**UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ**

**DOUGLAS MATEUS ALABORA**

# DESENVOLVIMENTO DE ALGORITMOS PARA ESTIMAÇÃO DO DESLOCAMENTO DE TRANSDUTOR ULTRASSÔNICO EM RELAÇÃO AO OBJETO INSPECIONADO

**PATO BRANCO**

**2023**

**DOUGLAS MATEUS ALABORA**

# DESENVOLVIMENTO DE ALGORITMOS PARA ESTIMAÇÃO DO DESLOCAMENTO DE TRANSDUTOR ULTRASSÔNICO EM RELAÇÃO AO OBJETO INSPECIONADO

## Algorithm development for estimating the ultrassonic transducer displacement relative to the inspected object

Trabalho de Conclusão de Curso de Graduação apresentado como requisito para obtenção do título de Bacharel em Engenharia Elétrica do Curso de Bacharelado em Engenharia Elétrica da Universidade Tecnológica Federal do Paraná.

Orientador: Prof. Dr. Giovanni Alfredo Guarneri

**PATO BRANCO**

**2023**

**DOUGLAS MATEUS ALABORA**

# DESENVOLVIMENTO DE ALGORITMOS PARA ESTIMAÇÃO DO DESLOCAMENTO DE TRANSDUTOR ULTRASSÔNICO EM RELAÇÃO AO OBJETO INSPECIONADO

Trabalho de Conclusão de Curso de Graduação apresentado como requisito para obtenção do título de Bacharel em Engenharia Elétrica do Curso de Bacharelado em Engenharia Elétrica da Universidade Tecnológica Federal do Paraná.

Data de aprovação: 21/11/2023

---

Giovanni Alfredo Guarneri
Doutor em Engenharia Elétrica
Universidade Tecnológica Federal do Paraná

---

Dalcimar Casanova
Doutor em Física Computacional
Universidade Tecnológica Federal do Paraná

---

Gustavo Weber Denardin
Doutor em Engenharia Elétrica
Universidade Tecnológica Federal do Paraná

**PATO BRANCO**
**2023**

Dedico este trabalho aos meus pais e ao meu irmão, pelo apoio e inspiração constante. Vocês são a base do meu sucesso. Gratidão eterna. Amo vocês.

## AGRADECIMENTOS

Agradeço primeiramente ao meu orientador, Professor Giovanni, por todo o apoio e dedicação ao longo do processo de pesquisa e escrita deste trabalho. Obrigado pela paciência, disponibilidade e também pela confiança que depositou em mim desde o início, sua experiência e orientação foram essenciais para o meu desenvolvimento profissional e acadêmico.

Gostaria de agradecer à todos os integrantes do grupo de pesquisa AUSPEX, por me receberem de braços abertos na equipe e por me auxiliarem principalmente nas etapas práticas deste trabalho, seus conhecimentos técnicos e práticos foram inestimáveis. Quero deixar registrado também meu agradecimento à Petrobras, por financiar os projetos de pesquisa do grupo.

Agradeço à Universidade Tecnológica Federal do Paraná (UTFPR), por proporcionar o ambiente e recursos necessários para meu desenvolvimento pessoal e profissional. Minha jornada acadêmica foi repleta de experiências enriquecedoras e de oportunidades de aprendizado, esse período ficará marcada na minha vida.

Aos meus amigos e colegas por todo o apoio, motivação, risadas compartilhadas, as noites passadas estudando em grupo e aos momentos de descontração. Com toda a certeza vocês tornaram essa fase da vida mais alegre e leve. Obrigado pelo carinho de todos e por poder compartilhar essa conquista com vocês.

Por fim, agradeço aos meus pais e meu irmão, por todas as palavras de encorajamento, apoio constante e amor incondicional. Obrigado por confiarem em mim e sempre me incentivarem, a presença de vocês foi essencial para me tornar quem sou hoje.

## RESUMO

Por meio da aplicação de Ensaios Não Destrutivos (END), torna-se viável conduzir inspeções sem causar danos ao objeto sob avaliação, preservando, assim, sua integridade física. Nos dias de hoje, a técnica mais utilizada pela indústria é a inspeção por ultrassom, em virtude de sua facilidade de implementação, capacidade de penetrar profundamente no interior das peças e por sua eficiência na transmissão de informações sobre as descontinuidades presentes no objeto inspecionado. Contudo, essa técnica apresenta algumas desvantagens. Para medição do deslocamento do transdutor durante as inspeções, são utilizados *encoders*, e por serem sensores mecânicos, podem gerar erros de medição por conta de escorregamentos do dispositivo com a peça. Além disso, seu funcionamento necessita de cabos para transmitir as informações captadas, o que pode limitar o movimento dos transdutores durante uma varredura. Por conta disso, este trabalho propõe a utilização de algoritmos de *Global Motion Estimation* (GME) em sequências de imagens reconstruídas de ultrassom. Três algoritmos foram implementados em corpos de prova distintos e seus desempenhos foram avaliados para translação e rotação do transdutor. A aquisição dos dados de simulação foi feita por meio do *software* CIVA e os testes com dados reais por meio do sistema M2M *Panther*. Os resultados obtidos mostram que o Algoritmo *Wavelets* apresenta bons resultados tanto para translação quanto para rotação em determinados intervalos de deslocamentos. Já os Algoritmos *Features* e *Keypoints* não tiveram bons resultados com imagens de ultrassom para nenhum dos tipos de movimentos.



**Palavras-chave:** end; gme; *wavelets*; *features*; *keypoints*.

**ABSTRACT**

Through the application of Non-Destructive Evaluation (NDE), it becomes feasible to conduct inspections without causing damage to the object under assessment, thus preserving its physical integrity. Nowadays, the most widely used technique by the industry is ultrasonic inspection due to its ease of implementation, the ability to penetrate deep within the parts, and its efficiency in transmitting information about discontinuities in the inspected object. However, this technique has some disadvantages. To measure the transducer's displacement during inspections, encoders are used, and since they are mechanical sensors, they can introduce measurement errors due to device slippage with the part. Furthermore, their operation requires cables to transmit the collected information, which can limit the movement of the transducers during a scan. Because of this, this work proposes the use of GME algorithms on reconstructed ultrasound image sequences. Three algorithms were implemented on different test specimens, and their performances were evaluated for transducer translation and rotation. Simulation data was acquired using CIVA software, and tests with real data were conducted using the M2M Panther system. The results obtained show that the Wavelets algorithm yields good results for both translation and rotation within certain displacement ranges. On the other hand, the Features and Keypoints algorithms did not perform well with ultrasound images for either type of motion.

**Keywords:** nde; gme; wavelets; features; keypoints.

## LISTA DE FIGURAS

## LISTA DE TABELAS

## LISTAGEM DE CÓDIGOS FONTE

## LISTA DE ABREVIATURAS E SIGLAS

### Siglas

| | |
|---|---|
| AGAST | *Adaptive and Generic Accelerated Segment Test* |
| AUSPEX | *Advanced Ultrasound Signal Processing for Equipmemt InspeCtionS* |
| BRIEF | *Binary Robust Independent Elementary Features* |
| BRISK | *Binary Robust Invariant Scalable Keypoints* |
| CEA | Comissão de Energia Atômica |
| DAS | *Delay and Sum* |
| DCT | *Discrete Cosine Transform* |
| DFT | *Discrete Fourier Transform* |
| END | Ensaios Não Destrutivos |
| FAST | *Features from Accelerated Segment Test* |
| FFT | *Fast Fourier Transform* |
| FMC | *Full Matrix Capture* |
| GME | *Global Motion Estimation* |
| NDE | Non-Destructive Evaluation |
| ORB | *Oriented FAST and Rotated BRIEF* |
| ROI | *Region of Interest* |
| SIFT | *Scale Invariant Feature Transform* |
| SURF | *Speeded Up Robust Features* |
| TFM | *Total Focusing Method* |
| UTFPR | Universidade Tecnológica Federal do Paraná |

**SUMÁRIO**

# 1 INTRODUÇÃO

O petróleo está presente na vida das pessoas de diversas formas, seja como fonte de energia ou como matéria-prima para fabricação de plásticos, tintas ou outros derivados. Segundo a Empresa de Pesquisa Energética, o petróleo é responsável por cerca de 31% da matriz energética mundial, sendo assim, a maior fonte de energia atualmente (EPE, 2020).

O Brasil é considerado um país de referência mundial em pesquisas no setor petrolífero, e no ano de 2006 atingiu a auto-suficiência na produção de petróleo (PETROBRAS, 2006). Esse aumento na produção está relacionado com o sucesso do país na exploração marítima (*offshore*), o que é resultado de investimentos de empresas petrolíferas em universidades e grupos de pesquisas (SCHIAVI; HOFFMANN, 2015).

Na exploração *offshore*, o petróleo é extraído de reservatórios no interior do solo marítimo e é conduzido para a superfície por meio de tubos de aço. Essas tubulações estão em constante exposição a ambientes agressivos, o que pode resultar em danos às suas estruturas físicas e em possíveis vazamentos (CORAMIK; EGE, 2017). Portanto, é necessário monitorar e avaliar as condições físicas dessas tubulações, a fim de garantir sua operação adequada e mitigar possíveis impactos ambientais e econômicos (CARVALHO *et al.*, 2006).

Por meio de Ensaios Não Destrutivos - END, é possível realizar essas inspeções sem que o objeto inspecionado sofra danos, preservando a integridade física da estrutura (GUERREIRO, 2020). As principais técnicas utilizadas para realizar END são feitas por meio de ensaios com líquidos penetrantes, ultrassom, partículas magnéticas, correntes de Foucault ou por emissões acústicas (SHULL, 2002).

Entre essas técnicas, a que se destaca e é mais utilizada na indústria é a *inspeção por ultrassom*, em virtude de sua facilidade de implementação, capacidade de penetrar profundamente no interior das peças inspecionadas e eficiência na transmissão de informações sobre as descontinuidades encontradas (THOMPSON; THOMPSON, 1985). São essas informações que tornam possível determinar localização, dimensões e orientação dos defeitos existentes no objeto (BAI; VELICHKO; DRINKWATER, 2018; GUARNERI, 2015; DOYLE; SCALA, 1978).

Atualmente, os END por ultrassom utilizam *encoders* para determinar o deslocamento, rotação, posição e direção do transdutor (CANDIDO, 2020). Os *encoders* possuem a vantagem de serem de baixo custo e de fácil manipulação, porém, por serem sensores eletro-mecânicos possuem algumas desvantagens na sua implementação. Durante as inspeções, devido ao movimento do transdutor ultrassônico, podem ocorrer escorregamentos do dispositivo em contato com a peça, ocasionando um erro na aquisição dos dados. Além disso, os cabos necessários para o funcionamento dos *encoders* podem limitar o movimento dos transdutores durante uma varredura.

Com o intuito de determinar translação e rotação de transdutores ultrassônicos sem o auxílio de *encoders*, este trabalho propõe o uso de algoritmos de estimativa de movimento global, do inglês GME, em imagens reconstruídas de ultrassom. Esse tipo de algoritmo consiste em

estimar o movimento entre dois *pixels* adjacentes em uma sequência de imagens (MUSARATH; SUPRIYA; SREELEKHA, 2018) e são muito utilizados em aplicações como: estabilização de câmera (RATAKONDA, 1998) e compressão de vídeos (PARKER *et al.*, 2017).

Para as aplicações desejadas, os algoritmos GME devem possuir grande precisão na medição dos deslocamentos, portanto é necessário o registro das imagens dentro de uma pequena fração de pixel (sub-pixel) (GUIZAR-SICAIROS; THURMAN; FIENUP, 2008). Com a implementação desses algoritmos, espera-se facilitar a realização das inspeções por ultrassom no fundo do mar e aumentar a sua precisão, assim como reduzir os custos necessários.

## 1.1 Objetivos

### 1.1.1 Objetivo geral

Implementar algoritmos de estimativa de movimento global (GME) entre duas imagens reconstruídas de ultrassom para determinar rotação e translação de transdutores ultrassônicos com precisão sub-*pixel*.

### 1.1.2 Objetivos específicos

- Reproduzir os algoritmos GME utilizando imagens naturais, a fim de avaliar sua precisão sub-*pixel*.
- Avaliar os algoritmos GME com imagens reconstruídas de ultrassom, visando determinar a viabilidade de sua utilização neste tipo de aplicação.
- Comparar e avaliar o desempenho dos algoritmos com dados reais e simulados.

## 1.2 Organização do trabalho

Este trabalho está estruturado da seguinte maneira. No Capítulo 2 é detalhado todo o embasamento teórico deste trabalho, explicando o funcionamento dos END por ultrassom, demonstrando os cálculos considerados para reconstrução das imagens e apresentando as diferentes técnicas de estimação de movimento global utilizadas para reprodução dos algoritmos. No Capítulo 3 são descritos todos os materiais e métodos utilizados para a reprodução dos três algoritmos GME deste trabalho, além de apresentar a estruturação de cada um. O Capítulo 4 expõe os resultados obtidos para testes de rotação e translação entre imagens realizados em dois corpos de prova distintos. Por fim, o Capítulo 5 apresenta as conclusões finais do trabalho e faz um comparativo entre os algoritmos reproduzidos.

# 2 FUNDAMENTAÇÃO TEÓRICA

Nesse capítulo é apresentada toda a teoria em que esse trabalho de baseia. Inicialmente, é feita uma apresentação sobre END por ultrassom, explicando seu princípio de funcionamento e os tipos de ensaios existentes. O modelo de transdutor ultrassônico utilizado e o método de captura escolhido para esse trabalho também são mostrados aqui. Em seguida, é feita uma revisão sobre as técnicas de reconstrução de imagens e de pós-processamento adotadas no trabalho. Por fim, o tema principal dessa monografia é abordado, apresentando os diferentes métodos utilizados por algoritmos GME, descrevendo seu funcionamento e citando suas respectivas vantagens e desvantagens.

## 2.1 Ensaios Não-Destrutivos por Ultrassom

END por ultrassom é a técnica mais utilizada nos dias de hoje pelas indústrias para a detecção de defeitos no interior das peças inspecionadas. Isso se deve ao fato da sua fácil implementação, sua portabilidade, sua capacidade de penetrar profundamente no interior das peças sob inspeção e por sua eficiência na transmissão de informações referentes aos defeitos encontrados (THOMPSON; THOMPSON, 1985). Além da detecção, o método por ultrassom é capaz de determinar localização, dimensões e a orientação das descontinuidades presentes no interior do objeto (BAI; VELICHKO; DRINKWATER, 2018; GUARNERI, 2015; DOYLE; SCALA, 1978).

Contudo, a técnica de END por ultrassom possui algumas desvantagens. É necessário que os operadores possuam experiência, tanto para a realização dos ensaios quanto para a análise dos dados obtidos. Ainda, para as varreduras, na maioria dos casos, é necessária a utilização de um meio acoplante entre o objeto inspecionado e o transdutor ultrassônico, geralmente água ou gel acoplante específico para este uso, o qual se não removido após os ensaios, pode causar danos à estrutura (GUERREIRO, 2020). Por fim, ondas ultrassônicas não são capazes de detectar defeitos planos com comprimento paralelo à direção de propagação das ondas (SHULL, 2002).

### 2.1.1 Princípio de funcionamento

Um sistema de END por ultrassom pode ser dividido em dois outros sistemas: sistema de medição e sistema de aquisição (GUARNERI, 2015). Na Figura 1 está representado um diagrama de blocos do conjunto desses dois sistemas.

O sistema de medição é responsável por gerar e transmitir as ondas de ultrassom no objeto inspecionado, bem como receber os sinais de eco emitidos pelos defeitos presentes no interior da peça e converte-los em sinais elétricos. O sistema de aquisição possui a função

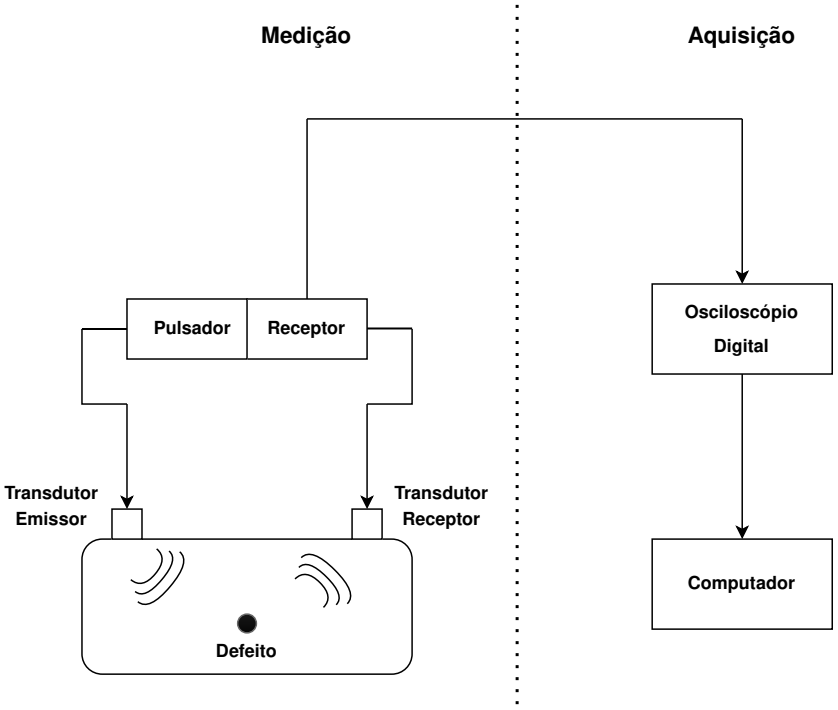


**Figura 1 – Diagrama de blocos de um sistema END ultrassônico.**
**Fonte: Adaptado de Guarneri (2015).**

de digitalizar esses sinais elétricos e disponibilizar os dados para computadores, os quais irão realizar o processamento e as análises dos sinais.

No sistema de medição está presente um elemento pulsador, responsável por gerar pulsos elétricos de curta duração, de aproximadamente 0,1 $\mu s$, e com amplitude na ordem de várias centenas de volts (SCHMERR, 2016). A Figura 2 exemplifica o sinal de saída do pulsador. Esses pulsos excitam o material piezoelétrico presente no interior do transdutor, o qual converte os pulsos elétricos em pulsos mecânicos e que resultam em ondas sonoras de alta frequência (ultrassom).

As ondas ultrassônicas são transmitidas pelo interior do objeto inspecionado e ao entrarem em contato com algum tipo de defeito ou descontinuidade são refletidas em diversas direções na forma de eco (BABOROVSKY; MARSH; SLATER, 1973). Os sinais de eco refle-

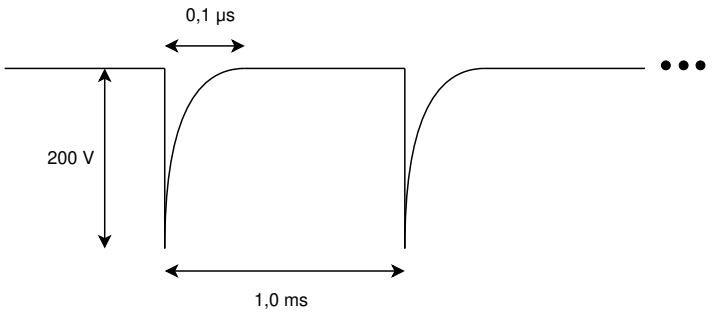


**Figura 2 – Sinal de saída típico de um pulsador ultrassônico.**
**Fonte: Adaptado de Schmerr (2016).**

tidos podem ser então captados pelo transdutor receptor, o qual também possui um material piezoelétrico em seu interior, responsável por converter os sinais de eco em sinais elétricos.

A digitalização desses sinais elétricos é feita no sistema de aquisição por meio de um conversor analógico-digital, localizado no interior do osciloscópio digital. Após amplificados, a amplitude desses sinais representa a energia instantânea dos ecos em função do tempo. Esses sinais são chamados de *A-scan* (SCHMERR, 2016) e na Figura 3 é apresentado um exemplo desse sinal.

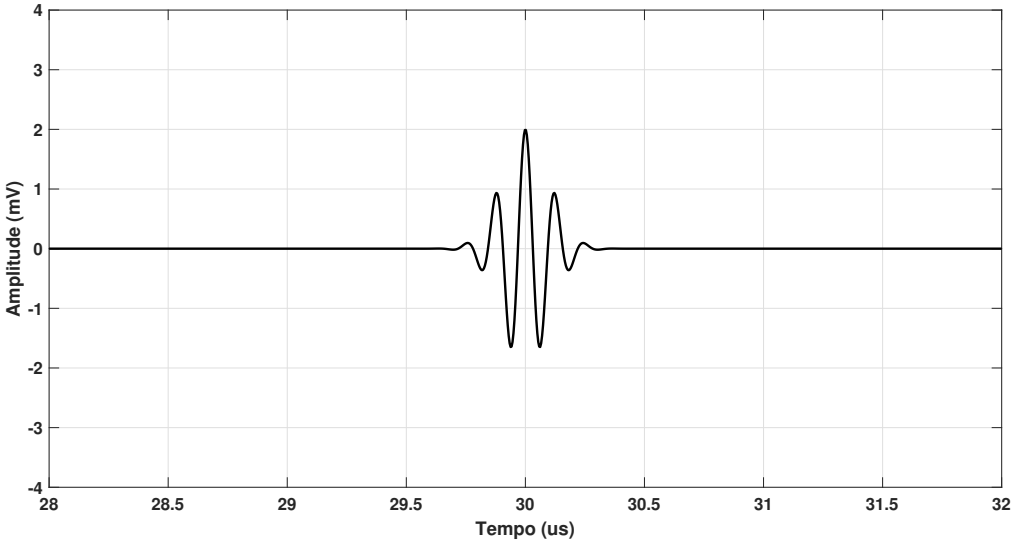


**Figura 3 – Exemplo de sinal A-scan.**
**Fonte: Adaptado de Guarneri (2015).**

### 2.1.2 Tipos de ensaios

Os ensaios não destrutivos por meio de ultrassom podem ser realizados de duas formas diferentes, por meio do contato direto ou indireto do transdutor ultrassônico com a peça inspecionada e por meio da submersão de ambos em algum material acoplante, geralmente a água.

O primeiro modo é chamado de ensaio por contato, nele o transdutor é colocado em contato direto com o objeto por meio de uma camada fina de algum acoplante ou por meio do uso de uma sapata (SOUZA, 2019), como mostra a Figura 4. O material acoplante tem a finalidade de possibilitar a compatibilidade acústica entre o material da peça inspecionada e o transdutor, melhorando a capacidade de penetração das ondas no objeto (GUERREIRO, 2020).

A utilização deste método tem como vantagem a maior intensidade das ondas sonoras no objeto, contudo apresenta algumas desvantagens. O material acoplante deve ser selecionado conforme o tipo de material que está sendo inspecionado, a fim de evitar o comprometimento da estrutura física da peça por conta de reações químicas indesejadas. Ainda, inspeções

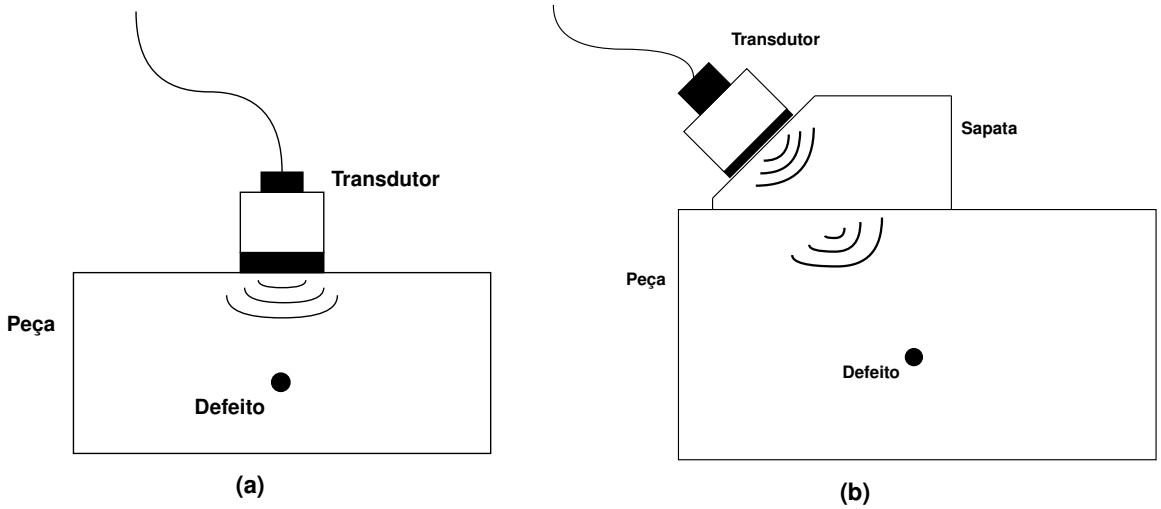


**Figura 4 – Posicionamento de transdutor para inspeção por contato. (a) Utilizando gel acoplante. (b) Utilizando sapata.**
**Fonte: Autoria Própria.**

em superfícies extensas podem ser inconvenientes de serem realizadas, já que necessitam da aplicação de acoplante em toda a área a ser inspecionada (COLLINGWOOD, 1987).

O outro tipo de inspeção é chamado de ensaio por imersão. Nesse tipo de ensaio, o transdutor e a peça são submersos em um meio que faz o papel de acoplante (geralmente água) e não ocorre o contato entre eles. Por meio desse método, a água promove o acoplamento constante entre o transdutor e a peça inspecionada e com isso a energia transmitida para a peça pode ser controlada de forma mais precisa durante as inspeções (SCHMERR, 2016). A Figura 5 exemplifica esse tipo de ensaio.

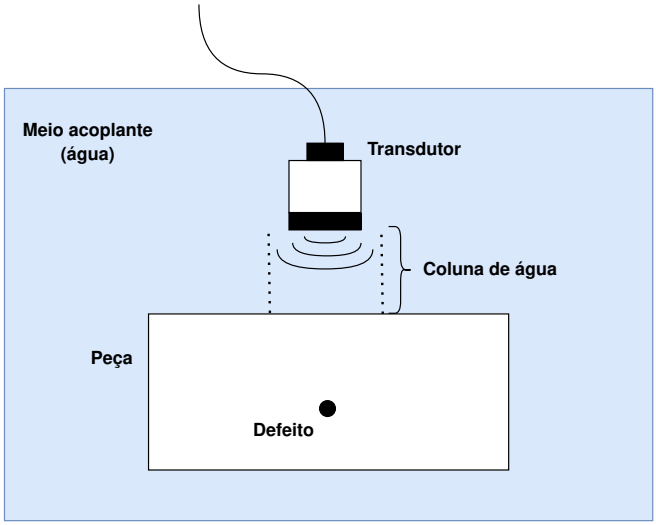


**Figura 5 – Disposição do transdutor para inspeção por imersão.**
**Fonte: Autoria própria.**

### 2.1.3 Transdutor *array*-linear

O transdutor ultrassônico é um dispositivo responsável por emitir e receber ondas acústicas de alta frequência (CHEN *et al.*, 2022). Isso ocorre por meio do material piezoelétrico presente no interior do aparelho, o qual transforma pulsos elétricos em ondas sonoras, bem como ondas sonoras em sinais elétricos.

*Array* é um tipo de transdutor ultrassônico que possui mais de um elemento responsável por emitir e captar as ondas sonoras. Na configuração linear, esses elementos estão dispostos em um mesmo eixo, tornando ele unidimensional (1-D), como mostra a Figura 6. Cada elemento possui formato retangular, com comprimento $L$ e largura $a$. Os elementos estão separados por uma distância constante, denominada de $g$. É comum que, para inspeções, os transdutores sejam especificados pela distância centro-a-centro entre os elementos, chamada de *pitch* $\rho$ (GUERREIRO, 2020).

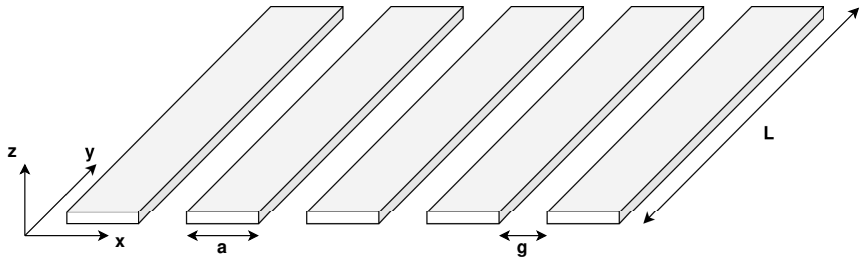


**Figura 6 – Configuração dos elementos de um transdutor *array*-linear.**
**Fonte: Autoria própria.**

Um maior número de elementos no aparelho permite uma maior flexibilidade em relação a emissão de ondas durante uma inspeção, por meio do controle individual do disparo dos elementos (DRINKWATER; WILCOX, 2006). Os transdutores mais comuns apresentam disposição linear com 32, 64 e 128 elementos. A Figura 7 exemplifica duas configurações de disparo diferentes. Em (a) todos os elementos são disparados de forma simultânea, formando uma onda resultante com grande energia, capaz de penetrar mais profundamente. Em (b) há uma diferença no tempo de disparo dos elementos, produzindo uma onda resultante focada.

### 2.1.4 *Captura Full Matrix Capture* - FMC

A captura de dados se refere ao método utilizado para emitir e receber os sinais acústicos durante as inspeções por ultrassom. Uma técnica muito adotada com o uso de transdutores do tipo *array*-linear é a captura *Full Matrix Capture* (FMC). Esse método é baseado em disparar um único elemento do transdutor e receber o sinal de eco por meio de todos eles, repetindo isso até todos os elementos terem realizado o seu disparo (LI *et al.*, 2013). A Figura 8 exemplifica esse tipo de captura para um transdutor *array*-linear de 6 elementos. O processo de captura se inicia com o disparo do Elemento 0, enquanto todos os elementos do transdutor, inclusive o Ele-

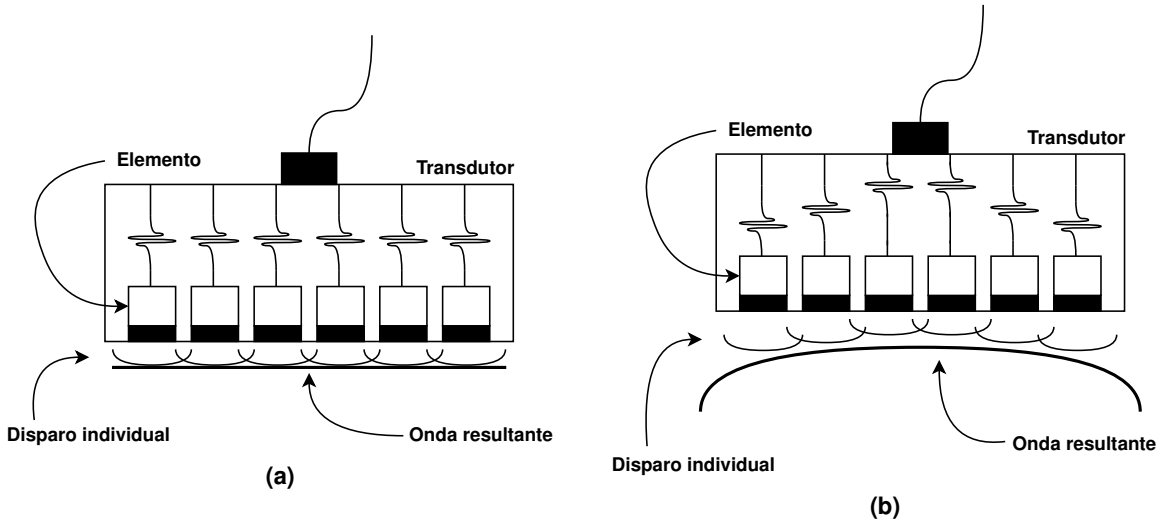


**Figura 7 – Transdutor *array*-linear com diferentes configurações de disparos dos elementos. (a) Disparo simultâneo. (b) Disparo com instantes de tempo diferentes.**
**Fonte: Adaptado de Guerreiro (2020).**

mento 0, recebem os sinais de eco. Em seguida, o Elemento 1 é disparado e ocorre o mesmo processo. Isso se repete para todos os elementos, finalizando a captura com o Elemento 5.

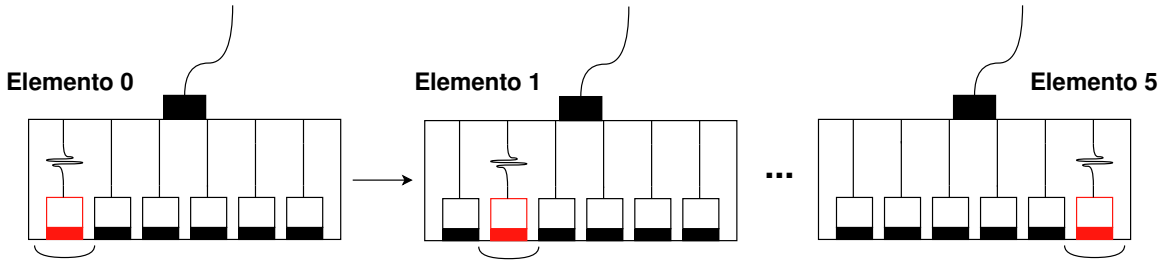


**Figura 8 – Sequência de disparos para captura FMC.**
**Fonte: Autoria própria.**

Cada emissão resulta em um conjunto de dados que pode ser representado por uma matriz 2-D, formada pelos dados de A-scan e chamada de B-scan. Os dados obtidos a partir do disparo de cada elemento do transdutor podem ser agrupados em uma única matriz 3-D, a qual recebe o nome de "matriz de dados FMC". A Figura 9 ilustra o processo de formação dessa matriz.

O método de captura FMC oferece a maior quantidade de informação que pode ser obtida por meio do uso de um transdutor do tipo *array*. Contudo, esse tipo de captura oferece algumas desvantagens. Cada elemento do transdutor deve individualmente emitir um pulso, o que pode tornar o processo de inspeção demorado. Ainda, a quantidade de dados gerados durante uma inspeção é elevado, uma vez que cada elemento gera uma matriz 2-D de dados.

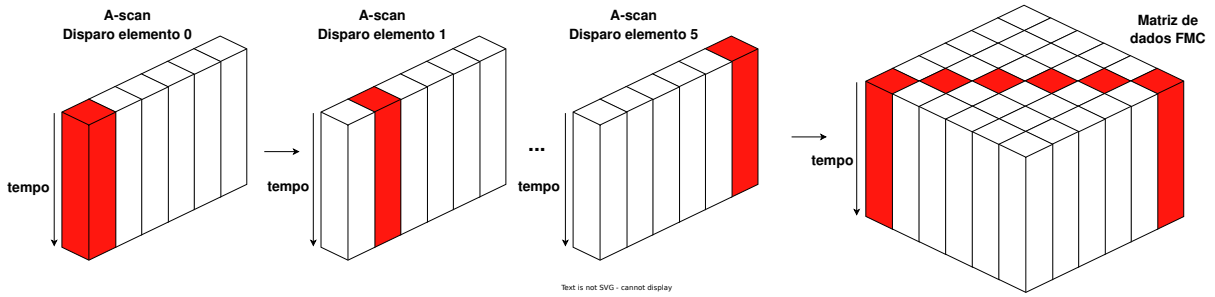


**Figura 9 – Conjunto de dados de uma captura FMC.**
**Fonte: Autoria própria.**

## 2.2 Reconstrução de Imagens - *Total focusing method*

A interpretação dos dados brutos obtidos a partir de END pode ser uma tarefa complicada. Para facilitar esse processo e obter uma representação mais precisa sobre a estrutura interna do objeto inspecionado, técnicas de processamento de sinais são utilizadas. Essas técnicas consistem em criar imagens do interior da peça a partir de dados de A-scan registrados durante a sua inspeção (CANDIDO, 2020). Por conta disso, é comum que essas técnicas de processamento sejam chamadas de algoritmos de reconstrução de imagens ou algoritmos de imageamento.

A técnica mais utilizada para o processamento de dados de FMC é utilizando o algoritmo *Total Focusing Method* (TFM), o qual é uma implementação no domínio do tempo e se baseia no conceito de atraso-e-soma (DAS, do inglês *Delay-and-Sum*). Em uma imagem formada por um algoritmo DAS, cada ponto é resultado da combinação de vários sinais de eco recebidos pelos elementos do transdutor, assim, um sinal A-scan detectado por um elemento pode ser confirmado por sinais detectados pelos outros elementos durante uma inspeção. A Figura 10 demonstra o princípio de funcionamento de um algoritmo DAS.

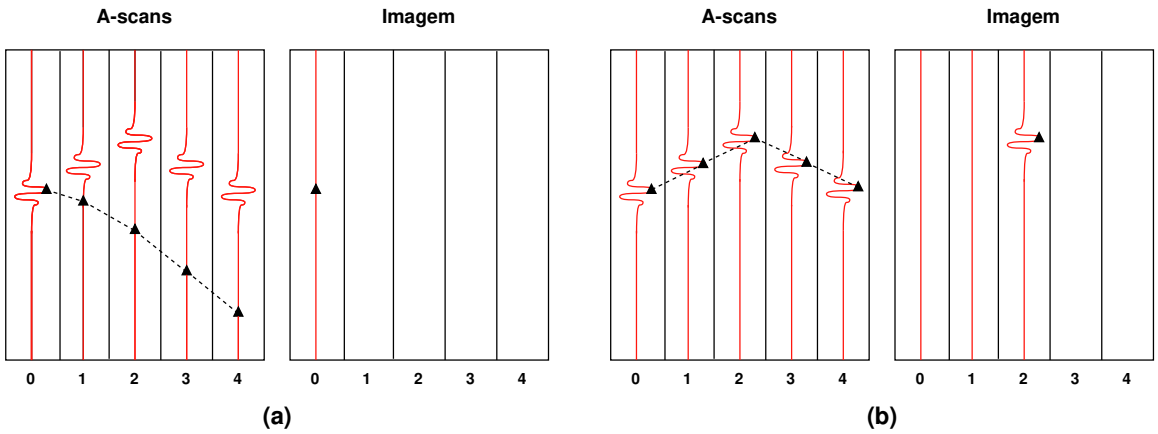


**Figura 10 – Princípio de funcionamento do algoritmo DAS com dados de A-scan. (a) Reconstrução da coluna 0 da imagem. (b) Reconstrução da coluna 2 da imagem.**
**Fonte: Adaptado de Guerreiro (2020).**

Na Figura 10 é apresentado um transdutor com 5 elementos durante uma captura FMC. A ilustração à esquerda (a) representa a reconstrução da Coluna 0 da imagem a partir de dados obtidos por disparos do elemento 0, já a ilustração à direita (b), representa a Coluna 2 da imagem a partir de disparos do Elemento 2 do transdutor.

O sinal A-scan detectado pelo Elemento 0 do transdutor apresenta uma característica típica de um sinal eco de uma descontinuidade, o que poderia significar que há um defeito na peça logo abaixo desse elemento. Contudo, isso não é confirmado por nenhum outro elemento do transdutor. Na reconstrução da Coluna 0, o pico do sinal detectado é destacado por um triângulo preto. Caso o sinal fosse de uma descontinuidade presente naquela posição, esse mesmo pico seria detectado pelos outros elementos, porém com um atraso decorrente da propagação da onda na peça inspecionada. Isso não ocorre, já que as amostras detectadas pelos outros elementos não possuem nenhum pico na posição específica. Já na reconstrução da Coluna 2, o pico no sinal de A-scan detectado pelo Elemento 2 é confirmado pelos outros elementos do transdutor com o atraso esperado, o que indica que naquele local há uma descontinuidade na peça.

Portanto, o algoritmo DAS consiste em somar as amostras que possuem informações sobre um mesmo ponto do objeto, sobrepondo os sinais A-scan e resultando em amplitudes elevadas nos pontos em que há de fato um defeito e amplitudes reduzidas quando não há a confirmação por outros elementos do transdutor.

A Figura 11 exemplifica a distância percorrida pela onda durante uma captura do tipo FMC. Nesse caso o Elemento 0 do transdutor, posicionado em $x_e$, gera um pulso ultrassônico que percorre o interior do objeto inspecionado até um ponto $(x_0, z_0)$, apresentando um deslocamento dado por $d_{e,0} = \sqrt{(x_e - x_0)^2 + z_0^2}$. Caso haja uma descontinuidade nesse ponto, a onda será refletida em forma de eco, o qual será recebido pelo Elemento 3 do transdutor, que se encontra em $x_r$. A distância percorrida pelo sinal de eco é dada pela expressão $d_{r,0} = \sqrt{(x_r - x_0)^2 + z_0^2}$. Assim, o tempo entre emissão e recepção da onda de ultrassom é calculado pela Equação (1)

$$\tau(x_e, x_r, x, z) = \frac{\sqrt{(x_e - x)^2 + z^2} + \sqrt{(x_r - x)^2} + z^2}{c}, \tag{1}$$

em que $c$ é a velocidade de propagação do som no objeto.

Assim, denominando $s(x_{(E,R)}, t)$ o sinal A-scan referente a emissão pelo Elemento $E$ e recepção pelo Elemento $R$, a amostra $s(x_{(E,R)}, \tau(x_e, x_r, x, z))$ possui informação sobre o ponto $(x, z)$ no interior do objeto inspecionado. Por meio do conjunto de informações da combinação entre todos os elementos emissores e receptores do transdutor, é possível formar a imagem de um ponto específico da peça (JENSEN *et al.*, 2006), como mostra a Equação (2)

$$o(x, z) = \sum_{j=0}^{N-1} \sum_{i=0}^{N-1} s(x_{(i,j)}, \tau(x_j, x_i, x, z)), \tag{2}$$

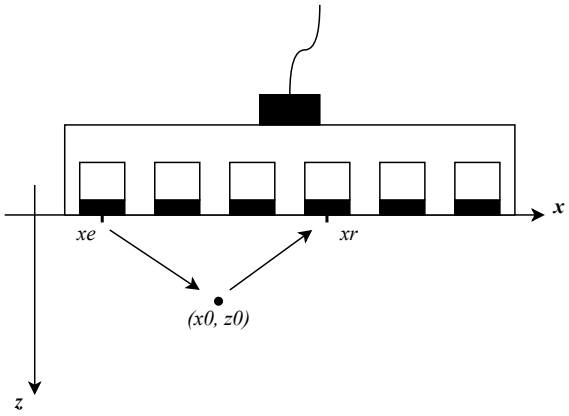


**Figura 11 – Distância percorrida pela onda de ultrassom para aquisição por FMC.**
**Fonte: Autoria própria.**

em que $o(x, z)$ é o ponto determinado no objeto e $N$ a quantidade de elementos no transdutor.

Na Figura 12 é apresentado um corpo de prova de alumínio simulado no *software* CIVA. Ele possui 80 mm de comprimento, 60 mm de altura e 25 mm de profundidade (não mostrado). A peça possui um defeito do tipo furo passante com 1 mm de diâmetro e localizado no ponto $(x, z) = (40, 40)$, considerando como origem do sistema de coordenadas cartesianas o canto superior esquerdo do objeto. O transdutor utilizado na simulação é do tipo *array*-linear com 64 elementos, frequência central de 5 MHz, *pitch* de 0,3 mm e frequência de amostragem de 170 MHz.

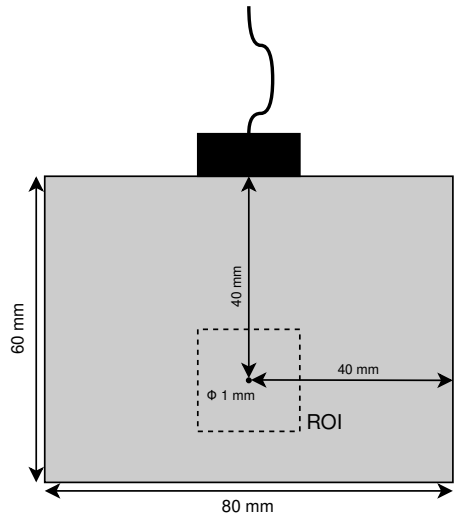


**Figura 12 – Peça simulada no CIVA.**
**Fonte: Autoria própria.**

A Figura 13 mostra as imagem reconstruídas pelo algoritmo TFM a partir dos dados da captura FMC da simulação realizada no corpo de prova mostrado na Figura 12. Por meio da imagem reconstruída pelo algoritmo TFM (esquerda) é possível identificar a existência de uma descontinuidade no interior da peça, a qual fica mais nítida após a utilização técnicas de pós-

processamento (direita), permitindo a identificação das dimensões e do tipo de defeito presente na peça inspecionada.

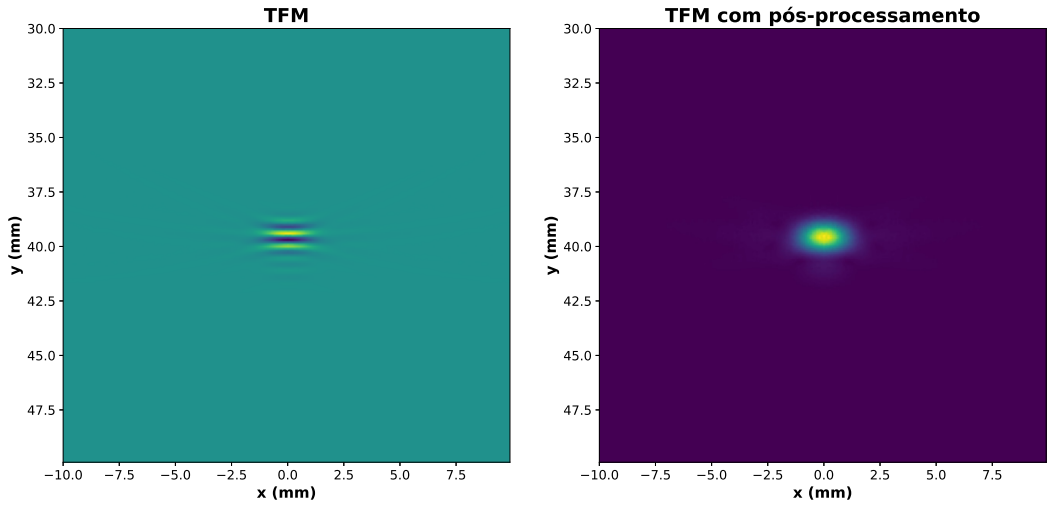


**Figura 13 – Imagem reconstruída pelo algoritmo TFM (Esquerda) e imagem reconstruída pelo algoritmo TFM com pós-processamento (Direita)**

**Fonte: Autoria própria.**

## 2.3 Pós-Processamento

Com o objetivo de aprimorar os resultados obtidos com a reconstrução de imagens de ultrassom, técnicas de pós-processamento podem ser aplicadas. Dentre os métodos mais utilizados, destacam-se os métodos de extração de envelope por amplitude do sinal e a normalização dos dados, também conhecidos por *envelope* e *normalize* (GUERREIRO, 2020).

Essencialmente o *envelope* é uma função que descreve o contorno de um sinal e é determinado por um detector de envelopes que interliga os seus picos (MATHWORKS, 2023b). Um método comum utilizado para detectar envelopes é baseado na Transformada de Hilbert, a qual é responsável por criar o sinal analítico do sinal original.

A Transformada de Hilbert de uma função $x(t)$ pode ser definida por

$$\hat{x}(t) = \frac{1}{\pi} \int_{-\infty}^{\infty} \frac{x(\tau)}{t-\tau} d\tau. \tag{3}$$

Como a Equação (3) é uma integral de convolução, ela pode ser reescrita conforme a Equação (4) (MARQUES, 2013)

$$\hat{x}(t) = \frac{1}{\pi t} * x(t). \tag{4}$$

Assim, matematicamente, o envelope de um sinal pode ser descrito pela Equação (5)

$$e(t) = \sqrt{x(t)^2 + \hat{x}(t)^2}. \tag{5}$$

A Figura 14 exemplifica o funcionamento dessa função.

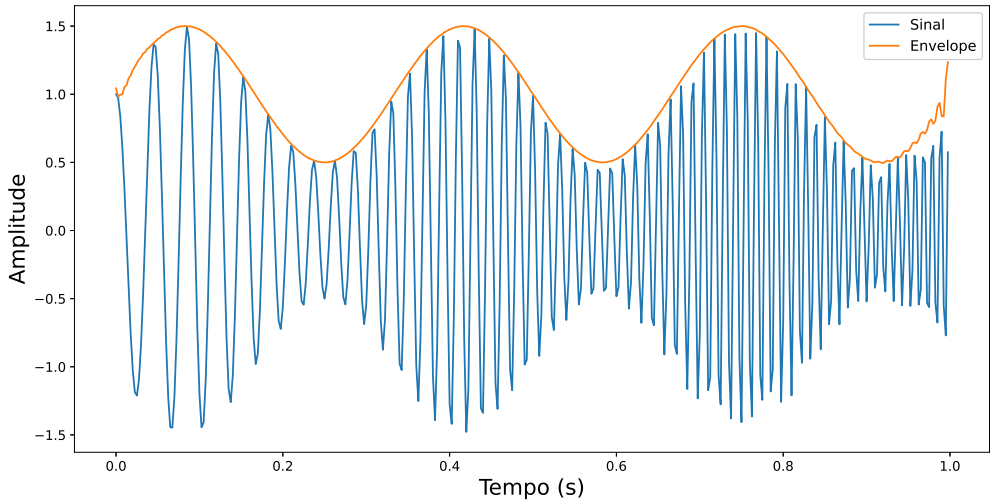


**Figura 14 – Exemplo da função envelope de um sinal**
**Fonte: Adaptado de Feldman (2011).**

A função *normalize* é responsável por ajustar os valores de um conjunto de dados em uma escala comum, aplicando neles um fator de escala. Existem diferentes métodos para isso, como por exemplo a normalização *Z-score*, em que os valores são ajustados para a faixa (-1, 1). Outra técnica comum é a normalização min-máx, em que os valores dos dados são adaptados de acordo com a Equação (6)

$$X_{normalizado} = a + \left( \frac{X - X_{min}}{X_{max} - X_{min}} \right) \cdot (b - a), \tag{6}$$

em que a faixa de valores é dada por (a, b) e $X_{min}$ e $X_{max}$ são, respectivamente, o menor e o maior valor dentro do conjunto de dados (MATHWORKS, 2023c).

## 2.4 Algoritmos para Estimativa de Movimento Global - (GME)

O movimento global é o movimento que afeta toda a cena apresentada em uma sequência de imagens. Ele geralmente é causado pela movimentação da câmera ou pela mudança dos objetos na cena como um todo. O processo para estimar esses movimentos é chamado de Estimativa de Movimento Global, do inglês *Global Motion Estimation* - GME. Essa técnica é uma importante ferramenta utilizada na visão computacional e no processamento de imagens, podendo ser aplicada em estabilização de imagens, compressão de vídeos, remoção de ruídos e detecção de objetos em movimento (SU; SUN; HSU, 2003). A Figura 15 exemplifica alguns dos movimentos que são esperados em uma sequência de imagens.

Existem diferentes abordagens para estimar o movimento global entre imagens e diversos algoritmos foram desenvolvidos para isso. Alguns exemplos dessas abordagens são:

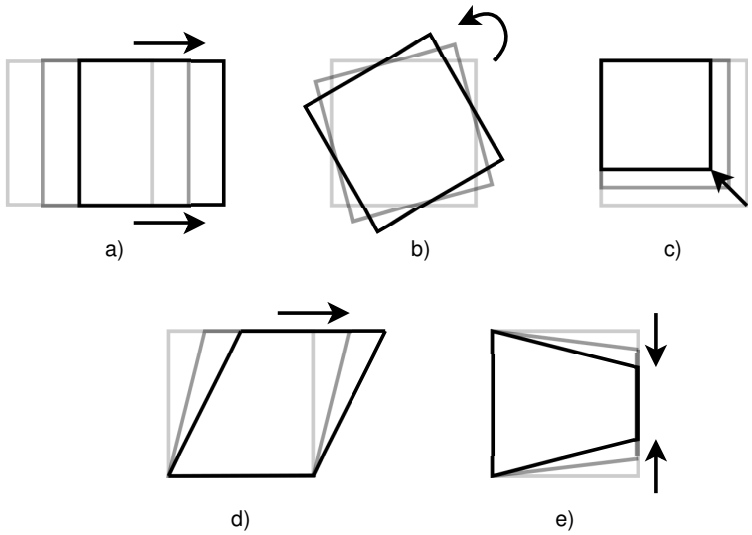


**Figura 15 – Tipos de movimentos entre imagens. (a) Translação (b) Rotação (c) Zoom (d) Inclinação (e) *Pan***
**Fonte: Adaptado de Candido (2020).**

estimação de movimento com base nas fases (KUMAR; BISWAS; NGUYEN, 2004), estimação de movimento baseada em *features* (TORR; ZISSERMAN, 2000), estimação de movimento por fluxo óptico (RAVESHIYA; BORISAGAR, 2012) e estimação de movimento baseada em blocos (BARJATYA, 2004). Com o propósito de dar seguimento à pesquisa de Candido (2020), este estudo concentrará seus esforços na abordagem fundamentada na diferença de fases. Adicionalmente, com o objetivo de enriquecer os resultados deste trabalho, será explorado o método de estimação por registro de características, uma vez que essa técnica demonstra eficácia em sequências de imagens com baixa quantidade de informações.

### 2.4.1 *Phase-based motion estimation*

Uma das alternativas mais utilizadas para estimar o movimento entre imagens é com base na diferença de fases entre as transformadas de imagens correspondentes. Para isso, é possível utilizar a Transformada Discreta de Cosseno (*Discrete Cosine Transform* (DCT)) ou a Transformada Rápida de Fourier (*Fast Fourier Transform* (FFT)), sendo essa a mais vantajosa (LI *et al.*, 2004).

Quando há o movimento entre duas imagens, os picos das frequências espaciais presentes nas imagens se deslocam em uma determinada direção. A partir dessa diferença, por meio da correlação das fases, é possível determinar o deslocamento dos picos e estimar o valor real entre as imagens (FUJISAWA; IKEHARA, 2019).

Essa abordagem recebe o nome de correlação de fases, do inglês *Phase Correlation*, e é considerada uma técnica robusta e precisa para estimar deslocamento. Entretanto, sua aplicação em tempo real para imagens inteiras não é adequada, por conta do seu elevado custo

computacional para realizar a Transformada de Fourier e a Transformada Inversa de Fourier (ERTURK, 2003).

O deslocamento entre duas imagens, utilizando o *Phase Correlation*, é estimado por meio da Equação (7)

$$R(x,y) = \mathcal{F}^{-1}\left(\frac{\mathcal{F}(F(x,y)) \circ \mathcal{F}(G(x,y))^*}{|\mathcal{F}(F(x,y)) \circ \mathcal{F}(G(x,y))^*|}\right), \tag{7}$$

em que $\mathcal{F}(F(x,y)$ e $\mathcal{F}(G(x,y)$ são, respectivamente, a Transformada de Fourier da imagem original e da imagem deslocada, $\mathcal{F}^{-1}$ é a Transformada Inversa de Fourier, $\circ$ é a multiplicação elemento a elemento das matrizes das imagens e $\mathcal{F}(G(x,y))^*$ é o complexo conjugado de $\mathcal{F}(G(x,y))$. Ainda, a divisão matricial também é realizada na forma elemento a elemento.

O deslocamento $R(x,y)$ estimado pela Equação (7) representa apenas a translação nos eixos $x$ e $y$ das imagens. Para que seja possível estimar a rotação e a mudança de escala, é necessário aplicar a Transformada Log-Polar na sequência de imagens antes de realizar a Transformada de Fourier (FUJISAWA; IKEHARA, 2019).

O motivo disso é o fato de a rotação e a mudança de escala no plano cartesiano serem representadas por simples translações no domínio log-polar, tornando possível a utilização da técnica de *Phase Correlation* (SARVAIYA; PATNAIK; BOMBAYWALA, 2009). As coordenadas no plano log-polar são dadas pela distância logarítmica $\rho$ e o ângulo $\theta$. Qualquer ponto $(x,y)$ no plano cartesiano pode ser expresso por meio da Equação (8)

$$\rho = log\left(\sqrt{\omega_x^2 + \omega_y^2}\right), \theta = tan^{-1}\left(\frac{\omega_x}{\omega_y}\right), \tag{8}$$

em que $\omega_x$ e $\omega_y$ são as distâncias dos pontos $(x,y)$ até o centro da imagem.

Com isso, por meio da Transformada Log-Polar, as imagens cartesianas $F(x,y)$ e $G(x,y)$ são convertidas para $F(\rho,\theta)$ e $G(\rho,\theta)$ no domínio log-polar. Assim, a rotação e a mudança de escala podem ser estimadas por meio da Equação (9)

$$R(\rho,\theta) = \mathcal{F}^{-1}\left(\frac{\mathcal{F}(F(\rho,\theta)) \circ \mathcal{F}(G(\rho,\theta))^*}{|\mathcal{F}(F(\rho,\theta)) \circ \mathcal{F}(G(\rho,\theta))^*|}\right). \tag{9}$$

Com o intuito de reduzir o custo computacional de algoritmos baseados em *phase-correlation*, Zhou *et al.* (2022) desenvolveu uma técnica utilizando parcelas menores da sequência de imagens. Esses fragmentos são chamados de sub-imagens, eles são resultado da divisão da imagem original e deslocada e devem apresentar dimensões semelhantes entre si.

Com dimensões menores, menor será o tempo de processamento do algoritmo, porém, a quantidade de informações disponíveis nas sub-imagens diminuem drasticamente e com isso a precisão do método é prejudicada.

Após as sub-imagens serem definidas, é necessário determinar qual delas apresenta a maior quantidade de informações. Para isso, a Transformada de Wavelet é aplicada em cada uma delas, decompondo as imagens em uma sub-banda de baixa frequência, chamada de LL,

e em três sub-bandas de alta frequência, denominadas de LH, HL e HH. Por meio disso, um coeficiente Wavelet pode ser determinado, de acordo com a Equação (10)

$$E = (LL)^2 + (LH)^2 + (HL)^2. \tag{10}$$

Assim, a sub-imagem que apresentar o maior coeficiente $E$ será escolhida para ser utilizada no algoritmo por *phase-correlation*. Ainda, nota-se que a sub-banda HH é desconsiderada nesse método, o que se deve à sua natureza ruidosa (ZHOU *et al.*, 2022).

### 2.4.2 *Features-based registration*

O registro de características de imagens é uma técnica amplamente utilizada em diferentes aplicações da visão computacional. Essa abordagem consiste em três passos principais: detecção de pontos de interesse, descrição de características e correspondência de características (BAY; TUYTELAARS; GOOL, 2006).

Primeiramente, pontos-chave são identificados em ambas as imagens por meio de um detector, chamado de *keypoint detector*. Em seguida, um descritor é aplicado com o objetivo de representar o entorno de cada ponto de interesse por meio de um vetor de características. Por fim, os pontos-chave da sequência de imagens podem ser correspondidos por meio de um comparador, chamado de *keypoints matcher* (LEUTENEGGER; CHLI; SIEGWART, 2011). Essa combinação normalmente é feita com base na distância entre os vetores de características correspondentes e com isso é possível determinar translação, rotação e mudança de escala entre as imagens.

Existem diferentes detectores e descritores de pontos-chave de imagens, cada um com suas características e especialidades. Alguns exemplos comuns de algoritmos utilizados para visão computacional são: *Scale Invariant Feature Transform* (SIFT), *Speeded Up Robust Features* (SURF), *Oriented FAST and Rotated BRIEF* (ORB) e *Binary Robust Invariant Scalable Keypoints* (BRISK). Por serem abordagens mais recentes, serem adequados para aplicações em tempo real e apresentarem robustez a ruídos, este trabalho irá se aprofundar nos dois últimos exemplos.

#### 2.4.2.1 *Oriented FAST and Rotated BRIEF - ORB*

O algoritmo ORB é uma técnica eficiente para detectar e descrever pontos-chave em imagens. Ele consiste na união do algoritmo *Features from Accelerated Segment Test* (FAST), o qual determina os pontos nas imagens utilizando o método de centróide em escala de cinzas, com o descritor *Binary Robust Independent Elementary Features* (BRIEF) (DAI; WU, 2023).

Quando comparado com outros algoritmos de registro de características já existentes, ORB se apresenta vantajoso, por conta de sua robustez à ruídos e de seu rápido processa-

mento, o que o torna adequado para aplicações em tempo real que envolvem grandes quantidades de dados (RUBLEE *et al.*, 2011).

Apesar de suas vantagens, o algoritmo ORB apresenta algumas limitações de desempenho em aplicações com variação de iluminação. Isso ocorre por conta de seu principio de funcionamento, já que ele opera com um valor fixo de limiar (*threshold*) para determinar qual pixel é um ponto-chave. Isso implica que, em situações com grande variação de iluminação, os valores dos pixels podem ultrapassar os limites estabelecidos, resultando na detecção incorreta de pontos-chave (DAI; WU, 2023).

Inicialmente, o detector FAST seleciona um ponto central $\rho$ qualquer na imagem e em seguida identifica um conjunto de 16 pixels dentro da região com raio de 3 pixels em torno do ponto central, como mostra a Figura 16. O algoritmo então, avalia a diferença entre os valores das escalas de cinza dos pixels 1, 5, 9 e 13, em relação ao ponto central. Caso, no mínimo, três desses pixels forem considerados mais brilhantes ou mais escuros que $\rho$, esse ponto é classificado como ponto-chave (ROSTEN; DRUMMOND, 2006).

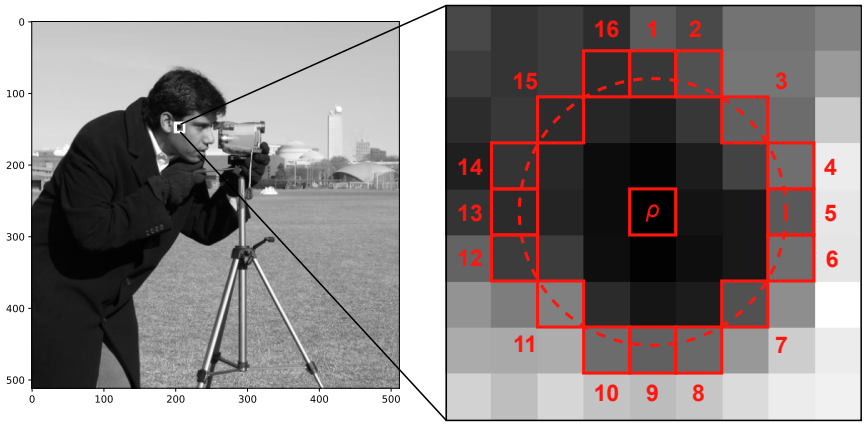


**Figura 16 – Funcionamento do algoritmo para detecção de *keypoints***
**Fonte: Autoria própria.**

Para determinar se um pixel é mais brilhante ou mais escuro, o algoritmo FAST utiliza o seguinte modelo matemático:

$$S = \begin{cases} 0 & \text{se } I_k > I_\rho - T \\ 1 & \text{se } I_k \leq I_\rho + T \end{cases}$$

no qual $I_k$ e $I_\rho$ representam, respectivamente, os valores das escalas de cinza do pixel selecionado e do pixel central. O valor $T$ é dado pelo valor de *threshold* e $S$ o valor binário que indica a luminosidade do pixel, sendo 0 mais brilhante e 1 mais escuro, em comparação com o ponto central.

Já o algoritmo BRIEF, tem como princípio de funcionamento a comparação entre valores de escala de cinza dos pixels. Primeiramente, o descritor estabelece uma região em torno do ponto-chave e seleciona $N$ pares de pixels por meio da amostragem aleatória gaussiana. Essa comparação resulta em um vetor binário teste $\tau$ definido por:

$$\tau(p; a, b) = \begin{cases} 0 & \text{se } p(a) \geq p(b) \\ 1 & \text{se } p(a) < p(b) \end{cases}$$

em que $a$ e $b$ representam o par de pontos e $p(a)$ e $p(b)$ seus respectivos valores de escala de cinza (CALONDER *et al.*, 2010). Por fim, o ponto-chave pode ser descrito pelo seguinte vetor de características:

$$f_N(p) = \sum_{1 \leq i \leq N} 2^{i-1} \tau(p; a, b) \tag{11}$$

#### 2.4.2.2 *Binary Robust Invariant Scalable Keypoints* - BRISK

Da mesma forma que o ORB, o algoritmo BRISK é uma técnica com baixo custo computacional para detectar e descrever pontos-chave em imagens. Além disso, como o nome sugere, esse método é invariante para rotação e mudança de escala, apresentando resultados comparáveis aos obtidos por meio de algoritmos convencionais (LEUTENEGGER; CHLI; SIEGWART, 2011).

A abordagem utilizada pelo BRISK para detectar *features* nas imagens é baseada no algoritmo *Adaptive and Generic Accelerated Segment Test* (AGAST), o qual utiliza os mesmos critérios de detecção do algoritmo FAST, mas com um melhor desempenho para imagens arbitrárias (MAIR *et al.*, 2010).

Apesar de utilizar os mesmos critérios para pixel mais brilhante ou mais escuro que o algoritmo FAST, o método adaptativo AGAST permite o ajuste da região em torno do ponto central de acordo com as características locais do pixel, o que soluciona os problemas com variação de iluminação nas imagens. Além disso, esse método promove ao algoritmo BRISK robustez à mudanças de escala, já que ele determina a escala real de cada ponto-chave da imagem.

A descrição de pontos-chave por meio do algoritmo BRISK é feita a partir da amostragem do entorno do ponto central. O algoritmo determina $N$ amostras circulares igualmente espaçadas e concêntricas com o ponto chave. A Figura 17 exemplifica uma configuração de 27 pontos.

Para evitar os efeitos de *aliasing* na amostragem da intensidade de um ponto $p_i$, a suavização gaussiana é aplicada com desvio padrão $\sigma_i$ proporcional a distância entre os pontos $p_i$ e $p_j$, internos à uma mesma amostra circular. Problemas com *aliasing* surgem quando um sinal

é amostrado a uma taxa inadequada, resultando em artefatos indesejados na representação do sinal, os quais podem atrapalhar a descrição correta de pontos chave nas imagens.

Considerando um par de pontos amostrados $(p_i, p_j)$ de um ponto-chave $k$, é possível estimar o gradiente local $g(p_i, p_j)$ pela equação:

$$g(p_i, p_j) = (p_j - p_i) \cdot \frac{I(p_j, \sigma_j) - I(p_i, \sigma_i)}{\| p_j - p_i \|^2}, \tag{12}$$

em que $I(p_i, \sigma_i)$ e $I(p_j, \sigma_j)$ são os valores suavizados de intensidade nos respectivos pontos.

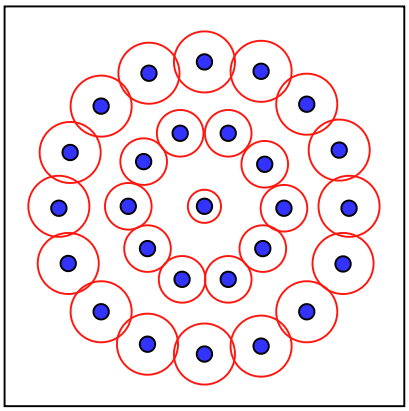

**Figura 17 – Amostragem feita pelo algoritmo BRISK com $N = 27$ pontos. Os pontos azuis representam os locais das amostras; as circunferências vermelhas apresentam raio $\sigma$ correspondente ao desvio padrão da suavização gaussiana aplicada na amostragem.**
**Fonte: Adaptado de Leutenegger, Chli e Siegwart (2011).**

A partir do conjunto $A$ de todos os pares de pontos amostrados:

$$A = (p_i, p_j) \in \mathbb{R}^2 \times \mathbb{R}^2 | i < N \wedge j < i \wedge i, j \in \mathbb{N}, \tag{13}$$

os sub conjuntos de curta distância $S$ e de longa distância $L$ podem ser expressos:

$$\begin{aligned} S &= (p_i, p_j) \in A | \ \| p_j - p_i \| < \delta_{max} \subseteq A, \\ L &= (p_i, p_j) \in A | \ \| p_j - p_i \| > \delta_{min} \subseteq A. \end{aligned} \tag{14}$$

Os valores de $\delta_{min}$ e $\delta_{max}$ representam as distâncias miníma e máxima para o *threshold*, sendo elas determinadas conforme o valor da escala $t$ do ponto-chave $k$, conforme:

$$\begin{aligned} \delta_{min} &= 13{,}67 \cdot t, \\ \delta_{max} &= 9{,}75 \cdot t. \end{aligned} \tag{15}$$

Assumindo que os valores de gradientes locais se anulam, é possível estimar o padrão característico de direcionamento do ponto-chave $k$ e descrevê-lo por meio da iteração dos elementos do sub-conjunto $L$ (LEUTENEGGER; CHLI; SIEGWART, 2011):

$$g = \begin{pmatrix} g_x \\ g_y \end{pmatrix} = \frac{1}{L} \cdot \sum_{(p_i, p_j) \in L} g(p_i, p_j). \tag{16}$$

Tanto para o algoritmo ORB quanto para o BRISK, a correspondência de pontos-chave em imagens é feito por meio do cálculo da distância Hamming, o qual é responsável por mensurar a similaridade entre eles e avaliar as melhores combinações de características.

## 2.5 *Transformada de Wavelet*

A Transformada de Wavelet é uma ferramenta matemática utilizada para analisar imagens e séries temporais (PERCIVAL; WALDEN, 2000). Enquanto a Transformada de Fourier consiste em decompor um sinal em funções senoidais com frequências específicas, a Transformada de Wavelet faz essa decomposição em sinais deslocados e commudança de escala de uma mesma *wavelet* (MATHWORKS, 2023a).

Uma *wavelet* $\psi$ é uma pequena onda (pulso) que cresce e decaí dentro de um intervalo de tempo limitado (STRANG; NGUYEN, 1996) e que possui duas propriedades básicas:

- A integral de $\psi(t)$ é zero

$$\int_{-\infty}^{\infty} \psi(t) dt = 0. \tag{17}$$

- A integral do quadrado de $\psi(t)$ é unitária

$$\int_{-\infty}^{\infty} \psi^2(t) dt = 1. \tag{18}$$

Existem diferentes tipos de *wavelets* para diferentes tipos de aplicações. A Figura 18 ilustra alguns exemplos desse tipo de onda.

A Transformada de Wavelet é uma transformada de tempo-frequência muito utilizada para análise de sinais não estacionários e permite observar as variações dos sinais em diferentes frequências com diferentes resoluções (PERCIVAL; WALDEN, 2000). Essa análise é feita por meio do escalonamento da *wavelet* escolhida. Uma onda de comprimento maior é melhor para verificar componentes de baixa frequência nos sinais. Já uma *wavelet* de comprimento menor é mais adequada para analise de fenômenos de alta frequência (AKANSU; HADDAD, 2001).

A Equação (19) descreve essa análise

$$W\{f(u,s)\} = \int_{-\infty}^{\infty} f(t) \frac{1}{\sqrt{|s|}} \psi^* \left( \frac{t-u}{s} \right) dt. \tag{19}$$

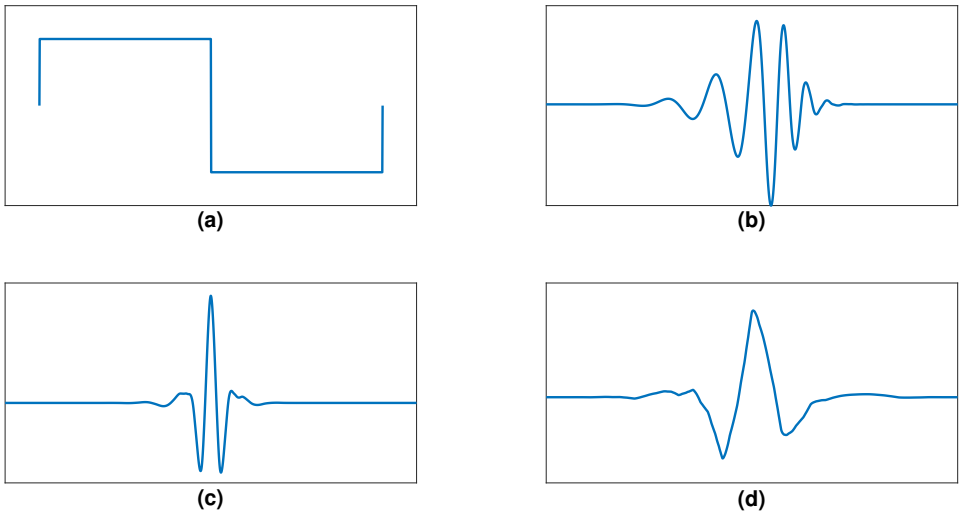


**Figura 18 – Exemplos de famílias de Wavelets. (a) Família Haar (b) Família Daubechies (c) Família Coiflet (d) Família Symlet**

**Fonte: Autoria própria.**

A componente

$$\psi^* \left( \frac{t-u}{s} \right),$$

é obtida a partir da *wavelet* original escolhida para a análise do sinal. A translação dessa onda é determinada por $u$ e o seu escalonamento é dado por $s$.

O coeficiente de Wavelet $W\{f(u,s)\}$ descreve o quão bem o sinal original se ajusta a *wavelet* escolhida, fornecendo importantes informações sobre as propriedades desse sinal. Com isso, essa transformada pode ser utilizada em diferentes aplicações, tais como remoção de ruído, compressão de imagens, análises financeiras, entre outras (MALLAT *et al.*, 2009).

# 3 MATERIAIS E MÉTODOS

Nesse capítulo, será realizada uma descrição dos principais materiais empregados nesse trabalho, juntamente com os métodos utilizados para o desenvolvimento dos algoritmos GME. Além disso, serão detalhados os códigos correspondentes, escritos em Python e baseados na biblioteca `scikit-image`.

## 3.1 Materiais

Essa seção destina-se à descrição dos materiais utilizados no desenvolvimento dos algoritmos desse trabalho. Cada subseção fornecerá um detalhamento específico de cada material, incluindo exemplos de implementações.

### 3.1.1 *Software* CIVA

O CIVA é um *software* para simulações de ensaios não-destrutivos, desenvolvido pela Comissão de Energia Atômica da França (Comissão de Energia Atômica (CEA)) e distribuído, desde 2010, pela EXTENDE. Nele, é possível realizar simulações de diferentes técnicas de END, como por exemplo por ultrassom, ondas-guiadas e correntes parasitas (GUERREIRO, 2020).

Esse *software* permite a simulação de corpos de prova de diferentes geometrias e materiais, além de possibilitar a escolha do tipo e da dimensão dos defeitos aplicados. Ainda, o CIVA permite a configuração de diversos parâmetros do transdutor e do tipo de varredura que será realizada, sendo possível especificar a quantidade de disparos do transdutor e aplicar deslocamento e/ou rotação à ele.

Por meio da variação de diversos parâmetros relevantes para ensaios não-destrutivos, o CIVA se apresenta como uma importante ferramenta para validação de algoritmos de processamento de sinais, além de auxiliar na determinação dos melhores métodos de inspeção para cada ensaio (EXTENDE, 2023).

### 3.1.2 *Framework* AUSPEX

Um *framework* é um conjunto de ferramentas prontas e reutilizáveis que visam facilitar o desenvolvimento de projetos em determinado *software*. Essa estrutura permite que desenvolvedores façam suas modificações para as aplicações específicas desejadas (GAMMA *et al.*, 1995). A partir dessa ideia, o grupo de pesquisa *Advanced Ultrasound Signal Processing for Equipmemt InspeCtionS* (AUSPEX) desenvolveu um *framework* em linguagem Python com ferramentas que auxiliam na leitura, processamento e exibição de dados de inspeções por ul-

trassom (AUSPEX, 2020). A Figura 19 descreve a organização do *framework* utilizado nesse trabalho.

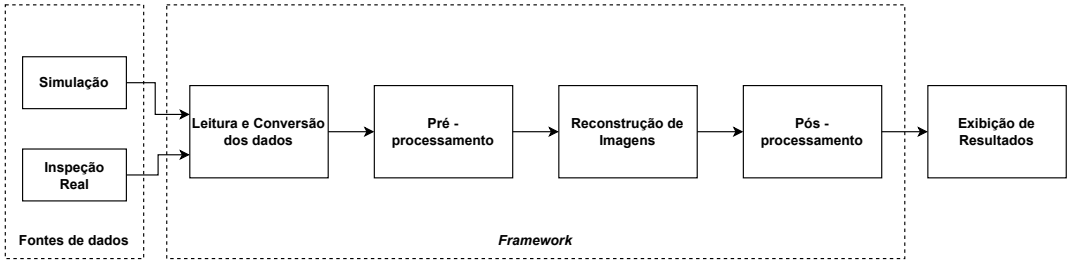


**Figura 19 – Diagrama da organização do *framework* AUSPEX**
**Fonte: Adaptado de AUSPEX (2020).**

Os dados de inspeções podem vir de diferentes fontes, sendo de simulações ou de equipamentos reais. Como cada fonte gera os dados de maneira distinta, a etapa inicial do *framework* realiza a leitura e padronização desses dados. A etapa seguinte é a de pré-processamento, em que são aplicados alguns filtros e adicionado ruídos. Em seguida os dados pré-processados passam pela etapa de processamento, em que algoritmos de reconstrução de imagens são aplicados, como por exemplo o TFM. Ainda, é possível que os dados passem por uma etapa final de pós-processamento, com o objetivo de melhorar a visualização dos dados.

### 3.1.3 Linguagem de programação Python

Desenvolvida no início da década e 90 pelo matemático e programador holandês Guido Von Rossum, Python é uma linguagem de programação de alto nível que surgiu com a proposta de ser uma linguagem simples e versátil, baseando-se na filosofia de legibilidade e minimalismo de códigos (NAGPAL; GABRANI, 2019).

Python é amplamente utilizada em uma variedade de aplicações, como programação WEB, análise de dados, *machine learning*, automação, desenvolvimento de jogos, entre outros. Parte dessa popularidade se deve ao fato de sua biblioteca padrão oferecer uma ampla gama de módulos e funções prontas.

Ainda, por ser uma linguagem gratuita e *open-source*, existem diversas outras bibliotecas criadas pela própria comunidade de usuários que estão documentadas na *WEB* (VASILEV, 2020), o que significa uma grande variedade de recursos disponíveis para facilitar a programação e agilizar o trabalho do desenvolvedor, além de diversificar ainda mais as aplicações dessa linguagem de programação. Por esses fatores, a popularidade do Python apresenta um grande aumento nos últimos anos.

A plataforma Stack Overflow é um site de perguntas e respostas que reúne as comunidades de desenvolvedores de diversas linguagens de programação. Ela é considerada uma das principais plataformas para tirar dúvidas de códigos e por meio do aplicativo Stack Overflow Trends é possível visualizar quais linguagens de programação foram mais citadas durante

o passar dos anos. A Figura 20 mostra um gráfico comparando o crescimento do Python com outras linguagens de programação populares atualmente.

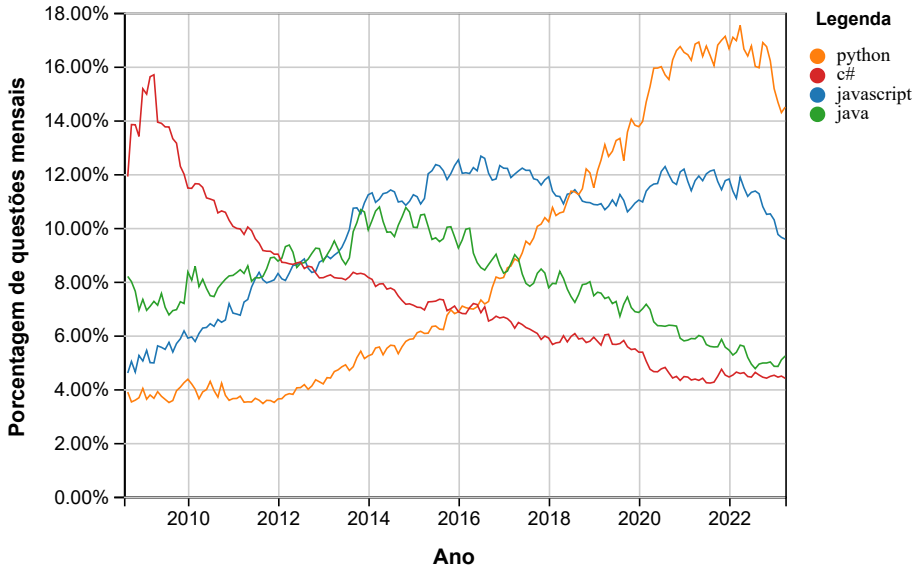


**Figura 20 – Linguagens de programação mais populares no Stack Overflow**
**Fonte: StackOverflow (2023).**

### 3.1.4 `Scikit-image`

`Scikit-image` é uma biblioteca de processamento de imagens baseada em Python e possui diversas funções e algoritmos prontos para manipulação, análise e processamento de imagens. Ela foi desenvolvida utilizando algumas outras bibliotecas do Python, como por exemplo a `Numpy`, que permite realizar operações em imagens no fomato de *numpy-arrays* (GOUILLART, S.I.).

Essa biblioteca possui diversos módulos que fornecem uma grande variedade de funcionalidades, como por exemplo registro de imagem (*Image Registration*), transformações de imagens (*Transform*), registro de características (*Features*), entre outros (TEAM, 2023).

O repositório online do Scikit-Image possui uma interface de fácil utilização, com uma documentação detalhada e exemplos para cada função da biblioteca, o que a torna acessível para desenvolvedores de todos os níveis. Além disso, ela possui uma comunidade ativa, responsável por manter a biblioteca atualizada e contribuir para o seu aperfeiçoamento contínuo.

### 3.1.5 `OpenCV`

O `OpenCV` é uma biblioteca *open-source* de visão computacional e *machine learning*. Ela foi inicialmente desenvolvida pela Intel com o objetivo de proporcionar uma estrutura comum para aplicações de visão computacional em produtos comerciais (OPENCV, 2023). Ela fornece uma grande variedade de funções e algoritmos que permitem a manipulação e o processamento de imagens e vídeos.

Essa biblioteca é escrita em C++, entretanto possui interfaces disponíveis para Python e outras linguagens de programação. Ela dispõe de diversas funções e algoritmos que permitem a manipulação e o processamento de imagens e vídeos. O `OpenCV` é conhecido por sua eficiência e desempenho, sendo utilizado em muitas aplicações comerciais e de pesquisa.

Algumas das principais funcionalidades para visão computacional dessa biblioteca incluem a detecção e rastreamento de objetos, reconhecimento facial, calibração de câmera, estimação de movimento e segmentação de imagens.

## 3.2 Métodos

Nessa seção serão apresentados os métodos adotados para a reprodução de três algoritmos GME para aplicação em imagens reconstruídas de ultrassom. As subseções a seguir apresentam exemplos dos códigos desenvolvidos e explicam o seu princípio de funcionamento, destacando as modificações que foram necessárias para torná-los adequados para o processamento das imagens de ultrassom. Ainda, para verificar o funcionamento dos algoritmos, foram realizados testes com imagens naturais para determinar deslocamento e rotação entre imagens.

### 3.2.1 Algoritmo *Wavelets*

O primeiro algoritmo foi desenvolvido baseado no trabalho de Zhou *et al.* (2022). O método proposto combina a abordagem de diferença de fases com a multiplicação de matriz *Discrete Fourier Transform* (DFT) para calcular com precisão sub-pixel o movimento entre imagens. Além disso, introduz um passo inicial no processamento da sequência de imagens para melhorar a eficiência do algoritmo. A Figura 21 mostra um fluxograma exemplificando a sequência lógica desse método.

O Algoritmo *Wavelets* pode ser dividido nas seguintes etapas: inicialização dos dados, reconstrução de imagens, extração de sub-imagens, estimação de deslocamento e por fim apresentação dos resultados. Essa técnica possui bons resultados para imagens naturais, portanto, para validação do seu funcionamento, testes de controle foram realizados na imagem natural `camera()`, disponível na biblioteca `scikit-image`. Por meio do módulo `transform`, é

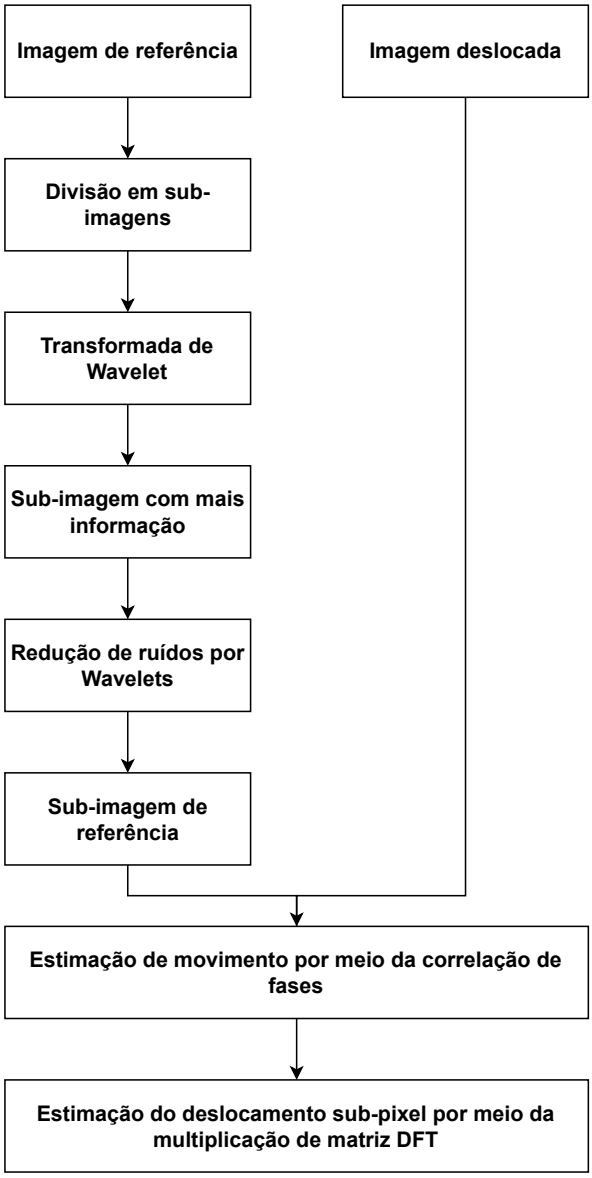


**Figura 21 – Representação do funcionamento do Algoritmo *Wavelets***
**Fonte: Adaptado de Zhou *et al.* (2022).**

possível aplicar valores conhecidos para translação ou rotação a essa imagem, que serão comparados e verificados com os resultados retornados pelo algoritmo.

Para a inicialização dos dados provenientes de simulações e de inspeções reais por ultrassom, o *framework* AUSPEX disponibiliza funções para leitura desses arquivos e padronização de seus dados. Ainda, a fim de reduzir a quantidade de dados que serão processados pelo algoritmo, uma região de interesse (*Region of Interest* (ROI), do inglês *Region of Interest*) na imagem é especificada.

A reconstrução das imagens da ROI é feita utilizando o método TFM, o qual também está presente no *framework*. A função disponibilizada utiliza como parâmetros a região delimitada do conjunto de dados e o número da sequência de disparos do transdutor, chamado de `sel_shot`. Para determinar o deslocamento ou rotação do transdutor, serão utilizados dois

TFMs de diferentes disparos para uma mesma ROI. O passo seguinte na reconstrução das imagens é a aplicação das técnicas de pós-processamento `normalize()` e `envelope()`. A Listagem 1 exemplifica o fragmento do algoritmo responsável pela inicialização dos dados e geração dos TFMs.

```
# --- Dados ---
# Carrega os dados da inspecao do arquivo de simula  o do CIVA.
data = file_civa.read('')

# --- ROI ---
height = a
width = b

corner_roi = np.array([x, 0.0, y])[np.newaxis, :]
roi = data_types.ImagingROI(corner_roi, height=height, width=width)
data.surf = Surface(data, surf_type=SurfaceType.CIRCLE_NEWTON)
data.surf.fit()

# TFMs
chave1 = tfm.tfm_kernel(data, roi=roi, sel_shot=0, c=data.specimen_params.cl)
chave2 = tfm.tfm_kernel(data, roi=roi, sel_shot=1, c=data.specimen_params.cl)

img1 = data.imaging_results[chave1].image
img2 = data.imaging_results[chave2].image

tfm1 = (abs(img1))
tfm1 /= tfm1.max()
tfm2 = (abs(img2))
tfm2 /= tfm2.max()

# Pos-Processamento
image1 = normalize(envelope(tfm1, -2))
image2 = normalize(envelope(tfm2, -2))
```

**Listagem 1 – Inicialização de dados e reconstrução das imagens**
**Fonte: Autoria própria (2023).**

Além disso, em ensaios com rotação entre imagens, é necessário a realização de um passo adicional. Para que o algoritmo seja capaz de estimar rotação em uma sequência de imagens, é preciso a aplicação da Transformada Log-Polar. Isso é feito utilizando a função `warp_polar()` da biblioteca *scikit-image*. A Listagem 2 mostra esse fragmento.

Para a etapa de extração das sub-imagens, é necessário que primeiro as suas dimensões sejam escolhidas. Ao optar por sub-imagens menores, o processamento dos dados será mais rápido, contudo haverá uma redução na quantidade de informações disponíveis em cada fragmento, o que pode comprometer a precisão do algoritmo. Portanto, é necessário encontrar as dimensões ideais para as sub-imagens, levando em consideração a necessidade de que a

```
# Transformada Log-Polar
radius = r
image1 = warp_polar(image1, radius=radius, scaling='log')
image2 = warp_polar(image2, radius=radius, scaling='log')
```

**Listagem 2 – Transformada Log-Polar das imagens**
**Fonte: Autoria própria (2023).**

imagem original seja dividida em fragmentos de medidas semelhantes. No algoritmo desenvolvido, a definição dessas dimensões foi feita de forma empírica para cada um dos casos.

Com as dimensões definidas, a divisão da imagem original foi feita utilizando laços `for` e a função `append()` da biblioteca *Numpy*. A Listagem 3 exemplifica a lógica desenvolvida para isso.

```
# Tamanho das sub-imagens
varx = x
vary = y

# Criacao das sub-imagens
x = int(image1.shape[0]/varx)
y = int(image1.shape[1]/vary)

# Listas que recebem as sub-imagens
sub_image = []
sub_rotated = []
for i in range(x * y):
    sub_image.append(np.zeros((varx, vary)))
    sub_rotated.append(np.zeros((varx, vary)))

k = 0
for i in range(x):
    for j in range(y):
        sub_image[k] = image1[((varx*i)+0):((varx*i)+varx), ((vary*j)+0):((vary*j)+vary)]
        sub_rotated[k] = image2[((varx*i)+0):((varx*i)+varx), ((vary*j)+0):((vary*j)+vary)]
        k = k+1
```

**Listagem 3 – Divisão da imagem original em sub-imagens**
**Fonte: Autoria própria (2023).**

O passo seguinte nessa etapa é a aplicação da Transformada de Wavelet para determinar qual das sub-imagens possui a maior quantidade de informação. Esse processo é realizado apenas nos fragmentos da imagem de referência e para isso foi utilizada a função `dwt2()` da biblioteca *PyWavelets*, que é capaz de extrair os coeficientes LL, LH, HL e HH de cada fragmento. Um dos parâmetros dessa função é o tipo de wavelet aplicada. Cada família de wavelet possui um formato de onda singular e, com isso, gera resultados diferentes. Por meio de testes, foi constatado que a `bior6.8` apresenta melhores resultados para as aplicações com imagens reconstruídas de ultrassom.

Por meio da Equação 10, o coeficiente Wavelet de cada sub-imagem é determinado e registrado em uma lista. Transformando essa lista em um *array numpy* com a função `np.array()`, é possível determinar o maior valor do conjunto utilizando a função `np.argmax()`. O fragmento referente à esse valor será utilizado como referência para determinar o deslocamento na sequência de imagens. O passo final dessa etapa é a aplicação de um filtro para redução de ruídos nos fragmentos das imagens. Para o Algoritmo *Wavelets* foi utilizada a função `denoise_wavelet()` do *scikit-image*. A Listagem 4 demonstra o processo adotado para a extração das sub-imagens.

A estimação do deslocamento entre as imagens é feita por meio do método de correlação de fases. A precisão sub-pixel desejada é obtida utilizando a multiplicação de matriz DFT, que irá aplicar um fator de sobreamostragem. Todo esse processo pode ser feito a par-

```python
# Aplicando Transformada de Wavelet nas sub-imagens
coeffs = []  # Lista que recebe os coeficientes LL, LH, HL e HH
for i in range(x * y):
    coeffs.append(pywt.dwt2(sub_image[i], 'bior6.8'))

# Determinando sub-imagem com max(E)
E = []
for i in range(x * y):
    E.append(coeffs[i][0][0]**2 + coeffs[i][1][0]**2 + coeffs[i][1][1]**2)

E = np.array(E)
trp = np.transpose(E, (0, 2, 1))
max = int(np.argmax(trp)/(E.shape[1]*E.shape[2]))

# Sub-imagem da imagem referencia
ref0 = sub_image[max]
# Sub-imagem da imagem rotacionada
ref1 = sub_rotated[max]

# Denoise das sub-imagens
ref0 = denoise_wavelet(ref0, rescale_sigma=True)
ref1 = denoise_wavelet(ref1, rescale_sigma=True)
```

**Listagem 4 – Extração de sub-imagem de referência**
**Fonte: Autoria própria (2023).**

tir da função `phase_cross_correlation`, que possui como um de seus parâmetros o `upsample_factor`. Essa função, terá como resposta os valores de translação no eixo $x$ e no eixo $y$, medidos em pixels.

Por conta da Transformada Log-Polar, os ensaios para rotação entre imagens não necessitam nenhum cálculo extra, já que nesse domínio a translação, em pixel, no eixo $y$ já representa a rotação, em graus, entre as imagens originais. Porém, para determinar a translação em milímetros entre as imagens originais, é preciso multiplicar o resultado obtido por um fator $k$, que depende da quantidade de pixels e de seu tamanho. A Listagem 5 exemplifica a parte final do Algoritmo *Wavelets*.

```python
# PC para determinar a translacao da imagem
shifts, error, phasediff = phase_cross_correlation(ref0, ref1, upsample_factor)
k = -width/image1.shape[1]
print("Recovered value for translation in mm: "
  f"{shifts[1]*k}")


#PC para determinar a rotacao da imagem
shifts, error, phasediff = phase_cross_correlation(ref0, ref1, upsample_factor)
print("Recovered value for counterclockwise rotation in degrees: "
  f"{shifts[0]}")
```

**Listagem 5 – Determinação do deslocamento entre imagens com PC**
**Fonte: Autoria própria (2023).**

### 3.2.2 Algoritmo *Features*

O segundo algoritmo desenvolvido é uma reprodução do algoritmo ORB, que utiliza características das imagens para determinar deslocamento e translação. No método proposto,

é feita a combinação do detector FAST com o descritor BRIEF, além de possuir uma etapa para determinar a orientação de cada ponto de interesse.

Baseado em seu princípio de funcionamento, o Algoritmo *Features* é dividido nas seguintes etapas: inicialização dos dados, detecção e descrição de *features*, correspondência de *features* e por fim a extração dos resultados. Assim como no Algoritmo *Wavelets*, o seu desempenho foi avaliado por meio da imagem natural `camera()` transladada e rotacionada.

A inicialização dos dados obtidos em simulações ou inspeções reais, é realizada da mesma forma que no Algoritmo *Wavelets*, assim como a determinação da ROI e a geração dos respectivos TFMs, mostrados na Listagem 1. A diferença para esse algoritmo é a necessidade de transformar essas imagens para a escala de cinzas, adequando seus formatos para as etapas seguintes do algoritmo.

A etapa de detecção e descrição de *features*, começa com a inicialização do ORB, por meio da função `ORB_create()`, disponibilizada na biblioteca `OpenCV`. Essa função possui diversos parâmetros que podem ser especificados de acordo com a aplicação do algoritmo. Para os testes com imagens reconstruídas de ultrassom, esses parâmetros foram determinados de forma empírica. No entanto, para imagens naturais, os valores padrão dessa função apresentam boa precisão nos resultados.

Com o detector e o descritor inicializados, é possível aplicá-los na sequencia de imagens com a função `detectAndCompute()`. A Listagem 6 mostra a implementação dessas funções.

```
# Imagens em escala de cinzas
img1 = img_as_ubyte(img1)
img2 = img_as_ubyte(img2)

# Inicializar o detector e o descritor ORB
orb = cv2.ORB_create(nfeatures, scaleFactor, nlevels, edgeThreshold, WTA_K, patchSize,
    ↪ fastThreshold)

# Encontrar os keypoints e os descritores nas duas imagens
kp1, des1 = orb.detectAndCompute(img1, None)
kp2, des2 = orb.detectAndCompute(img2, None)
```

**Listagem 6 – Configuração do detector e descritor ORB**
**Fonte: Autoria própria (2023).**

Para a etapa de correspondência de *features*, é feita a inicialização do respectivo algoritmo. Existem diferentes métodos para isso, e seu uso varia de acordo com a aplicação e processamento requeridos. Para o Algoritmo *Features*, foi utilizada a técnica de correspondência por Força Bruta (*Brute-Force Matching*) com a medida de similaridade feita por meio da distância *Hamming*. Essa inicialização é feita utilizando o `BFMatcher.create()` e sua aplicação é realizada nas características descritas com a função `match()`.

A quantidade de *features* correspondidos é determinada pelo parâmetro `nfeatures` do `ORB_create`. A fim de considerar as melhores correspondências e melhorar a precisão do algoritmo, a função `sorted` é utilizada para ordenar as amostas de acordo com a distância

entre os pontos-chave. Em seguida é feita a seleção das $N$ melhores correspondências. A Listagem 7 exemplifica a lógica aplicada.

```
# Inicializar o algoritmo de correspondencia
bf = cv2.BFMatcher.create(cv2.NORM_HAMMING2, crossCheck=True)

# Encontrar as correspondencias entre os descritores das duas imagens
matches = bf.match(des1, des2)

# Ordenar as correspondencias pela distancia dos descritores
matches = sorted(matches, key=lambda x: x.distance)

# Selecionar as N melhores correspondncias
N = N
good_matches = matches[:N]
```

**Listagem 7 – Configuração do algoritmo de correspondência**
**Fonte: Autoria própria (2023).**

Por fim, para a última etapa do algoritmo, é realizada a extração das coordenadas dos $N$ melhores pontos-chave correspondidos. Com esses valores, a matriz de transformação das imagens pode ser determinada por meio da função `findHomography()`. A Matriz de Transformação Homográfica é uma matriz 3x3 que descreve a transformação projetiva entre dois planos e pode ser expressa conforme a Equação 20 (ALIBAY *et al.*, 2014).

$$H = \begin{pmatrix} cos(\theta) & sen(\theta) & \rho_x \\ -sen(\theta) & cos(\theta) & \rho_y \\ 0 & 0 & 1 \end{pmatrix}. \tag{20}$$

A rotação entre as imagens é dada em graus por $\theta$, já a translação $x$ e $y$ é dada em pixels por $\rho_x$ e $\rho_y$, respectivamente. Para obter o seu valor em milímetros é necessário a multiplicação de um fator $k$, como realizado no primeiro algoritmo apresentado. A Listagem 8 mostra o fragmento final do código desenvolvido.

### 3.2.3 Algoritmo *Keypoints*

O terceiro e último algoritmo desenvolvido é baseado no algoritmo BRISK, que similarmente ao ORB, realiza a extração de características nas imagens para determinar deslocamento. Esse algoritmo combina informações de escala e orientação para extrair pontos de interesse robustos e invariantes à transformações geométricas, além de ser adaptável às imagens analisadas por conta do detector AGAST.

Assim como no Algoritmo *Features*, o Algoritmo *Keypoints* pode ser dividido em: inicialização dos dados, detecção e descrição, correspondência e finalmente a extração dos resultados. A lógica aplicada e as funções utilizadas são as mesmas que no Algoritmo *Features* para todos os passos, exceto na etapa de detecção e descrição de características. Nela, a iniciali-

```python
# Extrair as coordenadas dos keypoints correspondentes nas duas imagens
pts1 = np.float32([kp1[m.queryIdx].pt for m in good_matches]).reshape(-1, 1, 2)
pts2 = np.float32([kp2[m.trainIdx].pt for m in good_matches]).reshape(-1, 1, 2)

# Calcular a matriz de transformacao que leva de pts1 para pts2
H, _ = cv2.findHomography(pts1, pts2, cv2.RANSAC, 4.0)

# Calcular o vetor de translacao a partir da matriz de transformacao
dx = M[0, 2]

k = -width/image1.shape[1]

# Resultado para translacao
print("Recovered value for translation in mm: "
          f"{dx*k}")

# Resultado para rotacao
theta = -np.arctan2(M[0, 1], M[0, 0]) * 180 / np.pi
print("Recovered value for counterclockwise rotation in degrees: "
          f"{theta}")
```

**Listagem 8 – Determinação do deslocamento entre imagens com matriz H**
**Fonte: Autoria própria (2023).**

zação do detector e descritor é feita para o método BRISK com a função `BRISK_create()`, também da biblioteca `OpenCV`.

Essa função possui menos parâmetros que a do ORB, o que se deve ao detector que se adapta a imagem. Os parâmetros para imagens reconstruídas de ultrassom foram determinados empiricamente e, igualmente ao Algoritmo *Features*, os valores padronizados possuem boa resposta para imagens naturais. A Listagem 9 mostra essa parte do algoritmo.

```python
# Inicializar o detector e o descritor BRISK
brisk = cv2.BRISK_create(thresh, octaves, patternScale)

# Encontrar os keypoints e os descritores nas duas imagens
kp1, des1 = brisk.detectAndCompute(img1, None)
kp2, des2 = brisk.detectAndCompute(img2, None)
```

**Listagem 9 – Configuração do detector e descritor BRISK**
**Fonte: Autoria própria (2023).**

## 3.3 Considerações Finais

O desenvolvimento dos algoritmos apresentados neste trabalho foi feito por meio da linguagem de programação Python, com base em ferramentas e funções prontas, disponíveis nas bibliotecas de processamento de sinais `Scikit-image` e `OpenCV`. Além disso, para realizar a leitura, processamento e exibição dos dados provenientes de ensaios por ultrassom, as ferramentas do *Framework* AUSPEX foram utilizadas. Ainda, os três algoritmos aqui mostrados foram baseados nos métodos de correlação de fases e registro de características, técnicas que já são consolidadas para estimação de movimento entre imagens naturais.

# 4 RESULTADOS

Esse capítulo tem como objetivo expor os resultados obtidos a partir da implementação dos Algoritmos *Wavelets*, *Features* e *Keypoints* em imagens reconstruídas de ultrassom. Os dados de captura FMC foram obtidos por meio de ensaios em dois corpos de prova diferentes, nos quais foram aplicados dois tipos de deslocamento distintos no transdutor, foram eles rotação e translação. O transdutor utilizado nos ensaios é do tipo *array-linear*, com 64 elementos, frequência central de 5 MHz, *pitch* de 0,3 mm e frequência de amostragem de 170 MHz.

Durante os ensaios realizados, para avaliar a rotação do transdutor ultrassônico, utilizou-se uma faixa de rotação que variava de 0 a 20°, com incrementos de 1,5°, considerando o eixo $z$ fixo. Já no caso dos ensaios para avaliar a translação, foi adotada uma faixa de deslocamento no eixo x que variou de 0 a 16,8 mm, com incrementos de 1,12 mm. Todos os dados desses ensaios foram obtidos por meio de simulações no *software* CIVA.

## 4.1 Imagem Natural

Para fins de validação do funcionamento dos algoritmos reproduzidos, testes de controle foram realizados considerando o intervalo de rotação especificado e uma translação no eixo $x$ de 0 a 120 pixels, com incrementos de 20 pixels.

Com dimensões de 512 x 512 pixels, a imagem natural `camera()` foi escolhida e a Figura 22 exemplifica as imagens original, rotacionada e transladada.

### 4.1.1 Algoritmo *Wavelets*

Devido à própria natureza da função `warp_polar()`, que converte rotação em deslocamento entre as imagens ao longo do eixo X, é essencial que, em casos de grandes rotações entre as imagens, as sub-imagens tenham uma maior quantidade de pixels nesse eixo. Portanto, para cobrir toda a faixa de rotação possível entre as imagens naturais, fragmentos de 360 x 72 pixels foram escolhidos. Já para os testes de translação, sub-imagens de 128 x 128 pixels foram avaliadas como mais adequadas para os deslocamentos aplicados. De acordo com os resultados expostos nas Tabelas 1 e 2, os valores detectados para rotação foram iguais aos valores aplicados nas imagens. Além disso, os valores para translação apresentaram erro máximo de apenas 0,1%.

### 4.1.2 Algoritmo *Features*

Para a aplicação do Algoritmo *Features* em imagens naturais, os seguintes parâmetros padrão da função `ORB_create()` foram utilizados: `nfeatures`=500, `scaleFactor`=1.2,

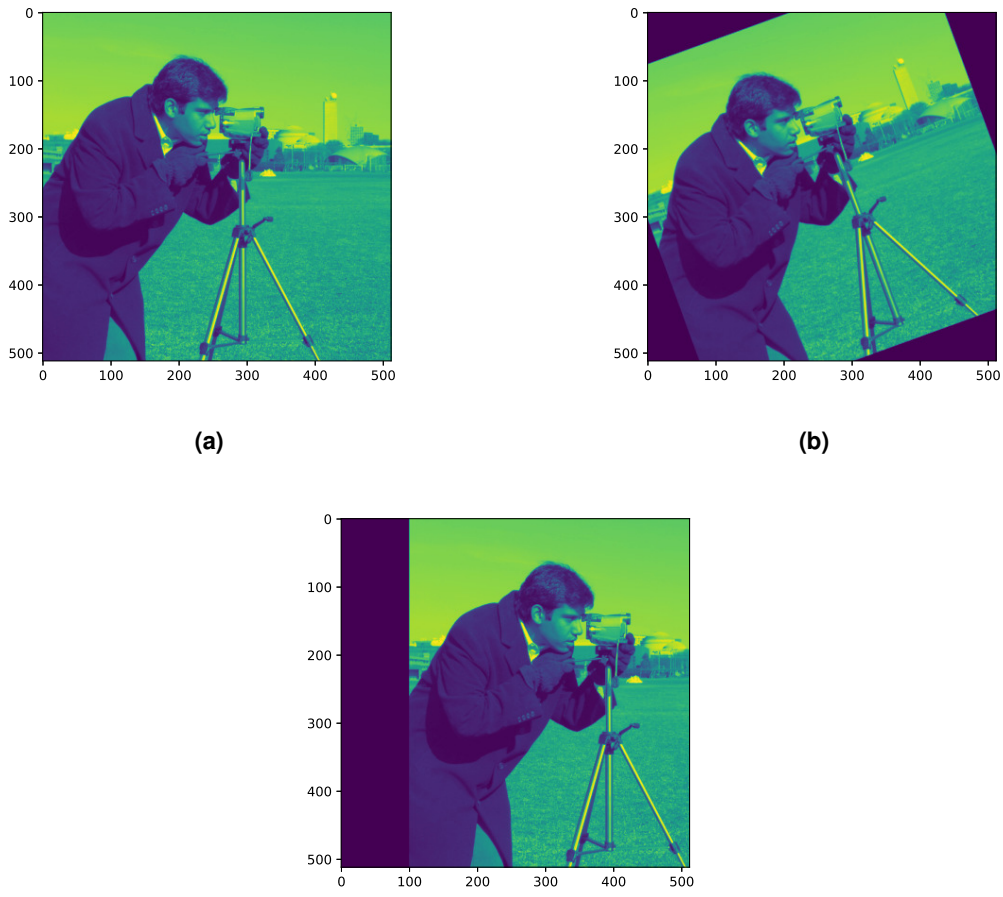


**Figura 22 – Exemplo de deslocamentos aplicados na imagem natural `camera()`. (a) Imagem original. (b) Imagem rotacionada em 20º no sentido anti-horário. (c) Imagem transladada em 100 pixels.**
**Fonte: Adaptado de Guerreiro (2020).**

`nlevels`=8, `edgeThreshold`=31, `WTA_K`=4, `patchSize`=31, `fastThreshold`=20. Considerando as 100 melhores correspondências, as Tabelas 3 e 4 mostram, respectivamente, os resultados obtidos para translação e rotação entre as imagens.

Ao comparar com os resultados obtidos nas Tabelas 1 e 2, nota-se que o Algoritmo *Features* apresenta maior acurácia na detecção de translação, porém se mostra inferior na estimação de rotação, exibindo um erro máximo de 18,6%.

### 4.1.3 Algoritmo *Keypoints*

Os resultados dos testes do Algoritmo *Keypoints* com imagens naturais estão exibidos nas Tabelas 5 e 6, considerando as 200 melhores amostras obtidas. Os parâmetros adotados para `BRISK_create()` foram os seguintes valores padrão: `thresh`=30, `octaves`=3 e `patternScale`=1.0.

**Tabela 1 – Rotação Imagem Natural - Algoritmo *Wavelets***

| Rotação Aplicada | Rotação Detectada |
|---|---|
| 2,5° | 2,5° |
| 5,0° | 5,0° |
| 7,5° | 7,5° |
| 10,0° | 10,5° |
| 12,5° | 12,5° |
| 15,0° | 15,0° |
| 17,5° | 17,5° |
| 20,0° | 20,0° |

**Fonte: Autoria própria (2023).**

**Tabela 2 – Translação Imagem Natural - Algoritmo *Wavelets***

| Translação Aplicada (pixels) | Translação Detectada (pixels) |
|---|---|
| 20,00 | 19,98 |
| 40,00 | 39,98 |
| 60,00 | 59,96 |
| 80,00 | 79,96 |
| 100,00 | 99,98 |
| 120,00 | 119,96 |

**Fonte: Autoria própria (2023).**

Analisando os resultados das Tabelas 5 e 6, é possível concluir que o Algoritmo *Keypoints* possui boa precisão em ambos os tipos de movimentos aplicados, apresentando erro máximo de 0,01% para estimação de translação e 8,0% para rotação. Ainda, em comparação com os outros algoritmos reproduzidos, o Algoritmo *Keypoints* possui melhor desempenho na

**Tabela 3 – Translação Imagem Natural - Algoritmo *Features***

| Translação Aplicada (pixels) | Translação Detectada (pixels) |
|---|---|
| 20,00 | 20,00 |
| 40,00 | 40,00 |
| 60,00 | 60,00 |
| 80,00 | 80,00 |
| 100,00 | 100,00 |
| 120,00 | 119,99 |

**Fonte: Autoria própria (2023).**

**Tabela 4 – Rotação Imagem Natural - Algoritmo *Features***

| Rotação Aplicada | Rotação Detectada |
|---|---|
| 2,5° | 2,58° |
| 5,0° | 4,07° |
| 7,5° | 7,39° |
| 10,0° | 10,13° |
| 12,5° | 12,41° |
| 15,0° | 15,63° |
| 17,5° | 17,59° |
| 20,0° | 20,43° |

**Fonte: Autoria própria (2023).**

**Tabela 5 – Translação Imagem Natural - Algoritmo *Keypoints* - Parâmetros Padrão**

| Translação Aplicada (pixels) | Translação Detectada (pixels) |
|---|---|
| 20,00 | 19,99 |
| 40,00 | 39,99 |
| 60,00 | 59,99 |
| 80,00 | 79,99 |
| 100,00 | 100,01 |
| 120,00 | 120,00 |

**Fonte: Autoria própria (2023).**

**Tabela 6 – Rotação Imagem Natural - Algoritmo *Keypoints* - Parâmetros Padrão**

| Rotação Aplicada | Rotação Detectada |
|---|---|
| 2,5° | 2,51° |
| 5,0° | 4,96° |
| 7,5° | 7,46° |
| 10,0° | 9,95° |
| 12,5° | 12,54° |
| 15,0° | 14,99° |
| 17,5° | 17,53° |
| 20,0° | 19,98° |

**Fonte: Autoria própria (2023).**

detecção de translação quando comparado com o Algoritmo *Wavelets* e maior acurácia na detecção de rotação se comparado com o Algoritmo *Features*.

## 4.2 Corpo de prova 1

O primeiro corpo de prova simulado no CIVA foi um peça de acrílico com 200 mm de comprimento e com base dentada, como mostra a Figura 23. Considerando como ponto de origem da peça as coordenadas (70, 25), a ROI especificada para a reconstrução das imagens de ultrassom foi (-25 mm $\leq$ x $\leq$ 25 mm) e (5 mm $\leq$ y $\leq$ 45 mm).

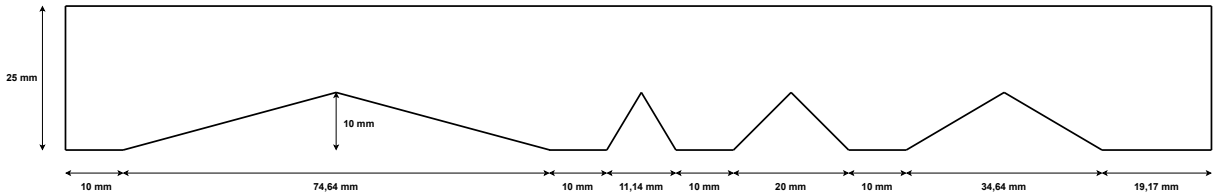


**Figura 23 – Dimensões do corpo de prova 01**
**Fonte: Autoria própria.**

A Figura 24 mostra a comparação do TFM gerado com a região delimitada no corpo de prova simulado.

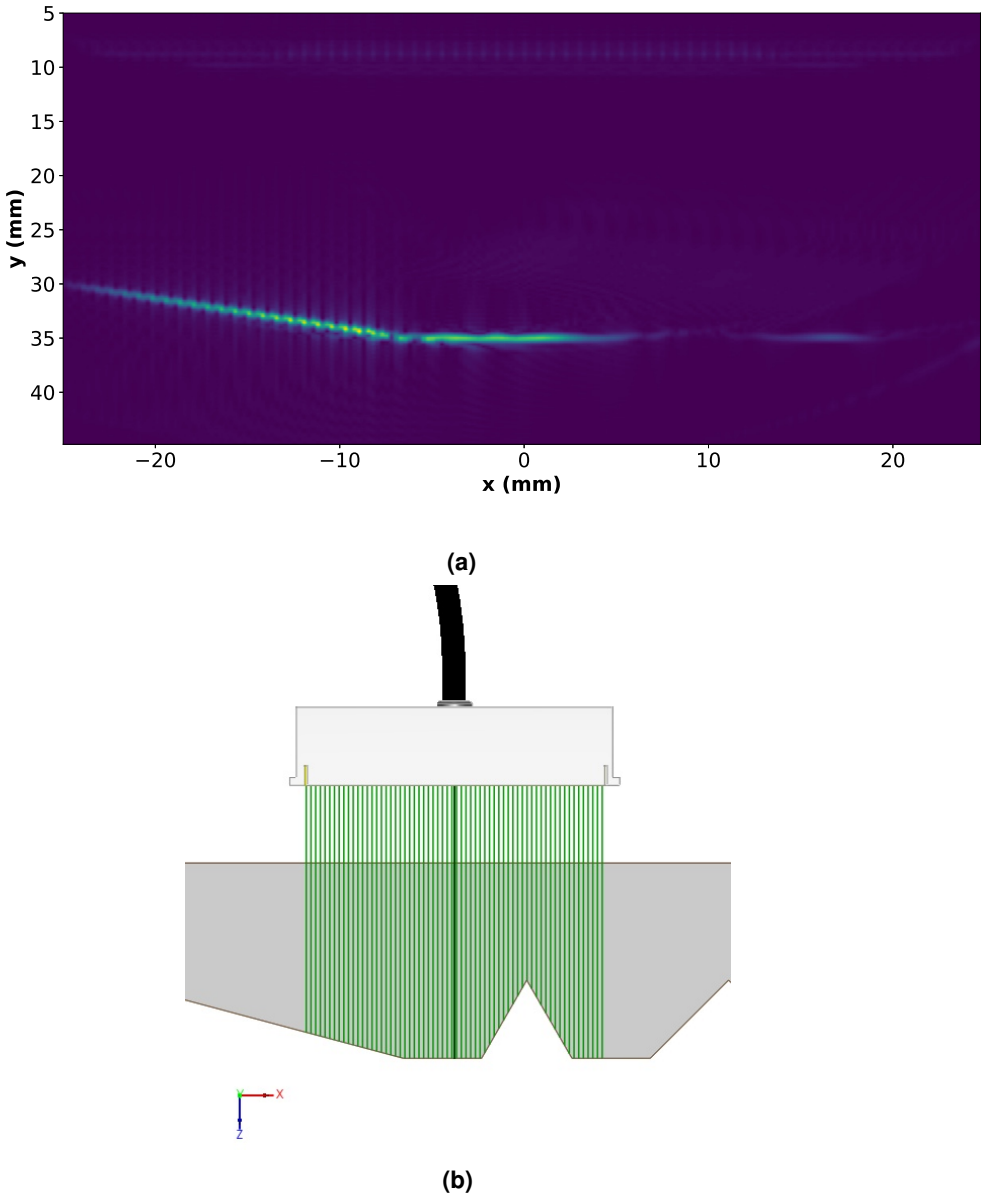


**Figura 24 – Corpo de prova 01. (a) TFM gerado. (b) Simulação CIVA.**
**Fonte: Autoria própria (2023).**

### 4.2.1 Algoritmo *Wavelets*

Nos testes envolvendo translação entre imagens com o corpo de prova 1, foram considerados fragmentos com dimensões de 50 x 200 pixels. A Tabela 7 mostra os resultados obtidos. Nota-se que, com erro máximo inferior a 1,0%, o algoritmo reproduzido apresenta boa resposta até deslocamentos de 11,2 mm.

Já para os testes de rotação, levando em consideração a ROI especificada anteriormente, concluiu-se que sub-imagens de dimensões 90 x 180 píxels produzem melhores resultados para objetos com formato linear. Os resultados obtidos são apresentados na Tabela 8. Nota-se que o algoritmo demonstra uma taxa de erro inferior a 10% para ângulos de rotação variando de 7,5° a 17,5°.

**Tabela 7 – Translação Corpo de prova 01 - Algoritmo *Wavelets***

| Translação Aplicada (mm) | Translação Detectada (mm) |
|---|---|
| 1,12 | 1,11 |
| 2,24 | 2,24 |
| 3,36 | 3,35 |
| 4,48 | 4,48 |
| 5,60 | 5,59 |
| 6,72 | 6,72 |
| 7,84 | 7,82 |
| 8,96 | 8,96 |
| 10,08 | 10,06 |
| 11,20 | 11,20 |
| 12,32 | 11,67 |
| 13,44 | 11,86 |
| 14,56 | 10,72 |

**Fonte: Autoria própria (2023).**

**Tabela 8 – Rotação Corpo de prova 01 - Algoritmo *Wavelets***

| Rotação Aplicada | Rotação Detectada |
|---|---|
| 2,5° | 3,42° |
| 5,0° | 5,54° |
| 7,5° | 6,82° |
| 10,0° | 9,94° |
| 12,5° | 11,98° |
| 15,0° | 13,98° |
| 17,5° | 17,40° |
| 20,0° | 11,10° |

**Fonte: Autoria própria (2023).**

### 4.2.2 Algoritmo *Features*

Para os testes de translação do Algoritmo *Features*, as seguintes configurações de parâmetros foram consideradas:

1. `nfeatures`=500, `scaleFactor`=1.2, `nlevels`=8, `edgeThreshold`=31, `WTA_K`=4, `patchSize`=31, `fastThreshold`=20, `N`=100 ;
2. `nfeatures`=500, `scaleFactor`=1.9, `nlevels`=10, `edgeThreshold`=40, `WTA_K`=4, `patchSize`=40, `fastThreshold`=30, `N`=20 .

A Configuração 1 adota os valores padrão da função `ORB_create` e, por apresentarem bons resultados com imagens naturais, foi utilizada nos testes com imagens reconstruídas para fins de comparação. Já a Configuração 2, dentro de várias outras configurações testadas, foi o melhor conjunto identificado.

A Tabela 9 exibe os resultados desses testes para ambas as configurações de parâmetros.

Considerando os mesmos parâmetros descritos anteriormente, os testes para rotação nos corpos de prova 01 estão apresentados na Tabela 10.

**Tabela 9 – Translação Corpo de prova 01 - Algoritmo *Features***

| Deslocamento Aplicado (milímetros) | Deslocamento Detectado (milímetros) 1 | 2 |
|---|---|---|
| **1,12** | 79,43 | 18,13 |
| **2,24** | 142,97 | 108,51 |
| **3,36** | 14,24 | 125,89 |
| **4,48** | 11,56 | 29,39 |
| **5,60** | 15,44 | 11,22 |
| **6,72** | 313,65 | 4,52 |

**Fonte: Autoria própria (2023).**

**Tabela 10 – Rotação Corpo de prova 01 - Algoritmo *Features***

| Rotação Aplicada | Rotação Detectada 1 | 2 |
|---|---|---|
| $2{,}50°$ | $72{,}18°$ | $68{,}13°$ |
| $5{,}00°$ | $75{,}84°$ | $86{,}93°$ |
| $7{,}50°$ | $84{,}60°$ | $76{,}87°$ |
| $10{,}00°$ | $104{,}40°$ | $81{,}30°$ |
| $12{,}50°$ | $75{,}35°$ | $65{,}64°$ |
| $15{,}00°$ | $80{,}30°$ | $74{,}37°$ |
| $17{,}50°$ | $75{,}84°$ | $87{,}69°$ |
| $20{,}00°$ | $79{,}17°$ | $88{,}45°$ |

**Fonte: Autoria própria (2023).**

Em relação aos resultados exibidos, observa-se que o Algoritmo *Features* não demonstrou bom funcionamento quando aplicado nas imagens reconstruídas de ultrassom do corpo de prova 01, apresentando grandes erros nas estimações de translação e rotação.

### 4.2.3 Algoritmo *Keypoints*

Para os testes com imagens reconstruídas de ultrassom do corpo de prova 01, as seguintes configurações de parâmetros foram consideradas:

1. `thresh`=30, `octaves`=3, `patternScale`=1.0, `N`=200 ;
2. `thresh`=30, `octaves`=8, `patternScale`=2.0, `N`=50 .

Os resultados dos testes do Algoritmo *Keypoints* para translação entre imagens estão apresentados na Tabelas11.

Já os resultados dos testes de rotação estão exibidos na Tabela 12, considerando as mesmas configurações de parâmetros descritas anteriormente.

Assim como o Algoritmo *Features*, o Algoritmo *Keypoints* não exibe bom funcionamento para estimar translação e rotação entre imagens reconstruídas de ultrassom do corpo de prova 01. Em todos os testes o algoritmos apresentou grandes erros e não mostrou nenhuma melhora significativa com a mudança de seus parâmetros.

**Tabela 11 – Translação Corpo de prova 01 - Algoritmo *Keypoints***

| Deslocamento Aplicado (milímetros) | Deslocamento Detectado (milímetros) 1 | 2 |
|---|---|---|
| **1,12** | 14,19 | 22,89 |
| **2,24** | 12,10 | 25,84 |
| **3,36** | 9,11 | 12,07 |
| **4,48** | 14,36 | 5,45 |
| **5,60** | 13,72 | 1,14 |
| **6,72** | 12,93 | 55,78 |

**Fonte: Autoria própria (2023).**

**Tabela 12 – Rotação Corpo de prova 01 - Algoritmo *Keypoints***

| Rotação Aplicada | Rotação Detectada 1 | 2 |
|---|---|---|
| $2{,}50°$ | $66{,}48°$ | $68{,}83°$ |
| $5{,}00°$ | $70{,}22°$ | $65{,}36°$ |
| $7{,}50°$ | $15{,}95°$ | $150{,}70°$ |
| $10{,}00°$ | $90{,}55°$ | $70{,}57°$ |
| $12{,}50°$ | $80{,}52°$ | $70{,}32°$ |
| $15{,}00°$ | $78{,}48°$ | $74{,}99°$ |
| $17{,}50°$ | $85{,}54°$ | $155{,}84°$ |
| $20{,}00°$ | $0{,}21°$ | $62{,}71°$ |

**Fonte: Autoria própria (2023).**

## 4.3 Corpo de prova 2

O segundo corpo de prova utilizado nos ensaios desse trabalho foi uma peça de acrílico dentada semi-circular, com raio externo de 70,75 mm, como mostra a Figura 25. Para essa peça, o ponto de origem considerado foi (0, 10, 10) e a ROI escolhida para reconstrução de suas imagens foi (-30 mm $\leq$ x $\leq$ 35 mm) e (0 mm $\leq$ y $\leq$ 50 mm).

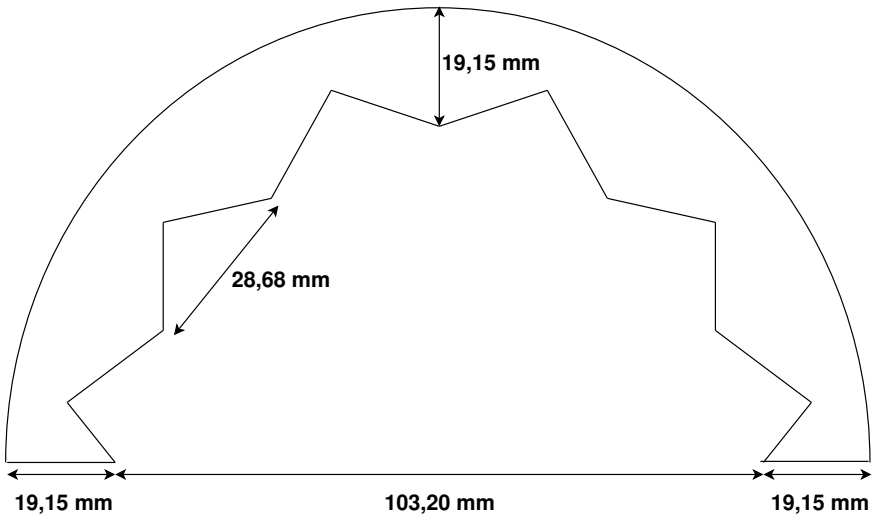


**Figura 25 – Dimensões do corpo de prova 02**
**Fonte: Autoria própria.**

Na Figura 26 é feita a comparação entre o TFM gerado com a ROI especificada na peça simulada.

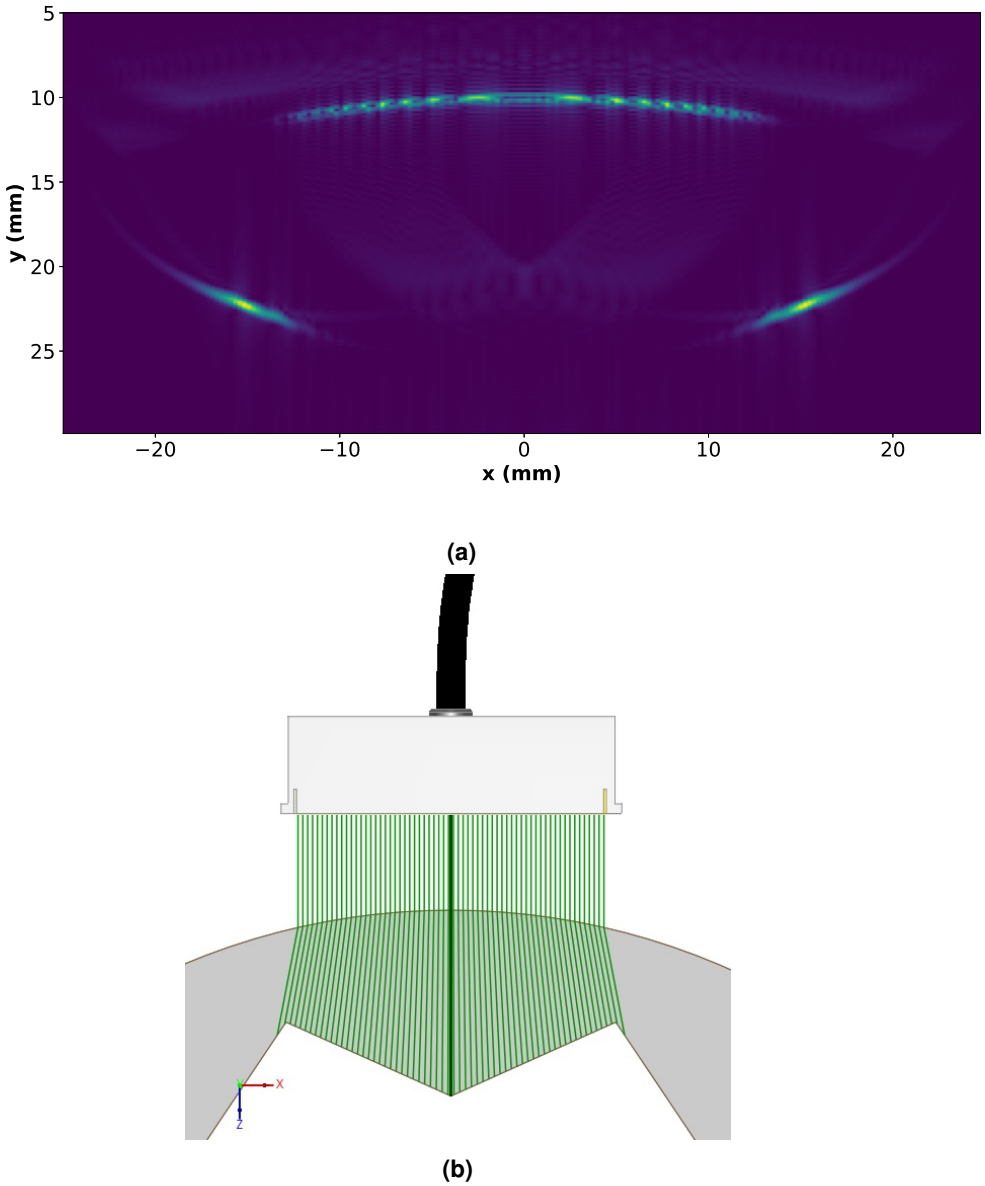


**Figura 26 – Corpo de prova 02. (a) TFM gerado. (b) Simulação CIVA.**
**Fonte: Autoria própria (2023).**

### 4.3.1 Algoritmo *Wavelets*

Para o corpo de prova 2, os testes para determinar rotação foram conduzidos utilizando sub-imagens com dimensões de 360 x 72 pixels. Os resultados obtidos estão apresentados na Tabela 13. Já para os testes de translação com esse corpo de prova, foram utilizados fragmentos de imagens com dimensões de 50 x 200 pixels. A Tabela 14 exibe os resultados obtidos, onde

é possível verificar um bom desempenho do Algoritmo *Wavelets* em todo o intervalo de teste, apresentando erro máximo de 3,3 %.

**Tabela 13 – Translação Corpo de prova 02 - Algoritmo *Wavelets***

| Translação Aplicada (mm) | Translação Detectada (mm) |
|---|---|
| 1,12 | 1,17 |
| 2,24 | 2,37 |
| 3,36 | 3,45 |
| 4,48 | 4,56 |
| 5,60 | 5,74 |
| 6,72 | 6,75 |
| 7,84 | 7,89 |
| 8,96 | 8,96 |
| 10,08 | 10,07 |
| 11,20 | 11,19 |
| 12,32 | 11,92 |
| 13,44 | 13,37 |
| 14,56 | 14,47 |
| 15,68 | 15,63 |
| 16,80 | 16,63 |

**Fonte: Autoria própria (2023).**

**Tabela 14 – Rotação Corpo de prova 02 - Algoritmo *Wavelets***

| Rotação Aplicada | Rotação Detectada |
|---|---|
| 2,5° | 2,30° |
| 5,0° | 4,32° |
| 7,5° | 6,44° |
| 10,0° | 9,14° |
| 12,5° | −8,26° |
| 15,0° | −7,90° |

**Fonte: Autoria própria (2023).**

### 4.3.2 Algoritmo *Features*

Para os testes de translação e rotação entre imagens reconstruídas de ultrassom do corpo de prova 02, foram considerados os mesmos parâmetros apresentados anteriormente. Os resultados obtidos pelo Algoritmo *Features* para translação e rotação estão exibidos, respectivamente, nas Tabelas 15 e 16, incluindo as duas configurações de parâmetros.

Assim como os resultados anteriores do Algoritmo *Features*, para os testes com o corpo de prova 02 o algoritmo baseado na técnica ORB não apresenta bons resultados em aplicações com imagens reconstruídas de ultrassom, apresentando grandes erros nos testes e até mesmo falhas na detecção de deslocamentos, como exibido na Tabela 16.

**Tabela 15 – Translação Corpo de prova 02 - Algoritmo *Features***

| Deslocamento Aplicado (milímetros) | Deslocamento Detectado (milímetros) 1 | 2 |
|---|---|---|
| **1,12** | 0,61 | 0,42 |
| **2,24** | 0,02 | 71,67 |
| **3,36** | 1,07 | 5,85 |
| **4,48** | 3,16 | 1,68 |
| **5,60** | 28,56 | 23,53 |
| **6,72** | 29,19 | 5,18 |

**Fonte: Autoria própria (2023).**

**Tabela 16 – Rotação Corpo de prova 02 - Algoritmo *Features***

| Rotação Aplicada | Rotação Detectada 1 | 2 |
|---|---|---|
| $2{,}50°$ | $0{,}45°$ | $0{,}03°$ |
| $5{,}00°$ | $6{,}91°$ | $0{,}63°$ |
| $7{,}50°$ | $3{,}42°$ | $121{,}49°$ |
| $10{,}00°$ | $10{,}69°$ | $72{,}89°$ |
| $12{,}50°$ | $167{,}37°$ | $85{,}40°$ |
| $15{,}00°$ | $85{,}14°$ | $82{,}10°$ |
| $17{,}50°$ | $105{,}79°$ | - |
| $20{,}00°$ | $81{,}65°$ | - |

**Fonte: Autoria própria (2023).**

### 4.3.3 Algoritmo *Keypoints*

Para os testes de translação e rotação com o corpo de prova 02, as mesmas configurações de parâmetros do Algoritmo *Keypoints* foram consideradas. Os resultados dos testes estão apresentados nas Tabelas 17 e 18.

**Tabela 17 – Translação Corpo de prova 02 - Algoritmo *Keypoints***

| Deslocamento Aplicado (milímetros) | Deslocamento Detectado (milímetros) 1 | 2 |
|---|---|---|
| **1,12** | 5,57 | 493,55 |
| **2,24** | 0,08 | 84,23 |
| **3,36** | 0,05 | 8,17 |
| **4,48** | 16,88 | 3,12 |
| **5,60** | 7,12 | 13,29 |
| **6,72** | 5,61 | 11,54 |

**Fonte: Autoria própria (2023).**

A partir dos resultados exibidos, nota-se que, assim como o Algoritmo *Features*, o algoritmo reproduzido baseado na técnica BRISK não apresenta bons resultados para imagens reconstruídas de ultrassom.

**Tabela 18 – Rotação Corpo de prova 02 - Algoritmo *Keypoints***

| Rotação Aplicada | Rotação Detectada 1 | Rotação Detectada 2 |
|---|---|---|
| 2,50° | 17,11° | 13,75° |
| 5,00° | 1,08° | 2,44° |
| 7,50° | 109,48° | 56,62° |
| 10,00° | 129,80° | 91,51° |
| 12,50° | 103,63° | 57,08° |
| 15,00° | 140,83° | 82,60° |
| 17,50° | 107,25° | 137,69 |
| 20,00° | 99,91° | 150,06 |

**Fonte: Autoria própria (2023).**

## 4.4 Ensaios com dados reais

Os ensaios com dados reais foram realizados utilizando o equipamento M2M Panther (Eddyfi NDT, Inc.) em conjunto do transdutor ultrassônico da Imasonic (IMASONIC SAS, France). As informações técnicas de sua configuração estão exibidas na Tabela 19.

**Tabela 19 – Parâmetros do Sistema de Aquisição**

| Parâmetro | Valor |
|---|---|
| Frequência de amostragem (MHz) | 125 |
| Número de elementos | 64 |
| *Pitch* (mm) | 0,6 |
| Largura dos elementos (mm) | 0,5 |
| Comprimento dos elementos (mm) | 10 |
| Frequência central (MHz) | 5,2 |
| Largura de banda em -6dB | 71% |

**Fonte: Autoria própria (2023).**

A peça inspecionada nesses ensaios foi a versão real em acrílico do corpo de prova 01, ilustrado na Figura 23, porém, para seu ponto de origem foram estabelecidas as coordenadas (0, 25). Os ensaios foram realizados apenas para avaliar a translação do transdutor e a ROI considerada foi a mesma dos testes da peça simulada. Além disso, a faixa dos deslocamentos aplicados no eixo x foi de 0 a 30 mm, com incrementos de 5 mm.

Assim como no ensaio com simulações, para o Algoritmo *Wavelets*, foram avaliados fragmentos de imagens de 50 x 200 pixels. O Algoritmo *Features* foi testado considerando a seguinte configuração de seus parâmetros: `nfeatures`=500, `scaleFactor`=1.2, `nlevel`=8, `edgeThereshold`=31, `WTA_K`=4, `patchSize`=31, `fastThreshold`=20 e `N`=100. Já para o Algoritmo *Keypoints*, sua configuração de parâmetros considerada foi: `thresh`=30, `octaves`=3, `patternScale`=1.0 e `N`=200. Os resultados dos testes dos três algoritmos estão apresentados na Tabela 20.

Analisando os resultados obtidos na Tabela 20, nota-se que os algoritmos reproduzidos nesse trabalho se comportam de forma similar com dados simulados e com dados reais. Tanto o Algoritmo *Features* quanto o Algoritmo *Keypoints* apresentam grande erro de medição e até

**Tabela 20 – Translação Corpo de prova real**

| Deslocamento Aplicado (milímetros) | Deslocamento Detectado (milímetros) | | |
|---|---|---|---|
| | Algoritmo *Wavelets* | Algoritmo *Features* | Algoritmo *Keypoints* |
| **5,00** | 5,03 | 225,47 | 19,72 |
| **10,00** | 9,92 | - | - |
| **15,00** | 14,86 | 19,71 | |
| **20,00** | 19,64 | 34,87 | - |
| **25,00** | 24,80 | 40,37 | - |
| **30,00** | 20,53 | 33,89 | - |

**Fonte: Autoria própria (2023).**

mesmo falhas na detecção de deslocamentos. Já o Algoritmo *Wavelets*, com erro máximo de $1{,}8\%$, mostra bom desempenho e precisão em translações de até 25 mm, considerando os parâmetros especificados.

## 4.5 Considerações Finais

Os resultados obtidos nos testes de rotação e translação com imagens naturais serviram para validar a reprodução dos três algoritmos e para provar a boa precisão dos métodos escolhidos. Nos testes com imagens reconstruídas de ultrassom obtidas por meio de ensaios simulados, percebeu-se bom desempenho do algoritmo *Wavelets* em determinadas faixas de rotação e translação para ambos os corpos de prova. Já os algoritmos *Features* e *Keypoints*, utilizando os parâmetros especificados, não demonstraram bom funcionamento para esse tipo de imagem, apresentando grandes erros de medição de deslocamento nas duas peças inspecionadas. Por fim, o teste realizado com corpo de prova real, serviu para comprovar a acurácia do algoritmo *Wavelets* na determinação de translação entre imagens de ultrassom, além de confirmar o baixo desempenho dos outros dois algoritmos.

# 5 CONCLUSÃO

Os ensaios não destrutivos por ultrassom representam uma poderosa ferramenta no mundo da inspeção e controle de qualidade. Ao usar ondas sonoras de alta frequência, esses ensaios permitem a detecção de defeitos internos em materiais e estruturas, sem comprometer sua integridade. Com uma ampla gama de aplicações em indústrias que vão desde a aeroespacial até a medicina, os END por ultrassom desempenham um papel fundamental na garantia da segurança e confiabilidade de produtos e estruturas, economizando tempo e recursos, enquanto mantêm altos padrões de qualidade.

No entanto, esta técnica apresenta alguns problemas. Durante as inspeções, o deslocamento do transdutor é determinado por sensores *encoders*, que são dispositivos mecânicos. Como resultado, durante uma varredura, podem ocorrer deslizamentos entre o transdutor e o objeto sob inspeção, o que por sua vez resultaria em imprecisões nas medições realizadas nos ensaios. Além disso, os *encoders* operam por meio de cabos conectados, os quais podem restringir os movimentos do transdutor durante as inspeções.

Com o objetivo de contornar esse problema, este trabalho popôs a utilização de algoritmos GME em sequências de imagens reconstruídas de ultrassom para determinar o deslocamento aplicado no transdutor. Utilizando a linguagem de programação *Python*, foram reproduzidos três algoritmos baseados em técnicas diferentes. Para os testes foram considerados deslocamentos de rotação e translação, aplicados em inspeções de dois corpos de prova distintos.

Comparando os resultados obtidos, observou-se que o Algoritmo *Wavelets*, baseado na técnica de correlação de fases, exibiu grande precisão na determinação de translação nos corpos de prova simulados e no real, apresentando robustez para grandes intervalos de deslocamento. Além disso, o algoritmo também se mostrou promissor nos testes com rotação entre imagens. Já os algoritmos *Features* e *Keypoints*, baseados em registro de características, não apresentaram bom desempenho em nenhum dos testes com imagens reconstruídas de ultrassom. Isso pode ser resultado da parametrização considerada nos algoritmos, como existem diversos parâmetros nas funções desses algoritmos, são necessárias melhores análises para determinar o impacto causado pela alteração de cada um deles, a fim de encontrar os valores mais adequados para esse tipo de aplicação.

Portanto, é possível concluir que a implementação do Algoritmo *Wavelets* para determinação de deslocamento em END por ultrassom é viável. Já para os algoritmos *Features* e *Keypoints*, por conta do tempo para conclusão desta dissertação, recomendam-se melhores avaliações com diferentes parametrizações, incorporando técnicas de redes neurais para determinação das suas melhores configurações.

# REFERÊNCIAS


AKANSU, A. N.; HADDAD, R. A. **Multiresolution signal decomposition: transforms, subbands, and wavelets**. [*S.l.*]: Academic press, 2001.

ALIBAY, M. *et al.* Hybrid visual and inertial ransac for real-time motion estimation. *In*: IEEE. **2014 IEEE International Conference on Image Processing (ICIP)**. [*S.l.*], 2014. p. 179–183.

AUSPEX. **Documentação AUSPEX - Versão 1.4**. 2020. Disponível em: http://auspex.xyz/. Acesso em: 28 mai. 2023.

BABOROVSKY, V.; MARSH, D.; SLATER, E. Schlieren and computer studies of the interaction of ultrasound with defects. **Non-Destructive Testing**, v. 6, n. 4, p. 200–207, 1973. ISSN 0029-1021. Disponível em: https://www.sciencedirect.com/science/article/pii/0029102173900339.

BAI, L.; VELICHKO, A.; DRINKWATER, B. W. Ultrasonic defect characterisation—Use of amplitude, phase, and frequency information. **The Journal of the Acoustical Society of America**, v. 143, n. 1, p. 349–360, 2018.

BARJATYA, A. Block matching algorithms for motion estimation. **IEEE Transactions Evolution Computation**, v. 8, n. 3, p. 225–239, 2004.

BAY, H.; TUYTELAARS, T.; GOOL, L. V. Surf: Speeded up robust features. **Lecture notes in computer science**, Springer, v. 3951, p. 404–417, 2006.

CALONDER, M. *et al.* Brief: Binary robust independent elementary features. *In*: SPRINGER. **Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part IV 11**. [*S.l.*], 2010. p. 778–792.

CANDIDO, F. B. G. **Estimação de movimento de transdutor ultrassônico utilizando algoritmos de estimação de movimento global**. 2020. Dissertação (Mestrado) — Programa de Pós-Graduação em Engenharia Elétrica – UTFPR, 2020.

CARVALHO, A. *et al.* Mfl signals and artificial neural networks applied to detection and classification of pipe weld defects. **NDT & E International**, v. 39, n. 8, p. 661–667, 2006. ISSN 0963-8695.

CHEN, W. *et al.* Design and fabrication of a high-frequency microconvex array transducer for small animals imaging. **IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control**, v. 69, n. 6, p. 1943–1951, 2022.

COLLINGWOOD, J. Nuclear ndt development at harwell. **NDT International**, v. 20, n. 1, p. 33–41, 1987. ISSN 0308-9126. Disponível em: https://www.sciencedirect.com/science/article/pii/0308912687903701.

CORAMIK, M.; EGE, Y. Discontinuity inspection in pipelines: A comparison review. **Measurement**, v. 111, n. 1, p. 359–373, 2017. ISSN 0263-2241.

DAI, Y.; WU, J. An improved orb feature extraction algorithm based on enhanced image and truncated adaptive threshold. **IEEE Access**, v. 11, p. 32073–32081, 2023.

DOYLE, P.; SCALA, C. Crack depth measurement by ultrasonics: a review. **Ultrasonics**, v. 16, n. 4, p. 164–170, 1978. ISSN 0041-624X.

DRINKWATER, B. W.; WILCOX, P. D. Ultrasonic arrays for non-destructive evaluation: A review. **NDT & e International**, Elsevier, v. 39, n. 7, p. 525–541, 2006.

EPE, E. de P. E. **Matriz Energética e Elétrica**. 2020. Site eletrônico. Disponível em: www.epe.gov.br/pt/abcdenergia/matriz-energetica-e-eletrica. Acesso em: 19 mar. 2022.

ERTURK, S. Digital image stabilization with sub-image phase correlation based global motion estimation. **IEEE transactions on consumer electronics**, IEEE, v. 49, n. 4, p. 1320–1325, 2003.

EXTENDE. **What CIVA brings to NDE**. 2023. Disponível em: https://www.extende.com/what-civa-brings-to-ndt. Acesso em: 14 mai. 2023.

FELDMAN, M. Hilbert transform in vibration analysis. **Mechanical systems and signal processing**, Elsevier, v. 25, n. 3, p. 735–802, 2011.

FUJISAWA, T.; IKEHARA, M. High-accuracy image rotation and scale estimation using radon transform and sub-pixel shift estimation. **IEEE Access**, IEEE, v. 7, p. 22719–22728, 2019.

GAMMA, E. *et al.* **Design patterns: elements of reusable object-oriented software**. [*S.l.*]: Pearson Deutschland GmbH, 1995.

GOUILLART, E. **Scikit-image: image processing**. S.I. Disponível em: https://scipy-lectures.org/packages/scikit-image/index.html. Acesso em: 20 mai. 2023.

GUARNERI, G. A. **Identificação de descontinuidades em peças metálicas utilizando sinais ultrassônicos e técnicas de problemas inversos**. 2015. Tese (Doutorado) — Programa de Pós-graduação em Engenharia Elétrica e Informática Industrial – UTFPR, 2015.

GUERREIRO, M. T. L. **Frequency-domain algorithms for ultrasonic imaging based on interpolation-free stolt migration**. 2020. Dissertação (Mestrado) — Programa de Pós-Graduação em Engenharia Elétrica – UTFPR, 2020.

GUIZAR-SICAIROS, M.; THURMAN, S. T.; FIENUP, J. R. Efficient subpixel image registration algorithms. **Optics letters**, Optical Society of America, v. 33, n. 2, p. 156–158, 2008.

JENSEN, J. A. *et al.* Synthetic aperture ultrasound imaging. **Ultrasonics**, Elsevier, v. 44, p. e5–e15, 2006.

KUMAR, E.; BISWAS, M.; NGUYEN, T. Q. Global motion estimation in frequency and spatial domain. *In*: IEEE. **2004 IEEE International Conference on Acoustics, Speech, and Signal Processing**. [*S.l.*], 2004. v. 3, p. iii–333.

LEUTENEGGER, S.; CHLI, M.; SIEGWART, R. Y. Brisk: Binary robust invariant scalable keypoints. *In*: **2011 International Conference on Computer Vision**. [*S.l.*: *s.n.*], 2011. p. 2548–2555.

LI, C. *et al.* Imaging composite material using ultrasonic arrays. **Ndt & E International**, Elsevier, v. 53, p. 8–17, 2013.

LI, M. *et al.* Dct-based phase correlation motion estimation. *In*: IEEE. **2004 International Conference on Image Processing, 2004. ICIP'04.** [*S.l.*], 2004. v. 1, p. 445–448.

MAIR, E. *et al.* Adaptive and generic corner detection based on the accelerated segment test. *In*: SPRINGER. **Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part II 11**. [*S.l.*], 2010. p. 183–196.

MALLAT, S. *et al.* A wavelet tour of signal processing: the sparce way. **AP Professional, Third Edition, London**, 2009.

MARQUES, J. R. **Aplicação da transformada de Hilbert-Huang na análise das vibrações dos motores de indução de máquinas ferramentas.** 2013. Tese (Doutorado) — Universidade de São Paulo, 2013.

MATHWORKS. **Analyze signals and images in the wavelet domain**. 2023. Disponível em: https://www.mathworks.com/discovery/wavelet-transforms.html. Acesso em: 14 mar. 2023.

MATHWORKS. **Envelope Detection in MATLAB**. 2023. Disponível em: https://www.mathworks.com/help/dsp/ug/envelope-detection.html. Acesso em: 19 mar. 2023.

MATHWORKS. **normalize**. 2023. Disponível em: https://se.mathworks.com/help/matlab/ref/double.normalize.html#mw_03d0c4f9-8ada-420d-8945-a3a901f2700f. Acesso em: 19 mar. 2023.

MUSARATH, T.; SUPRIYA, U. K.; SREELEKHA, G. Sub-block based global motion estimation for affine motion model. *In*: **2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI)**. [*S.l.*: *s.n.*], 2018. p. 665–670.

NAGPAL, A.; GABRANI, G. Python for data analytics, scientific and technical applications. *In*: IEEE. **2019 Amity international conference on artificial intelligence (AICAI)**. [*S.l.*], 2019. p. 140–145.

OPENCV. **About OpenCV**. 2023. Disponível em: https://opencv.org/about/. Acesso em: 20 mai. 2023.

PARKER, S. *et al.* Global and locally adaptive warped motion compensation in video compression. *In*: **2017 IEEE International Conference on Image Processing (ICIP)**. [*S.l.*: *s.n.*], 2017. p. 275–279.

PERCIVAL, D. B.; WALDEN, A. T. **Wavelet methods for time series analysis**. [*S.l.*]: Cambridge university press, 2000. v. 4.

PETROBRAS. **Conceito e significado da auto-suficiência**. 2006. Site eletrônico. Disponível em: https://www.agenciapetrobras.com.br. Acesso em: 19 mar. 2022.

RATAKONDA, K. Real-time digital video stabilization for multi-media applications. *In*: **ISCAS '98. Proceedings of the 1998 IEEE International Symposium on Circuits and Systems (Cat. No.98CH36187)**. [*S.l.*: *s.n.*], 1998. v. 4, p. 69–72 vol.4.

RAVESHIYA, H.; BORISAGAR, V. Motion estimation using optical flow concepts. **International Journal of Computer Technology & Applications**, Citeseer, v. 3, n. 2, 2012.

ROSTEN, E.; DRUMMOND, T. Machine learning for high-speed corner detection. *In*: SPRINGER. **Computer Vision–ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006. Proceedings, Part I 9**. [*S.l.*], 2006. p. 430–443.

RUBLEE, E. *et al.* Orb: An efficient alternative to sift or surf. *In*: IEEE. **2011 International conference on computer vision**. [*S.l.*], 2011. p. 2564–2571.

SARVAIYA, J. N.; PATNAIK, S.; BOMBAYWALA, S. Image registration using log-polar transform and phase correlation. *In*: **TENCON 2009 - 2009 IEEE Region 10 Conference**. [*S.l.*: *s.n.*], 2009. p. 1–5.

SCHIAVI, M. T.; HOFFMANN, W. A. M. Cenário petrolífero: sua evolução, principais produtores e tecnologias. **Revista digital de biblioteconomia e ciência da informação**, Universidade Estadual de Campinas, v. 13, n. 2, 2015. ISSN 1678-765X.

SCHMERR, L. W. **Fundamentals of ultrasonic nondestructive evaluation**: A modeling approach. [*S.l.*]: Springer, 2016.

SHULL, P. J. **Nondestructive Evaluation**: Theory, techniques, and applications. [*S.l.*]: CRC Press, 2002.

SOUZA, A. R. d. **Framework para auxílio no desenvolvimento de algoritmos de reconstrução de imagens em ensaios não destrutivos por ultrassom**. 2019. Dissertação (B.S. thesis) — Universidade Tecnológica Federal do Paraná, 2019.

STACKOVERFLOW. **Stack Overflow Trends**. 2023. Disponível em: https://insights.stackoverflow.com/trends?tags=r\%2Cstatistics. Acesso em: 28 mai. 2023.

STRANG, G.; NGUYEN, T. **Wavelets and filter banks**. [*S.l.*]: SIAM, 1996.

SU, Y.; SUN, M.-T.; HSU, V. Global motion estimation from coarsely sampled motion vector field and the applications. *In*: **2003 IEEE International Symposium on Circuits and Systems (ISCAS)**. [*S.l.*: *s.n.*], 2003. v. 2, p. II–II.

TEAM, S.-I. **General examples**. 2023. Disponível em: https://scikit-image.org/docs/stable/auto_examples/index.html. Acesso em: 20 mai. 2023.

THOMPSON, R.; THOMPSON, D. Ultrasonics in nondestructive evaluation. **Proceedings of the IEEE**, v. 73, n. 12, p. 1716–1755, 1985.

TORR, P. H.; ZISSERMAN, A. Feature based methods for structure and motion estimation. *In*: SPRINGER. **Vision Algorithms: Theory and Practice: International Workshop on Vision Algorithms Corfu, Greece, September 21–22, 1999 Proceedings**. [*S.l.*], 2000. p. 278–294.

VASILEV, D. Python programming training with the robot finch. *In*: IEEE. **2020 XXIX International Scientific Conference Electronics (ET)**. [*S.l.*], 2020. p. 1–4.

ZHOU, C. *et al.* A novel image registration algorithm using wavelet transform and matrix-multiply discrete fourier transform. **IEEE Geoscience and Remote Sensing Letters**, v. 19, p. 1–5, 2022.