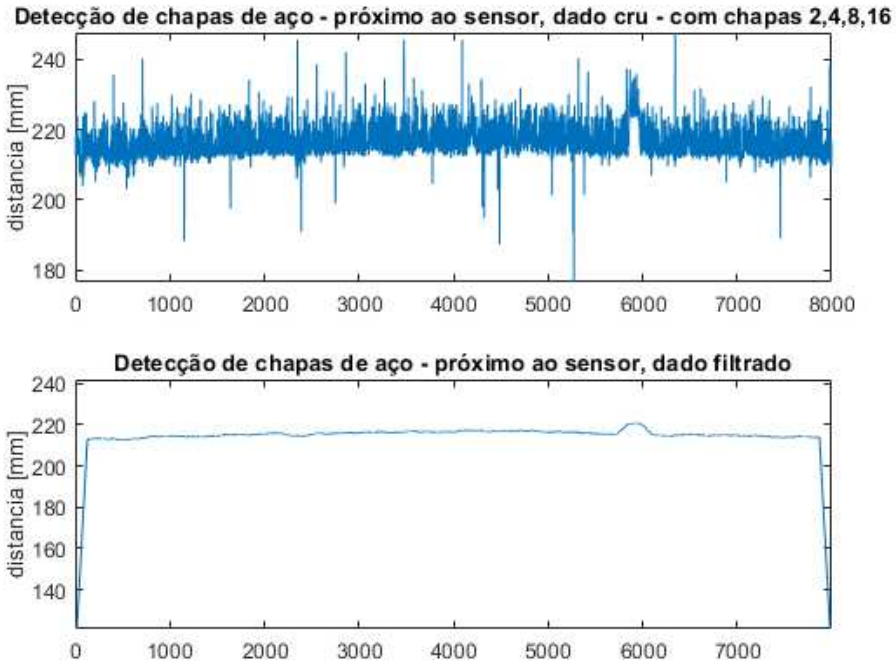
1. Foi montado uma base plana de apoio a aproximadamente 200 mm de distância do sensor laser;
2. Foram empilhadas chapas de aço de 0,5 mm de espessura da seguinte forma: uma pilha com duas chapas, uma com quatro, uma com oito e outra com dezesseis, conforme mostra a Figura 61;
3. Foi feito uma leitura de forma que o escâner passasse por cima do das chapas com o intuito de conseguir localizá-las.

Figura 61 – Teste da resolução do sensor com as chapas de aço.

Fonte: Do autor.

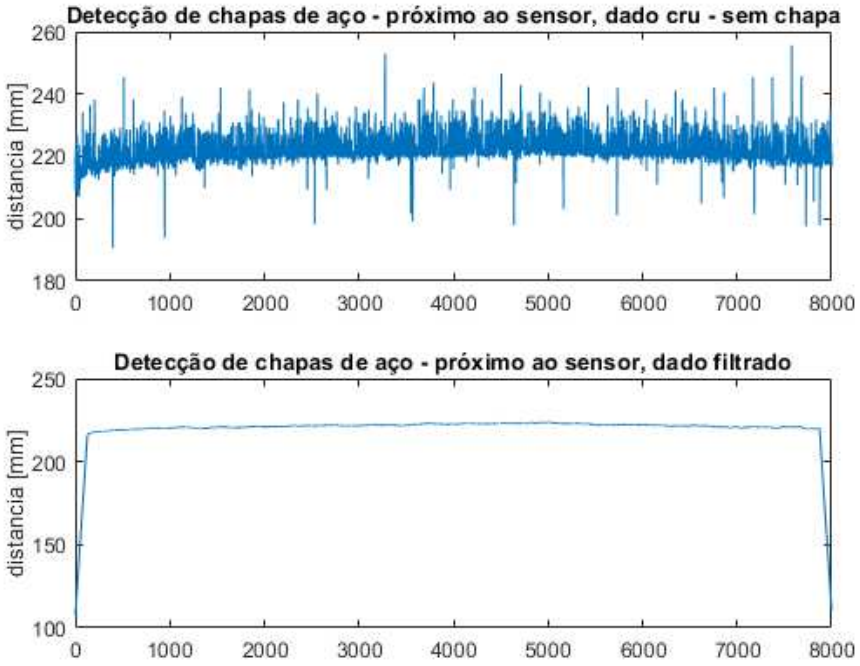
A Figura 62 mostra o resultado da varredura feita pelo escâner. A Figura 63 mostra a mesma varredura feita na superfície plana para efeito de comparação.

Figura 62 – Resultado da varredura das chapas de aço.



Fonte: Do autor.

Figura 63 – Varredura da superfície plana sem as chapas de aço.



Fonte: Do autor.

É possível observar na Figura 62 que o escâner conseguiu detectar bem suavemente a pilha de 8 chapas e de forma mais nítida a de 16 chapas. Considerando a espessura de 0,5 mm, o escâner na verdade tem uma resolução entre 4 e 8 mm. O sinal foi filtrado com uma filtro de média não causal de janela de tamanho de 128 amostras.

Os possíveis problemas que podem ter afetado a a resolução do sensor nesse teste foram: alinhamento do sensor, reflexão em superfície refletiva, valor medido muito próximo ao fundo de escala.

4.1.3 Aplicação Alternativa do Escâner

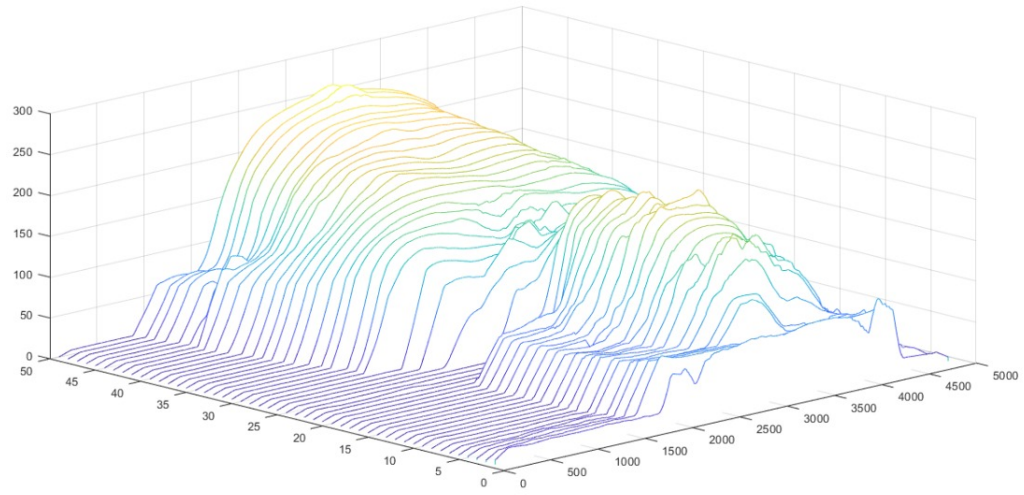
Apesar deste trabalho ter focado na utilização do escâner para levantar uma base de dados de peças metálicas e identificar falhas nessas peças, existem muitas outras funcionalidades para esse equipamento. Na seção 1.4 foi apresentado diversos trabalhos com diversas aplicações para o escâner 3D, inclusive com a utilização em seres humanos.

Uma vantagem do escâner desenvolvido aqui é a questão da área de varredura muito grande, permitindo que grandes objetos sejam escaneados, incluindo pessoas. Isso pode ser extremamente útil para aplicações biomédicas, por exemplo, onde há a necessidade do desenvolvimento de próteses e utensílios específicos para o corpo de cada pessoa.

Uma prova de que esse escâner pode ter essa funcionalidade foi um experimento realizado em que foi escaneado uma pessoa de corpo inteiro dentro do equipamento.

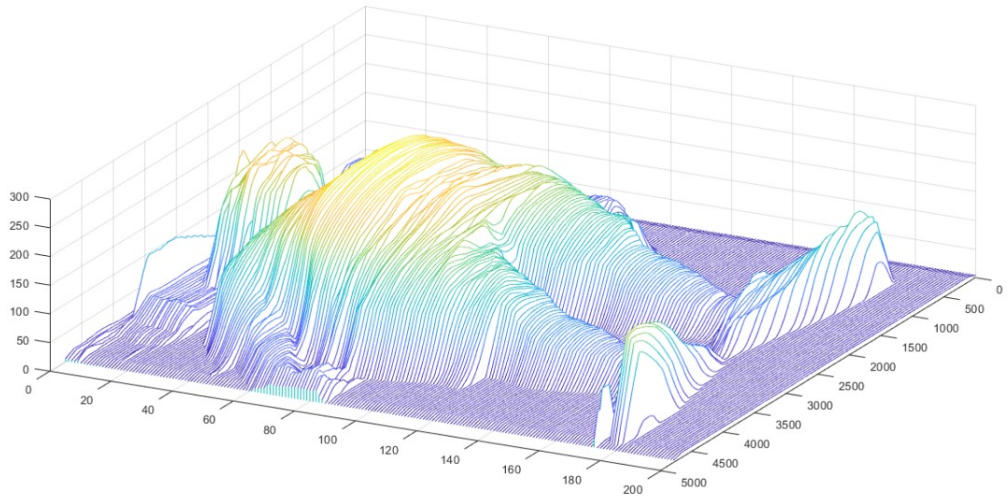
Utilizando-se das mesmas técnicas de aquisição e pré processamento de dados já apresentadas na seção de 3, os dados escaneados dessa pessoa foram plotados num *grid* 3D para visualização. O resultado pode ser observado nas Figuras 64 e 65.

Figura 64 – Digitalização de um ser humano pelo escâner 3D.



Fonte: Do autor.

Figura 65 – Digitalização de um ser humano pelo escâner 3D.



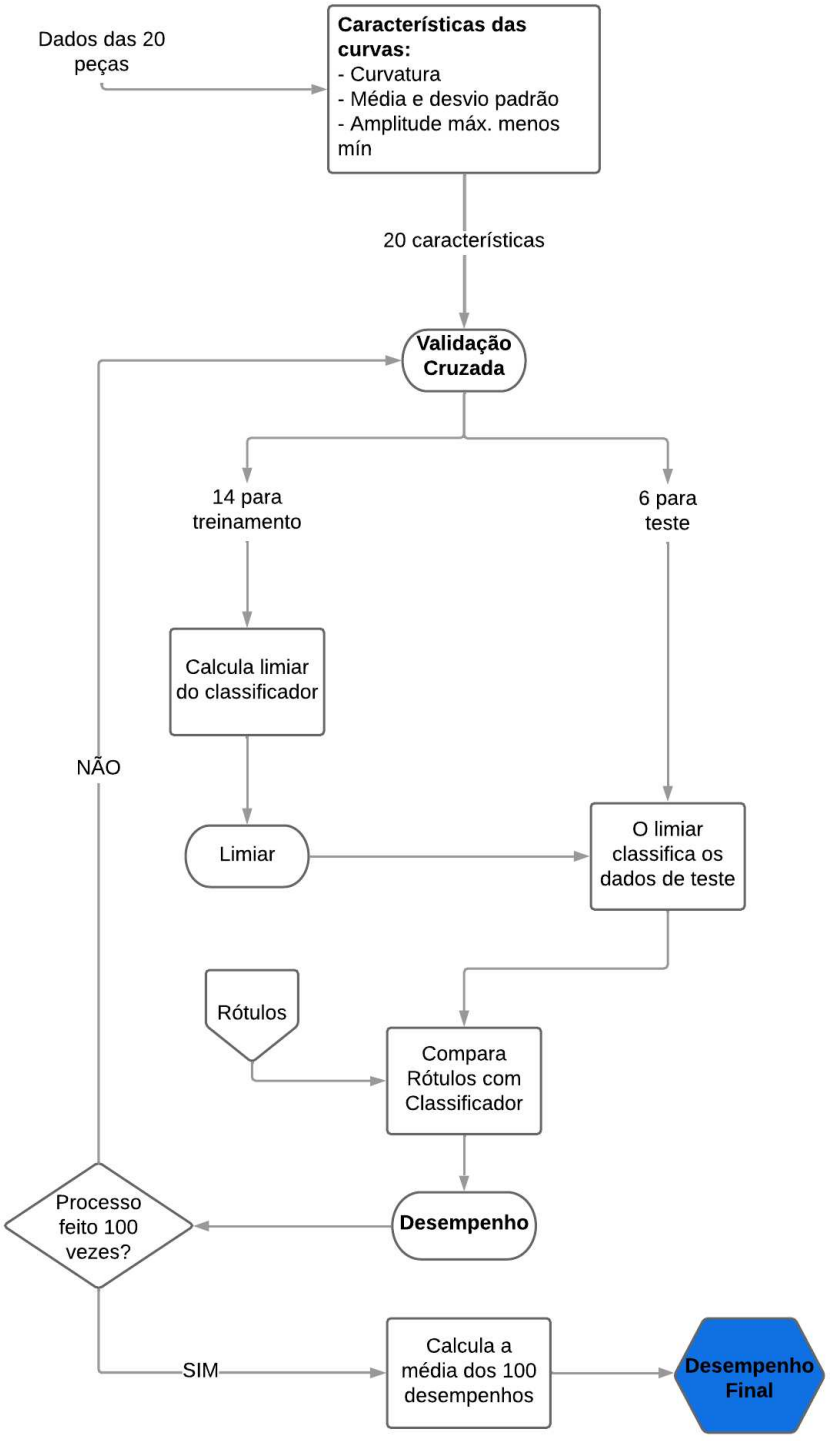
Fonte: Do autor.

É fácil observar que escâner consegue digitalizar a forma humana, mesmo que de uma maneira não tão exata. Algumas melhorias no projeto embarcado, como a utilização de um conversor AD com menos ruído de aquisição já poderia tornar essa representação ainda mais fidedigna.

4.2 Desempenho do Sistema de Classificação

Falando agora do sistema de identificação de falhas, e considerando os métodos de extração de característica mencionados e o classificador adotado, a Figura 66 mostra em forma de fluxograma o passo a passo do experimento realizado para validar o desempenho do sistema de classificação.

Figura 66 – Fluxograma que explica o experimento proposto para a identificação das falhas nas peças metálicas.



Fonte: Do autor.

O processo de validação cruzada foi repetido 100 vezes e o desempenho final é a média dos 100 desempenhos encontrados. Lembrando que para cada repetição, a validação cruzada seleciona os dados de treinamento e de teste de maneira aleatória.

O desempenho do classificador está baseado em três parâmetros: na taxa de loca-

lização de peças defeituosas, na taxa de falsos positivos e na taxa de acerto global.

A taxa de falsos positivos mostra o quanto de peças saudáveis o classificador classificou na classe de defeituosos. A taxa de localização indica a quantidade de peças defeituosas que foram classificadas corretamente de acordo com os rótulos da base de dados. Por fim, a taxa de acerto global é o complemento da taxa de erro total, ou seja, o quanto de peças defeituosas o classificador entendeu como saudáveis.

O processo mostrado na Figura 66 foi feito para os três tipos de característica propostos. Dessa forma, os resultados encontrados, utilizando o classificador linear baseado no limiar, estão mostrados na Tabela 4:

Tabela 4 – Resultados do sistema de classificação.

Tipo de Característica	TL	TFP	TA
Estatístico	94,24 %	10,09 %	91,66 %
Curvatura	89,69 %	10,60 %	89,54 %
Diferença de Amplitude	86,66 %	10,00 %	88,33 %

Fonte: Do Autor

É importante mencionar que TL é a taxa de localização de peças defeituosas, TFP é a taxa de falsos positivos e TA a taxa de acerto global.

As características estatísticas apresentaram um desempenho melhor, apesar de todas as taxas de acerto terem sido maiores que 85 %. O classificador linear utilizado é simples, baseado apenas em um limiar, mas mesmo assim conseguiu apresentar um bom desempenho.

Esse resultado mostra que os dados adquiridos pelo escâner são confiáveis e que podem ser utilizados para esse importante propósito, que é de identificar falhas estruturais em peças.

5 Considerações Finais

O desenvolvimento deste projeto envolveu a aplicação de várias áreas da engenharia elétrica, a consulta a muitas referências e bibliografias, além do aproveitamento de alguns projetos que serviram de base, como o desenvolvido em Lima (2020). Entretanto, as novas propostas colocadas aqui foram desafios que demandaram soluções, e soluções são compostas por esforços, testes, tentativa e erro, etc. Nesse sentido, será discutido neste capítulo os desafios por trás das soluções apresentadas ao longo do trabalho.

5.1 Discussão Sobre o Desenvolvimento do Equipamento

Primeiramente, em relação ao desenvolvimento do equipamento, o projeto mecânico foi aproveitado do projeto de (LIMA, 2020) e foi esse então o ponto de partida. Sendo assim, o projeto elétrico do escâner foi pensado em desenvolver um equipamento que se movimentasse de forma contínua, suave, e relativamente rápido. Foi mostrado ao longo da metodologia que o principal fator que contribuiu para garantir essa especificação, foi o desenvolvimento do sistema embarcado pensado para movimentar os motores de forma contínua, sem interrupção de movimento.

Entretanto, o atendimento a essa especificação do projeto fez nascer uma preocupação a respeito do funcionamento do sensor laser e sua aquisição com movimento; a ideia era conseguir realizar as leituras com o sensor se movimentando mas sem afetar a resolução e gerar ruído. Foi concluído por testes que a movimentação não afeta o desempenho do sensor, mas foi um desafio descobrir as origens dos ruídos de aquisição observados, mesmo com o sensor parado.

A solução para este problema foi a aplicação de filtro de média nos dados adquiridos, mas foi deixado claro que essa solução não é válida e melhor indicada para qualquer tipo de dado adquirido pelo escâner. Tem-se que lembrar que esta tratativa foi abordada para o problema específico, e que uma solução mais certa e definitiva pode ser alcançada com a utilização de um outro conversor analógico digital, como foi salientado no capítulo de metodologia.

Ademais, foi observado um funcionamento satisfatório do equipamento, sem ocorrência de falhas durante o período de desenvolvimento do trabalho. Dessa forma, pode-se concluir que o que foi pensado e proposto como ideia de projeto foi concebido e que o escâner como um todo funcionou como desejado.

5.2 Discussão Sobre o Desempenho do Sistema de Identificação de Falhas

Esta etapa foi tratada de maneira separada ao longo do trabalho pois o sistema de classificação, responsável por identificar as peças com falhas, foi desenvolvido em um sistema *offline*, fora do sistema embarcado do escâner. Em outras palavras, até a etapa de aquisição e armazenamento dos dados foi feito dentro do escâner, e as demais fora.

Foi observado no capítulo de resultados que o sistema desempenhou com mais de 85 % de taxa de acerto na localização de peças defeituosas, o que significa que o sistema de classificação foi bem dimensionado ao problema, apesar de contar com ferramentas computacionais mais simples, mas adequadas e assertivas.

Uma discussão que pode ser levantada é que no algoritmo de classificação, para o conjunto de peças disponíveis, mesmo com uma taxa de acerto acima de 80 % (vantagem), existem limitações que podem afetar a precisao, como velocidade de movimentação, resolução, ruído, interferências, etc. Talvez um melhor ajuste destas variáveis poderia ter impacto ainda mais positivamente o resultado.

Apesar disso, pode-se concluir que o sistema de classificação desempenhou bem por seguintes razões:

1. O desenvolvimento do sistema embarcado: as técnicas de programação de microcontrolador utilizadas permitiram que o movimento do escâner fosse contínuo, suave e relativamente ágil, otimizando o tempo de escaneamento sem perder precisão na leitura.

2. Algoritmo de classificação simples: apesar de ter sido desenvolvido fora do ambiente embarcado, o *design* do sistema de classificação foi feito utilizando técnicas computacionais mais simples, e mesmo assim atingiu níveis satisfatórios de desempenho.

Em suma, pode se observar grandes conquistas atingidas pelo trabalho com a proposta do projeto, seu desenvolvimento e resultados satisfatórios. Poderia acrescentar ainda que o trabalho foi além e ainda conseguiu mostrar outra aplicação para o escâner, mesmo que de forma mais sucinta, mas como ferramenta de inspiração para a continuidade e melhorias do equipamento, como será discutido a seguir.

5.3 Propostas para Trabalhos Futuros

Ficou claro até aqui que o equipamento desenvolvido atingiu seus objetivos e obteve êxito no que tange ao cumprimento da proposta. Entretanto, existem alguns pontos de melhoria no equipamento e na metodologia de desenvolvimento que podem ser estendidas para o surgimento de trabalhos futuros. Dentre essas propostas, vale destacar as seguintes:

- Utilizar os sensores de posição Encoders para conseguir ter um controle em malha fechada da movimentação dos motores de passo. Isso elimina a imprecisão da posição do suporte do sensor. A placa eletrônica PCI já tem entrada para três Encoders. Teria que adaptar o código de movimentação dos motores também. O grande desafio seria adaptar a estrutura mecânica para incorporar o sensor.

- Utilizar sensores de distância indutivos piezoelétricos para trabalhar como fim de curso na estrutura do escâner. É ideal para evitar contato frequente entre as peças e preservar a estrutura mecânica. As chaves fim de curso por contato, como foi utilizada nesse trabalho, ficam desgastadas com a quantidade de contato físico e frequentemente precisam ser trocadas. Nenhuma alteração no circuito da placa eletrônica precisaria ser feita.

- Utilizar um conversor analógico digital externo ao microcontrolador, com mais bits de resolução. Isso poderia ajudar a driblar alguns problemas no conversor do $ESP32$, como foi mostrado aqui. A placa eletrônica tem implementada em *hardware* uma adaptação para o módulo conversor AD $ADS1115$ de 16 bits. Teria que mudar o código para atribuição dos pinos do $ESP32$.

- Implementar o algoritmo de classificação embarcado no microcontrolador, já que se trata de um algoritmo mais simples do ponto de vista computacional e de processamento.

Referências

ABD-RAHEEM, A.; ALDEIRI, F.; ALYAMAN, M. Design of an automated 3d scanner. *2018 International Arab Conference on Information Technology (ACIT)*, IEEE, 2018. 4, 5

AL-SHOSHAN, A. I. Classification and separation of audio and music signals. In: QUEVEDO, E. (Ed.). *Multimedia Information Retrieval*. Rijeka: IntechOpen, 2021. cap. 6. Disponível em: <https://doi.org/10.5772/intechopen.94940>. 16

ALENCAR, M. D. *PROBABILIDADE E PROCESSOS ESTOCASTICOS*. [S.l.]: ERICA, 2009. ISBN 9788536502168. 21, 22

Andrushchak, N.; Neznaradko, Y.; Hnatyuk, V. Development and implementation of image processing technique for laser-based 3d scanner. In: *2017 14th International Conference The Experience of Designing and Application of CAD Systems in Microelectronics (CADSM)*. [S.l.: s.n.], 2017. p. 309–313. 4, 5

ANDRUSHCHAK, N.; VASYLYSHYN, ; CHORNENKYY, V. Comparative analysis of algorithms for projected laser line identification and recognition for 3d scanning devices. In: . [S.l.: s.n.], 2015. 10

ARETAKIS, N.; MATHIOUDAKIS, K.; STAMATIS, A. Identification of sensor faults on turbofan engines using pattern recognition techniques. *Control Engineering Practice*, v. 12, p. 827–836, 07 2004. 19

ARMESTO, J. et al. Terrestrial laser scanning used to determine the geometry of granite boulder for stability analysis purposes. *Geomorphology*, v. 106, p. 271–277, 05 2009. 5

ATMEL CORPORATION. *Atmel 8051 Microcontrollers Hardware Manual*. [S.l.], 2007. Disponível em: <https://ww1.microchip.com/downloads/en/DeviceDoc/doc4316.pdf>. Acesso em: 10 de novembro de 2021. 32

AXELSSON, P. M. Processing of laser scanner data—algorithms and applications. In: . [S.l.: s.n.], 1999. 10

BARROSO, L. C. *Cálculo Numérico (com aplicações)*. [S.l.]: Harbra, 1987. (2° edição). ISBN 9788529400891. 64

BEGA, E. A. et al. *Instrumentação Industrial*. [S.l.]: Editora Interciência, 2011. 12, 13

BO-LIN, S.; BI-FENG, S.; FEI, C. New sensor technologies in aircraft structural health monitoring. *2008 International Conference on Condition Monitoring and Diagnosis*, IEEE, 2007. 1

CAMPBELL, S. *BASICS OF THE SPI COMMUNICATION PROTOCOL*. 2017. Disponível em: <https://www.circuitbasics.com/basics-of-the-spi-communication-protocol>. Acesso em: 25 de janeiro de 2022. 34

CICIORA, W. et al. Chapter 4 - digital modulation. In: CICIORA, W. et al. (Ed.). *Modern Cable Television Technology (Second Edition)*. Second edition. San Francisco: Morgan Kaufmann, 2004, (The Morgan Kaufmann Series in Networking). p. 137–181. Disponível em: <https://www.sciencedirect.com/science/article/pii/B9781558608283500060>. 15

COLLINS, B.; SITAR, N. Application of high resolution 3d laser scanning to slope stability studies. In: . [S.l.: s.n.], 2004. 1, 5

DAFNA, E.; TARASIUK, A.; ZIGEL, Y. Automatic detection of whole night snoring events using non-contact microphone. *PloS one*, v. 8, p. e84139, 12 2013. 16

DANESHMAND, M. et al. 3d scanning: A comprehensive survey. *ArXiv*, abs/1801.08863, 2018. 4

DOEBELIN, E. *Measurement Systems: Application and Design*. McGraw-Hill, 1975. (International student edition). ISBN 9780070173361. Disponível em: <https://books.google.com.br/books?id=LBY7AQAAIAAJ>. 12

DUDA, R. O.; HART, P. E. Pattern recognition and scene analysis. In: . [S.l.: s.n.], 1973. 25, 26

ESPRESSIF SYSTEMS. *ESP32 Series datasheet*. [S.l.], 2016. Disponível em: <https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf>. Acesso em: 5 janeiro de 2021. 45

ESPRESSIF SYSTEMS. *ESP 32 Analog to Digital Converter (ADC)*. [S.l.], 2018. Disponível em: <https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/peripherals/adc.html>. Acesso em: 5 janeiro de 2021. 55, 56

FARO. *FARO SCANARM - The Ideal Contact/Non-Contact Portable Measurement System*. 2016. Disponível em: <https://www.faro.com/en-gb/products/3d-manufacturing/faro-scanarm/>. Acesso em: 10 de outubro de 2021. 9

GAZZIRO, M. A. *Projeto e Construção de um Scanner Antropométrico Baseado no Método de Triangulação a Laser*. Dissertação (Mestrado) — Centro de Ciências Exatas e de Tecnologia da Universidade Federal de São Carlos, São Carlos, São Paulo, Brasil, 8 2005. 6, 7

GHAZALI, R. et al. Evaluating the relationship between scanning resolution of laser scanner with the accuracy of the 3d model constructed. *IEEE International Conference on Control System, Computing and Engineering*, IEEE, 2011. 4, 12

GUTKIN, R. et al. On acoustic emission for failure investigation in cfrp: Pattern recognition and peak frequency analyses. *Mechanical Systems and Signal Processing*, v. 25, p. 1393–1407, 05 2011. 19

HE, K. et al. Enhancing the monitoring of 3d scanned manufactured parts through projections and spatiotemporal control charts. *Journal of Intelligent Manufacturing*, v. 28, n. 4, p. 899–911, Apr 2017. ISSN 1572-8145. Disponível em: <https://doi.org/10.1007/s10845-014-1025-1>. 2, 5

HEATH, J. *PWM: Pulse Width Modulation: What is it and how does it work?* 2017. Disponível em: <https://www.analogictips.com/pulse-width-modulation-pwm/>. Acesso em: 5 de janeiro de 2022. 30

JAIN, A. K. Data clustering: 50 years beyond k-means. *Pattern Recogn. Lett.*, Elsevier Science Inc., New York, NY, USA, v. 31, n. 8, p. 651–666, jun. 2010. ISSN 0167-8655. Disponível em: <http://dx.doi.org/10.1016/j.patrec.2009.09.011>. 25

KOHAVI, R. A study of cross-validation and bootstrap for accuracy estimation and model selection. In: *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1995. (IJCAI'95), p. 1137–1143. ISBN 1558603638. 27

KOHAVI, R.; PROVOST, F. Glossary of terms. *Machine Learning*, v. 30, n. 2-3, p. 271–274, February 1998. Disponível em: <http://www.springerlink.com/content/x105525142v51t20/>. 27

Lavelle, J. P.; Schuet, S. R.; Schuet, D. J. High speed 3d scanner with real-time 3d processing. In: *2004 IEEE International Workshop on Imaging Systems and Techniques (IST) (IEEE Cat. No.04EX896)*. [S.l.: s.n.], 2004. p. 13–17. 4, 5

LEE, E. A.; SESHIA, S. A. Introduction to embedded systems - a cyber-physical systems approach. In: . [S.l.: s.n.], 2013. 13, 15, 28, 29, 32

LEUZE ELETRONICS. *ODSL8 - Optical laser distance sensors*. [S.l.], 2015. Disponível em: <https://www.leuze.com/pt/brasil/produtos/sensores_de_medi_o/sensores_de_dist_ncias_pticos/odsl_16/selector.php>. 51

LIMA, G. H. de S. *Desenvolvimento de um sistema de leitura e análise estrutural utilizando técnicas espectrais*. 2020. Monografia (Bacharel em Engenharia Elétrica), UFOP (Universidade Federal de Ouro Preto), João Monlevade, Brazil. 39, 40, 48, 78

MARINDRA, A. M. J.; TIAN, G. Y. Chipless rfid sensor tag for metal crack detection and characterization. *IEEE TRANSACTIONS ON MICROWAVE THEORY AND TECHNIQUES*, IEEE, v. 66, n. 5, p. 2452–2462, 2008. 1

MARSHALL, J. *Curvature and Normal Vectors of a Curve*. 2021. [Online; accessed 2022-03-15]. 22, 23

MATHWORKS. *Designing the Filter*. 2019. Disponível em: <https://www.mathworks.com/help/signal/gs/designing-the-filter.html>. 17

MATHYS, A.; BRECKO, J.; SEMAL, P. Comparing 3d digitizing technologies: What are the differences? In: . [S.l.: s.n.], 2013. 9

MOLDER, A. et al. Laser line detection with sub-pixel accuracy. *Elektronika ir Elektrotechnika*, v. 20, 05 2014. 4

MOSTAFA, A.-B.; EBRAHIM, M. 3d laser scanners' techniques overview. *International Journal of Science and Research (IJSR)*, v. 4, p. 5–611, 10 2015. 4

NEXTENGINE. *About Company*. 2015. Disponível em: <http://www.nextengine.com/company/about>. Acesso em: 4 de outubro de 2021. 8

OPPENHEIM, A.; WILLSKY, A.; YOUNG, I. *Signals and systems*. [S.l.]: Prentice-Hall, 1983. (Prentice-Hall signal processing series). 14

OPPENHEIM, A. V. *Discrete-Time Signal Processing*. [S.l.]: Pearson Education, 2011. 14, 15, 17

PAWAR, S. et al. Review paper on design of 3d scanner. *International Conference on Innovative Mechanisms for Industry Applications*, IEEE, 2017. 4

RAFIQUZZAMAN, M. *Fundamentals of Digital Logic and Microcomputer Design*. 3rd. ed. [S.l.]: Rafi Systems, Incorporated, 2000. ISBN 0966498038. 30

RAMOS, A. C. da S. *Estudo de Técnicas de Reconhecimento de Padrões*. 2014. Monografia (Bacharel em Sistemas de Informação), UFOP (Universidade Federal de Ouro Preto), João Monlevade, Brazil. 19, 20, 24, 26

RUGGIERO, M.; LOPES, V. da R. *Cálculo numérico: aspectos teóricos e computacionais*. Pearson Makron Books, 1996. ISBN 9788534602044. Disponível em: <https://books.google.com.br/books?id=kuRDAAAACAAJ>. 64

SEN, P. *Principles of Electric Machines and Power Electronics, 3rd Edition: Third Edition*. [S.l.]: Wiley Global Education, 2013. ISBN 9781118804155. 46

STALLINGS, W. *Computer Organization and Architecture*. 6th. ed. [S.l.]: Prentice Hall Professional Technical Reference, 2002. ISBN 0130351199. 28

STEFANI, M. A. *Medidores de distância por Trinagulação a Laser*. Tese (Doutorado) — Instituto de Física da Universidade de São Paulo, São Carlos, São Paulo, Brasil, 1995. 13

Sukvichai, K. et al. Simple 3d splint reconstruction using a low cost smart phone line laser 3d scanner. In: *2018 International Conference on Electronics, Information, and Communication (ICEIC)*. [S.l.: s.n.], 2018. p. 1–4. 8

TEXAS INSTRUMENTS. *DRV8825 Stepper Motor Controller IC*. [S.l.], 2010. Disponível em: <https://www.ti.com/lit/ds/symlink/drv8825.pdf?HQS=TI-null-null-alldatasheets-df-pf-SEP-wwe>. 43, 46, 48

TEXAS INSTRUMENTS. *LM2596 SIMPLE SWITCHER Power Converter*. [S.l.], 2013. Disponível em: <https://pdf1.alldatasheet.com/datasheet-pdf/view/524001/TI/LM2596.html>. Acesso em: 7 de janeiro de 2021. 43

THEODORIDIS, S.; KOUTROUMBAS, K. *Pattern Recognition*. [S.l.]: Elsevier, 2009. 11, 19, 20, 21, 25

UKIDA, H. et al. 3d object reconstruction using image scanner with multiple light sources. 07 2010. 4

YAO, A. Applications of 3d scanning and reverse engineering techniques for quality control of quick response products. *The International Journal of Advanced Manufacturing Technology*, v. 26, n. 11, p. 1284–1288, Nov 2005. ISSN 1433-3015. Disponível em: <https://doi.org/10.1007/s00170-004-2116-5>. 5

Yunardi, R. T.; Imandiri, A. Design of the 3d surface scanning system for human wrist contour using laser line imaging. In: *2018 5th International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE)*. [S.l.: s.n.], 2018. p. 1–5. 4, 6

Apêndice A

Código do Sistema Embarcado

```
1  //-------------------------------------------------------------------------------
2  // Inicializa o das Bibliotecas
3
4  #include <Wire.h>
5  #include <Adafruit_ADS1015.h>
6  #include <ESPmDNS.h>
7  #include "FS.h"
8  #include "Network.h"
9  #include "Sys_Variables.h"
10 #include "CSS.h"
11 #include <SD.h>
12 #include <SPI.h>
13 #include <WiFi.h>
14 #include <WebServer.h>
15 #include <string.h>
16 #include <driver/adc.h>
17 #include "esp_adc_cal.h"
18
19
20
21 //-------------------------------------------------------------------------------
22 // Inicializa o das Vari veis Globais
23
24 // Pinos do motor
25 #define STEP 13
26 #define DIR 17
27 #define slp_MT 25
28 #define slp_ML 33
29
30 // Pinos do fc
31 #define fc1 4
32 #define fc2 15
33
34 // Pinos de controle do driver DRV8825
35 #define MODE0 26
36 #define MODE1 27
37 #define MODE2 14
38
39 // Pino do AD do ESP32
40 #define ads 36
41
```

```
42 // Dados dos Arquivos
43 File myFile;
44 String filename = "/datalog_0000001.txt";
45 String dataString = "";
46 int n_pts = 4800;
47 uint32_t reading[4800];
48 double reading_float = 0;
49 int aux = 0;
50
51 // Declara  o dos Timers
52 hw_timer_t * timer = NULL;
53 hw_timer_t * timer1 = NULL;
54
55 // Vari vel de Calibra  o do ADC
56 esp_adc_cal_characteristics_t adc_cal;
57
58 // Vari veis de Tempo
59 int T_timer = 2500;
60 int T_timer1 = 1000;
61
62 // Vari veis de Controle de Movimenta  o
63 String yfi;
64 String pl;
65 int x_step = 960;
66 int x_middle_step = 435;
67 int x_zero_step = 525;
68 int y_step;
69 int y_total_step;
70 int yf;
71 int y_zero_step;
72
73 // Vari veis Contadoras
74 int step_count = 0;
75 int adc_count = 0;
76 int delay_count = 0;
77 int total_delay = 150;
78 int idx = 0;
79
80 // Flags de Controle
81 int set_timer0 = 0;
82 int delay_set = 0;
83 int delay_timer = 0;
84 int control_flag = 0;
85 int finished = 0;
86 int finished_y = 0;
87 int adc = 0;
88 int set_adc = 0;
```

```
89  int fc_on = 0;
90  int fc_on2 = 0;
91  int finished_x = 0;
92  int working = 1;
93
94  // Estado de Chave
95  volatile byte state = LOW;
96
97  // SSID e Senha do Wifi
98  const char* ssid = "ESP32-ProcSiMoS-Manipulador";
99  const char* password = "12345678";
100 //IPAddress local_ip(192,168,1,1);
101 //IPAddress gateway(192,168,1,1);
102 //IPAddress subnet(255,255,255,0);
103
104 // Declarando o Server
105 WebServer server(80);
106
107 // Declarando a P gina HTML
108 const char index_html[] = R"rawliteral(
109 <!DOCTYPE HTML><html><head>
110   <title>ESP Input Form Manipulator </title>
111   <meta name="viewport" content="width=device-width, initial-scale=1">
112   </head><body>
113 <form action="/get">
114   <fieldset>
115     <legend><h2>ESP32 Web Server - ProcSiMoS Manipulador</h2></legend>
116     Y fim [mm]: <br><input type="number" min="0.00" max="2150" ...
          step="2" name="yf" required><br><br>
117     Passo em Y [mm]: <br><input type="number" min="2" max="2000" ...
          step="2" name="pl" required><br>
118     <legend><h4>Obs: O passo do manipulador em Y deve ser multiplo ...
          de 2 mm;<br>X corresponde ao deslocamento transversal e Y ao ...
          deslocamento Longitudinal.</h4></legend>
119     <input type="submit" value="Enviar">
120   </fieldset>
121 </form>
122 </body></html>)rawliteral";
123
124
125
126 //-------------------------------------------------------------------------
127 // Inicializa o dos Timers
128
129 // Timer 0 - Controle dos Motores
130 void IRAM_ATTR onTimer(){
131   if(delay_count==total_delay){
```

```
132     delay_set = 1;
133   }
134   else{
135     delay_count++;
136   }
137   if(set_timer0==1){
138     state = !state;
139     digitalWrite(STEP,state);
140     step_count++;
141   }
142 }
143
144 // Timer 1 - Controle da Aquisi  o do ADC
145 void IRAM_ATTR onTimer1(){
146   adc = 1;
147   if(adc==1 && set_adc==1 && aux<n_pts){
148     reading[aux] = adc1_get_raw(ADC1_CHANNEL_0);
149     aux++;
150     adc = 0;
151   }
152 }
153
154
155
156 //--------------------------------------------------------------------------------
157 // Inicializa  o do Setup
158
159 void setup(){
160   Serial.begin(115200);
161   // Inicializa  o da Conex o
162   while(!Serial){
163     ;
164   }
165   WiFi.softAP(ssid,password,6);
166   server.on("/",handle_OnConnect);
167   server.on("/get",handle_get);
168   server.on("/downloadsd",handle_downloadsd);
169   server.on("/leitura",handle_leitura);
170   server.on("/cancel",handle_cancel);
171   server.on("/download",File_Download);
172   server.on("/dir",SD_dir);
173   server.on("/concluido",leitura_finalizada);
174   server.onNotFound(handle_NotFound);
175   server.begin();
176   Serial.println("HTTP server started");
177   // Inicializa  o do M dulo do Cart o SD
178   Serial.println(MISO);
```

```
179   pinMode(19,INPUT_PULLUP);
180   Serial.print(F("Initializing SD card..."));
181   if(!SD.begin(SD_CS_pin)){ // see if the card is present and can be ...
          initialised. Wemos SD-Card CS uses D8
182     Serial.println(F("Card failed or not present, no SD Card data ...
            logging possible..."));
183     SD_present = false;
184   }
185   else{
186     Serial.println(F("Card initialised... file access enabled..."));
187     SD_present = true;
188   }
189   // Configuira o do AD
190   adc1_config_width(ADC_WIDTH_BIT_12);
191   adc1_config_channel_atten(ADC1_CHANNEL_0,ADC_ATTEN_DB_11);
192   esp_adc_cal_value_t val_type = ...
          esp_adc_cal_characterize(ADC_UNIT_1,ADC_ATTEN_DB_11,ADC_WIDTH_BIT_12,1100,&ad
193   // Configura o dos Pinos
194   pinMode(STEP,OUTPUT);
195   pinMode(DIR,OUTPUT);
196   pinMode(slp_MT,OUTPUT);
197   pinMode(slp_ML,OUTPUT);
198   pinMode(MODE0,OUTPUT);
199   pinMode(MODE1,OUTPUT);
200   pinMode(MODE2,OUTPUT);
201   pinMode(fc1,INPUT_PULLDOWN);
202   pinMode(fc2,INPUT_PULLDOWN);
203   attachInterrupt(digitalPinToInterrupt(fc1),end_switch,FALLING);
204   attachInterrupt(digitalPinToInterrupt(fc2),end_switch2,FALLING);
205   digitalWrite(MODE0,LOW);
206   digitalWrite(MODE1,LOW);
207   digitalWrite(MODE2,LOW);
208   //inicializa o do Timer 0
209   timer = timerBegin(0,80,true);
210   timerAttachInterrupt(timer,&onTimer,true);
211   timerAlarmWrite(timer,T_timer,true);
212   timerAlarmEnable(timer);
213   //inicializa o do Timer 1
214   timer1 = timerBegin(1,80,true);
215   timerAttachInterrupt(timer1,&onTimer1,true);
216   timerAlarmWrite(timer1,T_timer1,true);
217   timerAlarmEnable(timer1);
218 }
219
220
221
222 //-------------------------------------------------------------------------------
```

```
223 // Inicializa o do Loop
224
225 void loop(){
226   server.handleClient();
227   if (!working){
228   // Salva no Cart o de Mem ria
229   if(set_adc==0 && finished_x==1){
230     aux = 0;
231     finished_x = 0;
232     set_adc = 0;
233     for(idx=0;idx<n_pts;idx++){
234       reading_float = esp_adc_cal_raw_to_voltage(reading[idx],&adc_cal);
235       reading_float = reading_float*-0.29010989+600;
236       dataString += String(reading_float);
237       dataString += ",";
238       reading_float = 0;
239     }
240     myFile = SD.open(filename.c_str(),FILE_APPEND);
241     myFile.println(dataString);
242     myFile.println(";");
243     myFile.close();
244     dataString = "";
245   }
246   if(set_adc==0 && finished_x==0 && y_total_step==0){
247     finished_x = 2;
248   }
249   // Volta para o Zero no Fim da Leitura
250   if(y_total_step==0 && finished_y==0 && delay_set==1){
251     set_timer0 = 1;
252     finished_y = 1;
253     step_count = 0;
254     movement_generator(4);
255   }
256   if(fc_on2==1 && finished_y==1){
257     state = LOW;
258     step_count = 0;
259     finished_y = 2;
260     set_timer0 = 0;
261     delay_count = 0;
262     delay_set = 0;
263     fc_on = 0;
264     digitalWrite(slp_MT,LOW);
265     digitalWrite(slp_ML,LOW);
266   }
267   if(y_total_step==0 && finished_y==2 && delay_set==1){
268     delay_set = 1;
269     set_timer0 = 1;
```

```
270       finished_y = 3;
271       step_count = 0;
272       movement_generator(3);
273       fc_on = 0;
274     }
275     if(fc_on==1 && finished_y==3){
276       state = LOW;
277       step_count = 0;
278       finished_y = 4;
279       set_timer0 = 0;
280       delay_count = 0;
281       delay_set = 0;
282       fc_on = 0;
283       digitalWrite(slp_MT,LOW);
284       digitalWrite(slp_ML,LOW);
285     }
286     if(finished_y==4){
287       digitalWrite(slp_MT,LOW);
288       digitalWrite(slp_ML,LOW);
289       working = 1;
290       fc_on = 0;
291     }
292     // Movimenta os Motores
293     // Conta os Passos em Y
294     if(y_total_step>0){
295       //  L   o X
296       if(delay_set==1 && finished==0 && control_flag==0){
297         set_timer0 = 1;
298         set_adc = 1;
299         movement_generator(control_flag);
300         control_flag = 1;
301       }
302       if(step_count==x_step*2 && finished==0){
303         finished_x = 1;
304         set_adc = 0;
305         state = LOW;
306         step_count = 0;
307         finished = 1;
308         set_timer0 = 0;
309         delay_count = 0;
310         delay_set = 0;
311         digitalWrite(slp_MT,LOW);
312       }
313       // Volta para o Meio
314       if(delay_set==1 && finished==1 && control_flag==1){
315         set_timer0 = 1;
316         movement_generator(control_flag);
```

```
317         control_flag = 2;
318       }
319       if(step_count==x_middle_step*2 && finished==1){
320         state = LOW;
321         step_count = 0;
322         finished = 2;
323         set_timer0 = 0;
324         delay_count = 0;
325         delay_set = 0;
326         digitalWrite(slp_MT,LOW);
327       }
328       // Movimenta em Y
329       if(delay_set==1 && finished==2 && control_flag==2){
330         set_timer0 = 1;
331         movement_generator(control_flag);
332         control_flag = 3;
333       }
334       if(step_count==y_step*2 && finished==2){
335         state = LOW;
336         step_count = 0;
337         finished = 3;
338         set_timer0 = 0;
339         delay_count = 0;
340         delay_set = 0;
341         fc_on = 0;
342         digitalWrite(slp_ML,LOW);
343       }
344       // Zera o X
345       if(delay_set==1 && finished==3 && control_flag==3){
346         set_timer0 = 1;
347         movement_generator(control_flag);
348         control_flag = 0;
349         fc_on = 0;
350       }
351       if(finished==3 && fc_on==1){
352         state = LOW;
353         step_count = 0;
354         finished = 0;
355         set_timer0 = 0;
356         delay_count = 0;
357         delay_set = 0;
358         aux = 0;
359         fc_on = 0;
360         digitalWrite(slp_MT,LOW);
361       }
362     }
363     }
```

```
364 }
365
366
367
368 //-------------------------------------------------------------------------------
369 // Fun  es de Controle de Movimenta  o
370
371 void movement_generator(int flag){
372   // l   o X
373   if(flag==0){
374     digitalWrite(slp_MT,HIGH);
375     digitalWrite(slp_ML,LOW);
376     digitalWrite(DIR,HIGH);
377     digitalWrite(MODE0,LOW);
378     digitalWrite(MODE1,LOW);
379     digitalWrite(MODE2,LOW);
380   }
381   // Volta para o Meio
382   if(flag==1){
383     digitalWrite(slp_MT,HIGH);
384     digitalWrite(slp_ML,LOW);
385     digitalWrite(DIR,LOW);
386     digitalWrite(MODE0,LOW);
387     digitalWrite(MODE1,LOW);
388     digitalWrite(MODE2,LOW);
389   }
390   // Movimenta em Y
391   if(flag==2){
392     digitalWrite(slp_MT,LOW);
393     digitalWrite(slp_ML,HIGH);
394     digitalWrite(DIR,HIGH);
395     digitalWrite(MODE0,LOW);
396     digitalWrite(MODE1,LOW);
397     digitalWrite(MODE2,LOW);
398     y_total_step--;
399   }
400   // Zera o X
401   if(flag==3){
402     digitalWrite(slp_MT,HIGH);
403     digitalWrite(slp_ML,LOW);
404     digitalWrite(DIR,LOW);
405     digitalWrite(MODE0,LOW);
406     digitalWrite(MODE1,LOW);
407     digitalWrite(MODE2,LOW);
408   }
409   // Zera o Y
410   if(flag==4){
```

```
411     digitalWrite(slp_MT,LOW);
412     digitalWrite(slp_ML,HIGH);
413     digitalWrite(DIR,LOW);
414     digitalWrite(MODE0,LOW);
415     digitalWrite(MODE1,LOW);
416     digitalWrite(MODE2,LOW);
417   }
418 }
419
420 void end_switch(){
421   state = LOW;
422   fc_on = 1;
423 }
424
425 void end_switch2(){
426   state = LOW;
427   fc_on2 = 1;
428 }
429
430
431
432 //--------------------------------------------------------------------------------
433 // Fun  es de Conex o Wi-Fi e HTML
434
435 void handle_get(){
436   // Pegando os Valores do Usu rio do Web Server
437   yfi = server.arg("yf");
438   pl = server.arg("pl");
439   yf = yfi.toInt();
440   y_step = pl.toInt();
441   y_total_step = yf/y_step;
442   y_step = y_step*3*0.5;
443   y_zero_step = yf*3*0.5;
444   // Zerando as Vari veis de Flag
445   set_timer0 = 0;
446   delay_set = 0;
447   delay_timer = 0;
448   control_flag = 0;
449   finished = 0;
450   finished_y = 0;
451   adc = 0;
452   set_adc = 0;
453   fc_on = 0;
454   fc_on2 = 0;
455   finished_x = 0;
456   // D  o Enable para Come ar o Processo de Escanear
457   working = 0;
```

```
458   // Configura  o do Nome do Arquivo Novo
459   int contname = 1;
460   int test_name = 1;
461   int tamanho;
462   while(test_name){
463     tamanho = filename.length()-8;
464     filename.remove(8,tamanho);
465     filename.concat('_');
466     filename.concat(String(contname));
467     filename.concat(".txt");
468     Serial.println(filename);
469     if(SD.exists(filename.c_str())){
470       contname++;
471     }
472     else{
473       myFile = SD.open(filename.c_str(),FILE_WRITE);
474       test_name = 0;
475       myFile.close();
476     }
477   }
478   server.send(200, "text/html",Send_HTML_working());
479 }
480
481 void handle_OnConnect(){
482   if(working)
483     server.send(200,"text/html",SendHTML());
484   else
485     server.send(200,"text/html",SendHTML_leiturafinalizada());
486 }
487
488 void handle_leitura(){
489   if(working)
490     server.send(200,"text/html",index_html);
491   else
492     server.send(200,"text/html",SendHTML_leiturafinalizada());
493 }
494
495 void handle_NotFound(){
496   if(working)
497     server.send(404,"text/plain", "Not found");
498   else
499     server.send(200,"text/html",SendHTML_leiturafinalizada());
500 }
501
502 void handle_cancel(){
503   if (!working){
504     working = 1;
```

```
505       SD.remove(filename.c_str());
506       server.send(200,"text/html",SendHTML());
507     }
508     else{
509       server.send(200,"text/html",SendHTML());
510     }
511   }
512
513   void leitura_finalizada(){
514     server.send(200,"text/html",SendHTML_leiturafinalizada());
515   }
516
517   String SendHTML(){
518     String ptr = "<!DOCTYPE html> <html>\n";
519     ptr += "<head><meta name=\"viewport\" ...
          content=\"width=device-width, initial-scale=1.0, ...
          user-scalable=no\">\n";
520     ptr += "<title>ProcSiMoS - Inicio</title>\n";
521     ptr += "<style>html { font-family: Helvetica; display: ...
          inline-block; margin: 0px auto; text-align: center;}\n";
522     ptr += "body{margin-top: 50px;} h1 {color: #444444;margin: 50px ...
          auto 30px;} h3 {color: #444444;margin-bottom: 50px;}\n";
523     ptr += ".button {display: block;width: 100px;background-color: ...
          #3498db;border: none;color: white;padding: 13px ...
          30px;text-decoration: none;font-size: 25px;margin: 0px auto ...
          35px;cursor: pointer;border-radius: 4px;}\n";
524     ptr += ".button-on {background-color: #3498db;}\n";
525     ptr += ".button-on:active {background-color: #2980b9;}\n";
526     ptr += ".button-off {background-color: #34495e;}\n";
527     ptr += ".button-off:active {background-color: #2c3e50;}\n";
528     ptr += "p {font-size: 14px;color: #888;margin-bottom: 10px;}\n";
529     ptr += "</style>\n";
530     ptr += "</head>\n";
531     ptr += "<body>\n";
532     ptr += "<h1>ESP32 Web Server - ProcSiMoS Manipulador</h1>\n";
533     ptr += "<h3>Escolha a opcao desejada</h3>\n";
534     ptr += "<a class=\"button button-on\" href=\"/leitura\">Leitura ...
          3D</a>\n";
535     ptr += "<a class=\"button button-on\" ...
          href=\"/downloadsd\">Download </a>\n";
536     ptr += "</body>\n";
537     ptr += "</html>\n";
538     return ptr;
539   }
540
541   String Send_HTML_working(){
542     String ptr = "<!DOCTYPE html> <html>\n";
```

```
543   ptr += "<head><meta name=\"viewport\" ...
          content=\"width=device-width, initial-scale=1.0, ...
          user-scalable=no\">\n";
544   ptr += "<meta http-equiv=\"Content-Type\" ...
          content=\"text/html;charset=utf-8\">\n";
545   ptr += "<title>ProcSiMoS - Aguarde</title>\n";
546   ptr += "<style>html { font-family: Helvetica; display: ...
          inline-block; margin: 0px auto; text-align: center;}\n";
547   ptr += "body{margin-top: 50px;} h1 {color: #444444;margin: 50px ...
          auto 30px;} h3 {color: #444444;margin-bottom: 50px;}\n";
548   ptr += ".button {display: block;width: 120px;background-color: ...
          #3498db;border: none;color: white;padding: 13px ...
          30px;text-decoration: none;font-size: 25px;margin: 0px auto ...
          35px;cursor: pointer;border-radius: 4px;}\n";
549   ptr += ".button-on {background-color: #3498db;}\n";
550   ptr += ".button-on:active {background-color: #2980b9;}\n";
551   ptr += ".button-off {background-color: #34495e;}\n";
552   ptr += ".button-off:active {background-color: #2c3e50;}\n";
553   ptr += "p {font-size: 14px;color: #888;margin-bottom: 10px;}\n";
554   ptr += "</style>\n";
555   ptr += "</head>\n";
556   ptr += "<body>\n";
557   ptr += "<h1>ESP32 Web Server - ProcSiMoS Manipulador</h1>\n";
558   ptr += "<h3>O Manipulador est  efetuando a leitura.</h3>\n";
559   ptr += "<a class=\"button button-on:active\" ...
          href=\"/concluido\">Continuar</a>\n";
560   ptr += "</body>\n";
561   ptr += "</html>\n";
562   return ptr;
563 }
564
565
566
567 //--------------------------------------------------------------------------------
568 // Fun  es do Servidor de Download SD Card
569
570 void handle_downloadsd(){
571   if (working){
572     SendHTML_Header();
573     webpage += F("<a href='/download'><button>Download</button></a>");
574     webpage += F("<a href='/dir'><button>Directory</button></a>");
575     append_page_footer();
576     SendHTML_Content();
577     SendHTML_Stop();// Stop is needed because no content length was sent
578   }
579   else{
580     server.send(200,"text/html",Send_HTML_working());
```

```
581   }
582 }
583
584 void File_Download(){// This gets called twice, the first pass ...
      selects the input, the second pass then processes the command ...
      line arguments
585   if (working){
586     if (server.args()>0){// Arguments were received
587       if (server.hasArg("download")) SD_file_download(server.arg(0));
588     }
589     else SelectInput("File Download","Enter filename to ...
          download","download","download");
590   }
591   else {
592     server.send(200,"text/html",Send_HTML_working());
593   }
594 }
595
596 void SD_file_download(String filename){
597   if (SD_present){
598     File download = SD.open("/" + filename);
599     if (download){
600       server.sendHeader("Content-Type", "text/text");
601       server.sendHeader("Content-Disposition","attachment ; ...
            filename=" + filename);
602       server.sendHeader("Connection","close");
603       server.streamFile(download,"application/octet-stream");
604       download.close();
605     } else ReportFileNotPresent("download");
606   } else ReportSDNotPresent();
607 }
608
609 void SendHTML_Header(){
610   server.sendHeader("Cache-Control", "no-cache, no-store, ...
        must-revalidate");
611   server.sendHeader("Pragma", "no-cache");
612   server.sendHeader("Expires", "-1");
613   server.setContentLength(CONTENT_LENGTH_UNKNOWN);
614   server.send(200,"text/html", "");// Empty content inhibits ...
        Content-length header so we have to close the socket ourselves.
615   append_page_header();
616   server.sendContent(webpage);
617   webpage = "";
618 }
619
620 void SendHTML_Content(){
621   server.sendContent(webpage);
```

```
622   webpage = "";
623 }
624
625 void SendHTML_Stop(){
626   server.sendContent("");
627   server.client().stop();// Stop is needed because no content length ...
          was sent
628 }
629
630 void SelectInput(String heading1, String heading2, String command, ...
       String arg_calling_name){
631   SendHTML_Header();
632   webpage += F("<h3 class='rcorners_m'>"); webpage += heading1 + ...
          "</h3><br>";
633   webpage += F("<h3>"); webpage += heading2 + "</h3>";
634   webpage += F("<FORM action='/"); webpage += command + "' ...
          method='post'>"; // Must match the calling argument e.g. ...
          '/chart' calls '/chart' after selection but with arguments!
635   webpage += F("<input type='text' name='"); webpage += ...
          arg_calling_name; webpage += F("' value=''><br>");
636   webpage += F("<type='submit' name='"); webpage += ...
          arg_calling_name; webpage += F("' value=''><br><br>");
637   append_page_footer();
638   SendHTML_Content();
639   SendHTML_Stop();
640 }
641
642 void ReportSDNotPresent(){
643   SendHTML_Header();
644   webpage += F("<h3>No SD Card present</h3>");
645   webpage += F("<a href='/'>[Back]</a><br><br>");
646   append_page_footer();
647   SendHTML_Content();
648   SendHTML_Stop();
649 }
650
651 void ReportFileNotPresent(String target){
652   SendHTML_Header();
653   webpage += F("<h3>File does not exist</h3>");
654   webpage += F("<a href='/"); webpage += target + ...
          "'>[Back]</a><br><br>";
655   append_page_footer();
656   SendHTML_Content();
657   SendHTML_Stop();
658 }
659
660
```

```
661
662 //--------------------------------------------------------------------------------
663 // Fun  es do Diretorio do SD Card
664
665 void SD_dir(){
666   if(SD_present){
667     File root = SD.open("/");
668     if(root){
669       root.rewindDirectory();
670       SendHTML_Header();
671       webpage += F("<h3 class='rcorners_m'>SD Card Contents</h3><br>");
672       webpage += F("<table align='center'>");
673       webpage += F("<tr><th>Name/Type</th><th style='width:20%'>Type ...
              File/Dir</th><th>File Size</th></tr>");
674       printDirectory("/", 0);
675       webpage += F("</table>");
676       SendHTML_Content();
677       root.close();
678     }
679     else{
680       SendHTML_Header();
681       webpage += F("<h3>No Files Found</h3>");
682     }
683     append_page_footer();
684     SendHTML_Content();
685     SendHTML_Stop();   // Stop is needed because no content length ...
           was sent
686   } else ReportSDNotPresent();
687 }
688
689 void printDirectory(const char *dirname,uint8_t levels){
690   File root = SD.open(dirname);
691 #ifdef ESP8266
692   root.rewindDirectory(); //Only needed for ESP8266
693 #endif
694   if(!root){
695     return;
696   }
697   if(!root.isDirectory()){
698     return;
699   }
700   File file = root.openNextFile();
701   while(file){
702     if(webpage.length()>1000){
703       SendHTML_Content();
704     }
705     if(file.isDirectory()){
```

```
706       webpage += "<tr><td>" + String(file.isDirectory() ? "Dir" : ...
              "File") + "</td><td>" + String(file.name()) + ...
              "</td><td></td></tr>";
707       printDirectory(file.name(),levels-1);
708     }
709     else{
710       webpage += "<tr><td>" + String(file.name()) + "</td>";
711       webpage += "<td>" + String(file.isDirectory() ? "Dir" : ...
              "File") + "</td>";
712       int bytes = file.size();
713       String fsize = "";
714       if (bytes < 1024)                     fsize = String(bytes) + ...
              " B";
715       else if (bytes < (1024 * 1024))       fsize = String(bytes / ...
              1024.0, 3) + " KB";
716       else if (bytes < (1024 * 1024 * 1024)) fsize = String(bytes / ...
              1024.0 / 1024.0, 3) + " MB";
717       else                                  fsize = String(bytes / ...
              1024.0 / 1024.0 / 1024.0, 3) + " GB";
718       webpage += "<td>" + fsize + "</td></tr>";
719     }
720     file = root.openNextFile();
721   }
722   file.close();
723 }
724
725 String SendHTML_leiturafinalizada(){
726   String ptr = "<!DOCTYPE html> <html>\n";
727   ptr += "<head><meta name=\"viewport\" ...
          content=\"width=device-width, initial-scale=1.0, ...
          user-scalable=no\">\n";
728   ptr += "<meta http-equiv=\"Content-Type\" ...
          content=\"text/html;charset=utf-8\">\n";
729   ptr += "<meta http-equiv=\"refresh\" content=\"30\">\n";
730   ptr += "<title>ProcSiMoS - Inicio</title>\n";
731   ptr += "<style>html { font-family: Helvetica; display: ...
          inline-block; margin: 0px auto; text-align: center;}\n";
732   ptr += "body{margin-top: 50px;} h1 {color: #444444;margin: 50px ...
          auto 30px;} h3 {color: #444444;margin-bottom: 50px;}\n";
733   ptr += ".button {display: block;width: 100px;background-color: ...
          #3498db;border: none;color: white;padding: 13px ...
          30px;text-decoration: none;font-size: 25px;margin: 0px auto ...
          35px;cursor: pointer;border-radius: 4px;}\n";
734   ptr += ".button-on {background-color: #5de227;}\n";
735   ptr += ".button-on:active {background-color: #2980b9;}\n";
736   ptr += ".button-off {background-color: #34495e;}\n";
737   ptr += ".button-off:active {background-color: #2c3e50;}\n";
```

```
738   ptr += "p {font-size: 14px;color: #888;margin-bottom: 10px;}\n";
739   ptr += "</style>\n";
740   ptr += "</head>\n";
741   ptr += "<body>\n";
742   ptr += "<h1>ESP32 Web Server - ProcSiMoS Manipulador</h1>\n";
743   if(working){
744     ptr += "<h3>Sua Leitura foi finalizada com Sucesso. O nome do ...
          seu arquivo    " + filename + "</h3>\n";
745     ptr += "<a class=\"button button-on\" href=\"/\">Home</a>\n";
746     ptr += "<a class=\"button button-on\" ...
          href=\"/downloadsd\">Download </a>\n";
747   } else{
748     ptr += "<h3>O manipulador est  efetuando sua leitura. Por favor ...
          AGUARDE.</h3>\n";
749     ptr += "<a class=\"button button-off\" href=\"/cancel\">Cancelar ...
          </a>\n";
750   }
751   ptr += "</body>\n";
752   ptr += "</html>\n";
753   return ptr;
754 }
```