INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SANTA CATARINA - CÂMPUS FLORIANÓPOLIS
DEPARTAMENTO ACADÊMICO DE METAL-MECÂNICA
CURSO DE GRADUAÇÃO EM ENGENHARIA MECATRÔNICA

DANIEL RICHARTZ

DESENVOLVIMENTO DE PROGRAMA DE CONSTRUÇÃO DE IMAGEM OBTIDA POR MEIO DE APARELHOS DE TOMOGRAFIA DE IMPEDÂNCIA

FLORIANÓPOLIS, 2022.

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SANTA CATARINA - CÂMPUS FLORIANÓPOLIS
DEPARTAMENTO ACADÊMICO DE METAL-MECÂNICA
CURSO DE GRADUAÇÃO EM ENGENHARIA MECATRÔNICA

DANIEL RICHARTZ

DESENVOLVIMENTO DE PROGRAMA DE CONSTRUÇÃO DE IMAGEM OBTIDA POR MEIO DE APARELHOS DE TOMOGRAFIA DE IMPEDÂNCIA

Trabalho de Conclusão de Curso submetido ao Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina como parte dos requisitos para obtenção do título de Engenheiro Mecatrônico

Orientador:
Prof. Adriano Regis, Msc.

FLORIANÓPOLIS, 2022.

Ficha de identificação da obra elaborada pelo autor.

Richartz, Daniel
DESENVOLVIMENTO DE PROGRAMA DE CONSTRUÇÃO DE IMAGEM OBTIDAOBTIDA POROBTIDA POR MEIOOBTIDA POR MEIO DEOBTIDA POR MEIO DE APARELHOSOBTIDA POR MEIO DE APARELHOS DEOBTIDA POR MEIO DE APARELHOS DE TOMOGRAFIAOBTIDA POR MEIO DE APARELHOS DE TOMOGRAFIA DEOBTIDA POR MEIO DE APARELHOS DE TOMOGRAFIA DE IMPEDÂNCIAOBTIDA
86 p.
Trabalho de Conclusão de Curso (TCC) - Instituto Federal de Santa Catarina, Câmpus Florianópolis. Bacharelado em Engenharia Mecatrônica. Departamento Acadêmico de Metal Mecânica.
Inclui Referências.

1. TIE. 2. Sistemas Embarcados. 3. Processamento De Imagem. 4. Engenharia Mecatrônica. 5. Raspberry. I. Regis, Adriano. II. Instituto Federal de Santa Catarina. III. DESENVOLVIMENTO DE PROGRAMA DE CONSTRUÇÃO DE IMAGEM OBTIDAOBTIDA POROBTIDA POR MEIOOBTIDA POR MEIO DEOBTIDA POR MEIO DE APARELHOSOBTIDA POR MEIO DE APARELHOS

DESENVOLVIMENTO DE PROGRAMA DE CONSTRUÇÃO DE IMAGEM OBTIDA POR MEIO DE APARELHOS DE TOMOGRAFIA DE IMPEDÂNCIA

DANIEL RICHARTZ

Este trabalho foi julgado adequado para obtenção do título de Engenheiro Mecatrônico e aprovado na sua forma final pela banca examinadora do Curso Superior de Engenharia Mecatrônica do Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina.

Florianópolis, 19 de julho, 2022.

Banca Examinadora:

gov.br
Documento assinado digitalmente
ADRIANO REGIS
Data: 22/08/2022 11:40:52-0300
Verifique em https://verificador.iti.br

Adriano Regis, Msc.

GOV.BR
Documento assinado digitalmente
FRANCISCO RAFAEL MOREIRA DA MOTA
Data: 22/08/2022 14:37:37-0300
CPF: 012.003.153-16
Verifique as assinaturas em https://v.ufsc.br

Francisco Rafael Moreira da Mota, Dr.

IFSC
Documento assinado digitalmente
FRANCISCO EDSON NOGUEIRA DE MELO
Data: 22/08/2022 17:40:25-0300
CPF: 261.107.173-04
Verifique as assinaturas em https://v.ifsc.edu.br

Francisco Edson Nogueira De Melo, Me.

A todos que fizeram parte dessa jornada,
em especial minha família e amigos.

AGRADECIMENTOS

Durante a jornada que está para se encerrar, inúmeras são as pessoas ao qual agradeço por participarem e me fornecerem esse momento tão aguardado por muitos anos em minha vida.

Primeiramente agradecer a Deus e a todos de minha família, minha mãe por ser essa mulher maravilhosa que sempre me apoiou e me escutou, meu pai por comprar meu primeiro Arduino e a culpada por me apresentar a engenharia, minha irmã, que tanto amo, por todos os momentos de suporte que me deram e que possibilitaram chegar aqui, meu muito obrigado. Ainda me lembro do dia em que contei para vocês que entrei na faculdade, lembro do olhar de vocês com um sorriso no rosto e os olhos cheios de lágrimas de felicidade.

Agradeço a todos os colegas que fizeram parte dessa jornada nos últimos anos, vocês farão parte das melhores lembranças de minha vida. Me lembro das nossas colaborações ao estudar, e acreditem, vocês fizeram surgir uma versão minha que eu desconhecia.

Gostaria de agradecer as pessoas que se tornaram parte de minha vida, não somente dentro da faculdade, mas fora dela, aos meus amigos Tarcísio, Everton, Leonardo e Mikael, por suportarem minhas crises existenciais e me mostrarem que sou mais capaz do que imagino.

Agradecer a Duda e Luíza, pelos abraços que me deram quando estava desacreditado de mim mesmo, e um muito obrigado a minha chefe Maria, que sempre que precisei, me liberou e possibilitou vários momentos para eu realizar o presente trabalho, além de acreditar em mim.

Aos colegas de turma Bruno, João, Luca, Ingon e Ellen, uma verdadeira quadrilha que sempre me fez rir nos momentos que eu não poderia, alegrando meu dia. Aos colegas Matheus, Pedro, Lucas e Arthur, por sempre mostrarem pontos de vista diferentes e apresentarem um mundo muito mais tranquilo e feliz. Agradeço também a todos os demais colegas de grupos de pesquisa e amizades que encontrei ao longo desta jornada, por me fornecerem a possibilidade de compartilhar um pouco de suas experiências comigo.

Por último, mas não menos importante, a todos os professores do IFSC que me forneceram a solida base para esse momento, em especial meu orientador Adriano Regis, pela orientação no TCC e na vida, ao professor Francisco Mota por sanar dúvidas de forma eficaz e ao Jeremias Stein por ser um excelente professor.

"A lealdade é uma via de mão dupla"
Harvey Specter

RESUMO

Para os profissionais da área de saúde, é de suma importância a capacidade de identificar e visualizar regiões do corpo humano em busca de possíveis variações que possam comprometer o bem estar do paciente. No entanto, nem todas as partes vitais do corpo humano são de fácil observação, como os pulmões, sendo necessário o uso de aparatos de tomografia, capazes de visualizar estruturas anatômicas por meio de radiografia. Aparelhos esses que são caros e emitem radiação, o que impede o uso contínuo em pacientes em seus leitos. Uma solução para aparelhos que possam ser utilizados continuamente em leitos de UTI, são EIT, Electrical Impedance Tomography, ou traduzindo tomografia por impedância elétrica conhecido pela sigla TIE. Esses aparelhos têm a possibilidade de serem utilizados de forma contínua, seu funcionamento se dá através de correntes elétricas que percorrem o corpo do indivíduo, que retorna ao aparelho com variações de intensidade. Esse trabalho tem como objetivo o desenvolvimento do *software* de reconstrução de imagem, de forma a possibilitar a visualização de dados obtidos por aparelhos de tomografias de impedância, aplicando o *software* a sistemas embarcados.

Palavras-chave:TIE, Sistemas Embarcados,Processamento de Imagem, Engenharia Mecatrônica, Raspberry.

ABSTRACT

For professionals, it is extremely important to be able to identify and visualize the regions of the body in search of inclusion possibilities that are easily accessible to humans, not all vital parts of the health body. observation, such as the lungs, requiring the use of imaging and imaging devices, visualization through radiography structures. Devices that are expensive and emit radiation, or that prevent continued use on patients in their beds. A solution for devices that can be used continuously in ICU beds, EIT, Electrical Imped Tomography, or translating replacement by electrical impedance known by the acronym TIE. These devices have the possibility of being used continuously, their operation takes place through electric currents that travel through the individual's body, which return to the device with variations in intensity. This work aims to develop the *software* Image orientation, ie a display of enabling data protection devices impedance, applying *software* to embedded systems.

Keywords:TIE, Embedded Systems, Image Processing, Mechatronics Engineering, Raspberry.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AD	Analógico digital
ARM	Advanced RISC Machine
CAN	Controller Area Network
FPGA	Field programmable gate array
GND	Graduated neutral density filter
GPIO	General Purpose Input/Output
HEX	Hexadecimais
I2C	Inter-Integrated Circuit
IDE	Integrated Development Environment
INCA	Instituto Nacional do Câncer
LBP	Linear Back-Projection
OpenCV	Open Computer Vision
RGB	Red,Green,Blue
SPI	Serial Peripheral Interface
TIE	Tomografia por impedância elétrica
VSCode	Visual Studio Code

SUMÁRIO

1 INTRODUÇÃO

A tomografia computadorizada segundo Morsch (2022), é um exame semelhante ao raio X, utilizado para obtenção de imagens internas do paciente, produzindo diversas imagens transversais que, após serem processadas, geram imagens possíveis de serem analisadas pelos profissionais de saúde. Esse exame tem como objetivo agilizar o diagnóstico de forma não invasiva, no entanto, assim como no raio X, os aparelhos utilizam de radiação ionizante que, segundo o Instituto Nacional do Câncer INCA (2022), tem a capacidade de realizar danos às cadeias do DNA, e podem contribuir, com a exposição por longos períodos, ao desenvolvimento de câncer. Por isso, esse método não pode ser utilizado em regimes permanentes pelos pacientes, além de serem máquinas enormes e que demandam uma estrutura especializada para a sua utilização.

Para Singh et al (2019), uma solução que se vem mostrando promissora é a TIE, ou tomografia de impedância elétrica. Essa técnica consiste na aplicação de potenciais elétricas através do corpo por meio de múltiplos eletrodos dispostos no corpo do paciente. Ao obter os dados o equipamento os envia para a unidade de processamento que possui um *software* capaz de realizar cálculos complexos e, a partir dos resultados, reconstruir uma imagem que traduz visualmente o interior do corpo humano.

As primeiras utilizações do método segundo RIBEIRO (2017) “foram obtidas com o protótipo desenvolvido por Brown et al. (1985), do Departamento de Física Médica e Engenharia Clínica da Universidade de Sheffield (Reino Unido).” Nesse primeiro protótipo as imagens geradas eram de baixa qualidade, porém foi possível validar e dar início para desenvolver o método.

O presente trabalho, busca realizar o desenvolvimento de *software* de reconstrução de imagem para ser utilizado em plataformas embarcadas, de forma a possibilitar atualizações e melhorias constantes, ainda sim permitir a integração com diversos sensores de tomografia por impedância elétrica.

1.1 Justificativa

Ao utilizar tecnologias TIE, são necessários *softwares* capazes de realizar o processamento dos dados obtidos, com o objetivo de apresentar ao profissional de saúde de forma visual e fácil interpretação. Os sistemas embarcados tem como objetivo gerar uma maior disponibilidade ao acesso, assim como, facilidade em levar aparelhos TIE para diversas localidades e viabilizando a construção em diversas plataformas existentes no mercado atual.

Este trabalho inicia discussões do desenvolvimento de *softwares* que pos-

sam realizar esse processamento utilizando sistemas embarcados, uma vez que esse nicho ainda é pouco explorado.

1.2 Objetivo Geral

Os objetivos neste trabalho estão separados em principal, englobando o aspecto geral do desenvolvimento, e em específico, informando objetivos que proporcionaram o desenvolvimento completo do projeto

1.2.1 Principal

O objetivo principal do projeto é o desenvolvimento de um software de reconstrução de imagem para aparelhos de tomografia por impedância, possível de ser implementado em plataformas embarcadas como a Raspberry pi.

1.2.2 Objetivos Específicos

Os objetivos específicos são:

- Demonstrar o uso de plataformas embarcadas para a utilização na saúde;
- Vincular conhecimentos adquiridos na engenharia com aplicações relacionadas à medicina;
- Desenvolver um programa base que possa ser reutilizado e melhorado ao longo do tempo;
- Utilizar de linguagens de programação que tornem o processo de desenvolvimento eficiente e possuam vasta documentação disponível.

1.3 Estrutura do Trabalho

O trabalho foi dividido em fundamentação teórica com o embasamento do método a ser apresentado, materiais e métodos apresentando as ferramentas que foram utilizadas assim como a lógica de implementação, apresentação dos resultados com os resultados obtidos a partir de materias e métodos e por último considerações finais com os resultados e discussões finais, assim como, possíveis trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Segundo Perlin e Pinto (2013), tomografia é a composição de duas palavras gregas, sendo tomo que significa “corte” e grafia que significa “imagem”, tomografia pode ser descrita como a composição de imagens por cortes. O início do desenvolvimento se deu com os trabalhos do físico alemão Conrad Röntgen, quando seus trabalhos com radiação no ano de 1895 geraram o exame de raio X. A partir desse exame era possível realizar a primeira visualização interna de um corpo, no entanto, demonstrava um corpo físico tridimensional em somente duas, era necessário um método de transformar projeções bidimensionais em objetos complexos tridimensionais.

Os estudos que relacionam a reconstrução de objetos por meio de equações, são os trabalhos do matemático Johann Radon, no início do século XX na Áustria, que possuíam como objetivo mostrar a capacidade de descrever objetos tridimensionais realizando composições de projeções bidimensionais em múltiplos ângulos. As equações desse estudo foram denominadas de transformada de Radon.

Como comentado por Carvalho *et al.* (2007), ambos os estudos, de Radon e Röntgen, foram as bases para os estudos desenvolvidos a partir dos anos 50 no campo de reconstrução de imagem, que com o advento da computação nos anos seguintes e entrada de cena o engenheiro Hounsfild em conjunto com o Dr. James Ambrose, aplicando algoritmos de reconstrução de imagens ao raio X, apresentaram em 1972 um novo método, reconstruindo imagens tridimensionais com base em projeções bidimensionais.

Esse método foi a base para o desenvolvimento de diversos técnicas, que não se restringiram somente ao meio médico mas se aplicando em outros setores, utilizando múltiplas formas de conceber as imagens, como descrito por Perlin e Pinto (2013), com a aplicação do método de tomografia por ultrassonografia, utilizando o som para a verificar a não homogeneidade em vigas de concreto e tomografia capacitiva ECT descrita por Jia, Wang e Millington (2017), como boas alternativas para a indústria pela tempo de resposta inferior aos existentes.

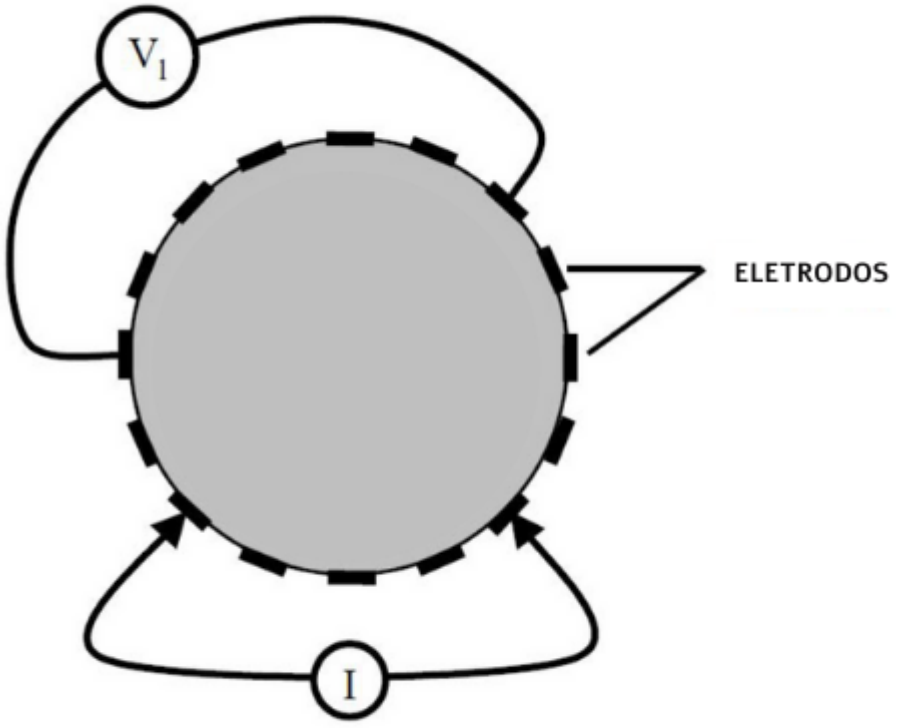
Na área médica umas das técnicas com grande atrativo, segundo singh *et al.* (2019), é a TIE, devido a suas aplicações clínicas indo à encontrar funções cerebrais, atividade neural, hipertermia, investigações patológicas de câncer, entre outras aplicações, sendo que não utiliza radiação ionizante. singh *et al.* (2019) relatam que essa técnica possui tempos de aquisição baixo, porém é suscetível a ruídos. Todavia sua aplicação se torna viável devido ao fato de ser consideravelmente barata e de aplicação ágil em comparação com métodos utilizados, sendo útil para monitoramentos contínuos.

2.1 Tomografia por impedância

A tomografia por impedância, TIE, segundo Arellano *et al.* (2021) consiste na aplicação de uma baixa corrente elétrica com o intuito de gerar imagem a partir da variação de impedância obtida no corpo de interesse. As aplicações desse método de tomografia abrangem aplicações médicas como detecção de câncer de mama, problemas respiratórios e monitoramento cardiovascular, aplicações industriais e no ramo da construção civil sendo aplicado no monitoramento de estruturas. Dentre os métodos de obtenção de imagem, a TIE oferece uma solução prática e de baixo custo operacional.

O método de obtenção de dados da TIE, conforme Arellano *et al.* (2021) comentam, consiste da utilização de eletrodos dispostos em uma seção transversal espaçados de forma igualitária do objeto de estudo em questão, sendo os eletrodos o meio físico para a obtenção das variações de impedância, os mesmos realizam duas funções, emissores de corrente e de leitura de tensão.

Figura 1 – Representação leitura TIE



Fonte: Adaptado Arellano *et al.* (2021).

2.2 Impedância elétrica

A impedância é descrita como a oposição à passagem de corrente, sendo essa grandeza a soma da associação de resistência, reatância e capacitância do meio por onde a corrente realiza seu percurso. O valor total de impedância em um meio é descrito pela Equação 1, sendo V a diferença de potencial, I a corrente e Z a impedância (BOLFE *et al.*, 2007).

$$Z = \frac{V}{I} \tag{1}$$

Sendo assim, tomando um valor fixo de corrente em um meio, ao medir a variação de tensão entre dois pontos é possível mensurar a impedância entre os pontos devido a variação de tensão.

2.3 Matriz sensibilidade

A matriz sensibilidade tem como intuito descrever o ganho entre as alterações que as variações de impedância dentro do campo transversal com a implementação dos eletrodos, sendo assim possível mensurar como cada eletrodo será afetado com as variações no sistema em determinado ponto da região de interesse. Para obter essa matriz é necessário realizar a discretização da seção transversal aos eletrodos com o intuito de compreender a interação dos elementos que serão introduzidos, realizando a variação dos pontos, a matriz sensibilidade podem ser obtida excitando o espaço discretizado utilizando os seguintes passos (MOTA, 2015).

- Realiza a discretização da seção transversal de forma a compreender diversos espaços menores, cada espaço discriminado de forma menor é denominado um pixel.
- Por meio de experimentação, a condutividade de um único espaço é variada, e as medidas dos eletrodos é tomada.
- O valor da condutividade no espaço discriminado é retornado ao normal.
- Por meio de um laço de repetição um próximo espaço é escolhido, até finalizar com todos espaços lidos e discriminados.

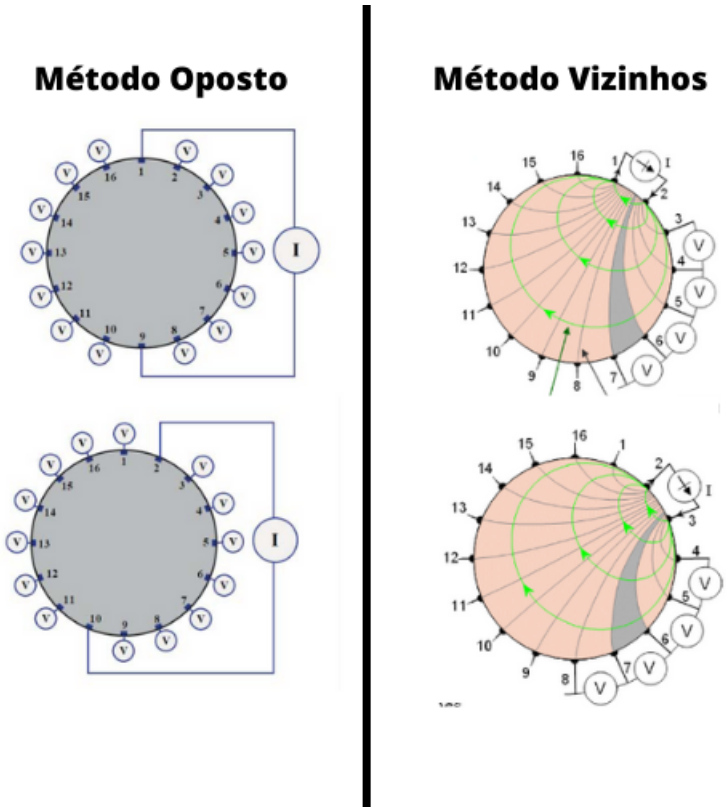
Essa forma de obtenção de variação pode ser feita por espaços discretização de diversas formas, tanto com formas bem definidas quanto em seções transversais orgânicas.

2.4 Modelos de aquisição

Assim como em qualquer método de tomografia, a aquisição dos dados é de suma importância para a reconstrução da imagem, sendo na TIE o ponto de aquisição os eletrodos. Para isso é realizada a passagem de uma corrente conhecida pelo meio, respeitando a Equação 1 da impedância elétrica total em um meio.

Para isso, os eletrodos realizam a função tanto de injeção de corrente quanto de leitores de tensão, existindo métodos de ordenar e ler os eletrodos. Descrevendo cada um desses métodos, Rymarczyk (2019) destaca os métodos emissores opostos e método de medição adjacente, em que o eletrodo exerce a função de injetar corrente no mensurando ou realizar a leitura da variação da tensão, tendo em vista ainda a existência de outros métodos onde o mesmo eletrodo que exerce a função de injetar corrente realizar leitura da variação de tensão. A Figura 2 mostra a leitura dos dois métodos descritos.

Figura 2 – Métodos



Fonte: Adaptado Rymarczyk (2019) e Arellano *et al.* (2021).

2.5 Trabalhos correlatos

Entre os estudos e sistemas TIE, uma série de variações de *hardware* é utilizada, tanto para comutar os eletrodos como para realizar as leituras analógicas e reconstrução da imagem. Comumente são utilizados multiplexadores para gerir as saídas, comutando qual eletrodo será destinado a leitura de tensão e qual será utilizado

para a emissão de corrente, no entanto outras formas também são utilizadas. O autor Aris (2018) utiliza de comutação utilizando relés, pois segundo o mesmo, a utilização de multiplexadores pode gerar a existência de impedâncias parasitas.

Porém, como comentado pelos autores Sohal *et al.* (2014), como o intuito é ter aquisições de forma mais rápidas e com circuitos menores tanto na parte de comutação, leitura e reconstrução de imagem, multiplexadores são alternativas viáveis assim como para compactar todo o *hardware*, utilizando de chips com arquitetura FPGA (*field programmable gate array*). Outros sistemas como o desenvolvido pelos autores Arellano *et al.* (2021), utilizam para gerenciar o multiplexador e realizar as leituras AD um Arduino mega, sendo que a reconstrução da imagem fica a cargo de um Raspberry pi, com implementação do sistema de reconstrução EIDORS, um sistema de código aberto desenvolvido para operar por meio das plataformas Octave e Matlab.

Autores como singh *et al.* (2019) utilizam o Raspberry para gerir os eletrodos e realizar as aquisições por meio de conversores AD, analógico digital, externos e posteriormente transmi o valor digital para um computador externo. Esses autores mostram a pluralidade das formas em desenvolvimento de tecnologias TIE, sendo utilizadas diversas tecnologias.

2.6 Reconstrução da imagem

Ao introduzir um corpo no meio eletrólito de estudo, uma variação de impedância é identificada, sendo assim, possui uma divergência entre as medições de tensão entre os eletrodos com base na medida de padrão do meio sem o corpo estranho. Os dados gerados por meio dessa variação apresentam características não lineares, no entanto, se convencionar uma variação pequena nos entornos do objeto, é possível realizar a linearização para gerar uma imagem utilizando a Equação 2 (JIA *et al.*, 2021)(YANG; PENG, 2003).

$$V = S \cdot \sigma \tag{2}$$

sendo que V (n x 1) o vetor com os valores de tensões lidas pelos eletrodos ao introduzir um objeto que provoca a variação das impedâncias, S é a matriz (m x n) de sensibilidade da e σ (n x 1) o vetor condutividade.

Para a reconstrução da imagem, se faz necessária uma abordagem inversa, já que a partir de valores de tensão V tem como objetivo a matriz σ , levando como premissa a linearidade do sistema para pequenas variações e equacionando a Equação 2, é possivel chegar na Equação 3.

$$\sigma = S^{-1} \cdot V \tag{3}$$

No entanto, como S não é uma matriz quadrada, como descreve MOTA (2015) é necessário utilizar de outro método para solucionar a equação, utilizando de aproximações e métodos interativos.

2.7 Algoritmo de retroprojeção linear

A retroprojeção linear do inglês *Linear Back-Projection*, LBP , é uma técnica utilizada para a resolução da Equação 2, sendo amplamente utilizada em métodos computacionais por proporcionar velocidade de execução e ainda gerar uma aproximação muito boa do sistema real, sendo obtida aproximando a matriz S^{-1} como sendo S^T (NADA, a; DAVIDSON *et al.*, 2004). Ao realizar essa aproximação se chega a Equação 4.

$$\sigma = S^T \cdot V \tag{4}$$

Nessa equação, S é a matriz sensibilidade e V é o valor da variação de tensão normalizado. Por se tratar de um método direto, apresenta uma velocidade de cálculo elevada comparado a outros métodos, porém por não ser interativo, apresenta uma baixa qualidade de construção da imagem (JIA *et al.*, 2021).

2.8 Algoritmo de Landweber

Na busca de solucionar a integral de Fredholm, Landweber propôs a criação de um método interativo pois em muitos aspectos físicos, as equações se sobressaem em casos onde seria altamente desejada a realização de sucessivas aproximações. O algoritmo tem como fundamento o método gradiente descendente chegando a Equação 5 (MOTA, 2015; YANG; PENG, 2003; LANDWEBER, 1951) .

:

$$minf(\sigma) = \frac{1}{2} \|S \cdot \sigma - \lambda\|^2 \tag{5}$$

E $f(\sigma)$ sendo descrita pela Equação 6.

$$f(\sigma) = \frac{1}{2}(\sigma^T \cdot S^T \cdot S \cdot \sigma - 2 \cdot \sigma^T \cdot S^T \cdot \sigma + \lambda^T \cdot \lambda) \tag{6}$$

O gradiente de $f(\sigma)$ pode ser definido pela Equação 7.

$$\nabla f(\sigma k) = S^T \cdot (S\sigma - \lambda) \tag{7}$$

Com esse gradiente de $f(\sigma)$, a cada interação decresce mais rapidamente com uma nova direção. Dessa forma é possível expressar o algoritmo em sua forma de interações com a Equação 8.

$$\sigma_{k+1} = \sigma_k + \mu \cdot S^T \cdot (\lambda - S \cdot \sigma_k) \tag{8}$$

Sendo que σ_k é o vetor calculado anteriormente, onde μ é o fator de ganho, e pode ser obtido a partir da Equacão 9 (MOTA, 2015).

$$\mu = \frac{2}{\delta max} \tag{9}$$

sendo que δmax é o maior autovalor de $S^T \cdot S$.

Um dos grandes desafios ao utilizar esse método é sua baixa taxa de convergência, sendo que para melhorar essa taxa pode se aplicar um operador de projeção sobre a equação entre as interações, nesse caso aplicando o operador F mostrado na Equação 10 (MOTA, 2015; YANG; PENG, 2003).

$$F(g(x)) = \left\{ \begin{array}{ccc} 0 & se & g(x) < 0 \\ g(x) & se & 0 \leq g(x) \leq 1 \\ 1 & se & g(x) > 1 \end{array} \right\} \tag{10}$$

sendo que g(x) é dado pela Equação 11.

$$g(x) = \sigma_k + \mu \cdot S^T \cdot (\lambda - S \cdot \sigma_k) \tag{11}$$

sendo assim, aplicando a g(x) em F se tem a Equação 12

$$\sigma_{k+1} = F(g(x)) \tag{12}$$

Sendo essa uma versão de aplicação do algoritmo de Landweber para a reconstrução de imagens por interações.

2.9 Eletrodos

A aquisição das variações de tensões, assim como a emissão de corrente ocorre por meio do auxílio de eletrodos em contato com o meio. Os eletrodos devem apresentar certas características a fim de gerar resultados precisos e satisfatórios para a leitura da variação de tensão. Entre as características estão a estabilidade do material, boa condutividade e pouco ou nenhuma reação com o meio (TAKA, 2008).

3 MATERIAS E MÉTODOS

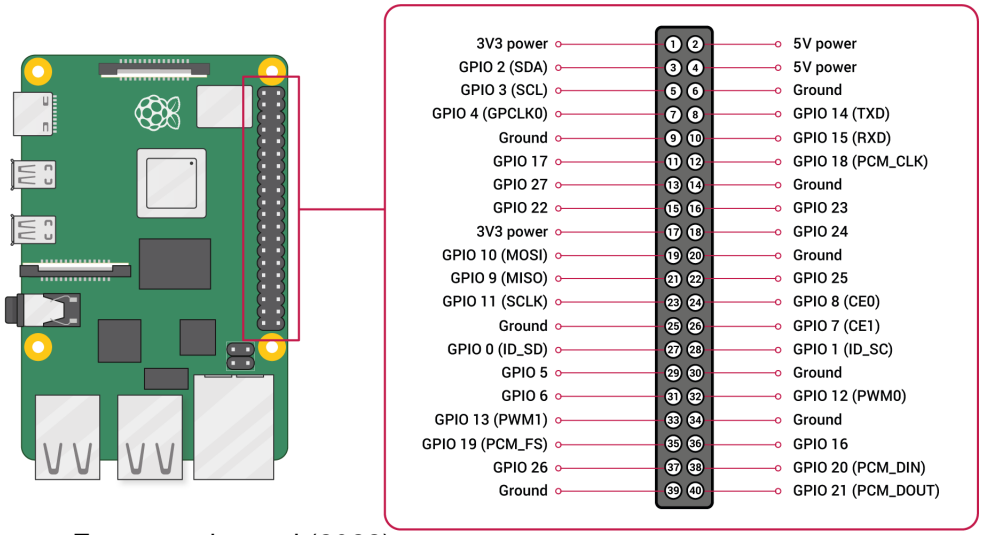
Com o intuito de realizar o desenvolvimento do programa de reconstrução de imagens, uma série de ferramentas foram selecionadas a fim de se obter a imagem reconstruída. Essas ferramentas teriam como objetivo um melhor aproveitamento e um desenvolvimento ágil. Entre as ferramentas podemos destacar o *hardware* embarcado, utilizando linguagem de programação e bibliotecas que auxiliam no desenvolvimento.

3.1 Rasberry pi 3b

Para a placa central, responsável por executar e realizar os algoritmos de reconstrução de imagens, optou-se pela utilização de uma Raspberry pi model 3 b. Sendo esse um computador de placa única, já possui processador integrado, conexão com rede ethernet, portas USB, saídas de imagem e áudio, sendo uma aplicação completa de um computador necessitando apenas de periféricos como teclado, *mouse* e monitor. O mesmo ainda possui 40 conectores sendo uma boa parte de portas lógicas programáveis conhecidas como GPIO, do inglês *General Purpose Input/Output,* permitindo realizar acionamento de periféricos, entre essas saídas ainda possuem a capacidade de integração com protocolos de comunicação com I2C, SPI e Serial (RASPBERRYPI, 2022).

O Raspberry pi permite minimizar aplicações e, dessa forma, é possível tornar o sistema móvel e de baixo consumo (SINGH *et al.*, 2019). A Figura 3 representa a pinagem do rasberry pi 3 com as funções de cada pino.

Figura 3 – GPIO Raspberry pi 3 B



Fonte:raspberrypi (2022)

3.2 Python

Entre as linguagens de programação para o desenvolvimento do programa, a que apresentou um relação custo/benefício adequada foi a linguagem python, pois uma linguagem com uma rápida curva de aprendizado, capaz de rodar em múltiplas plataformas inclusive na plataforma Raspberry pi (VALLAT, 2018).

O python nos últimos anos vem ganhando sua notoriedade, devido à sua sintaxe de alto nível, tornando seu aprendizado dinâmico, sendo amplamente utilizada em diversas aplicações como aprendizado de máquina, computação científica, análise de dados e visão computacional, tudo isso junto a uma comunidade ampla com múltiplas bibliotecas e material de apoio disponível, tornando umas das linguagens mais efetivas em desenvolvimentos rápidos no meio científico e de prototipagem (RASCHKA; PATTERSON; NOLET, 2020).

3.3 Bibliotecas de programação

Para auxiliar no desenvolvimento, foi utilizado uma série de bibliotecas de programação, podendo citar duas como as principais a serem utilizadas, a biblioteca *Numpy*, que fornece a possibilidade de trabalhar com arrays multidimensionais e ainda possibilita de forma prática a multiplicação matricial, transformadas de Fourier, equações de álgebra linear e simulações a tornando uma biblioteca fundamental para a computação científica (DEVELOPERS, 2022).

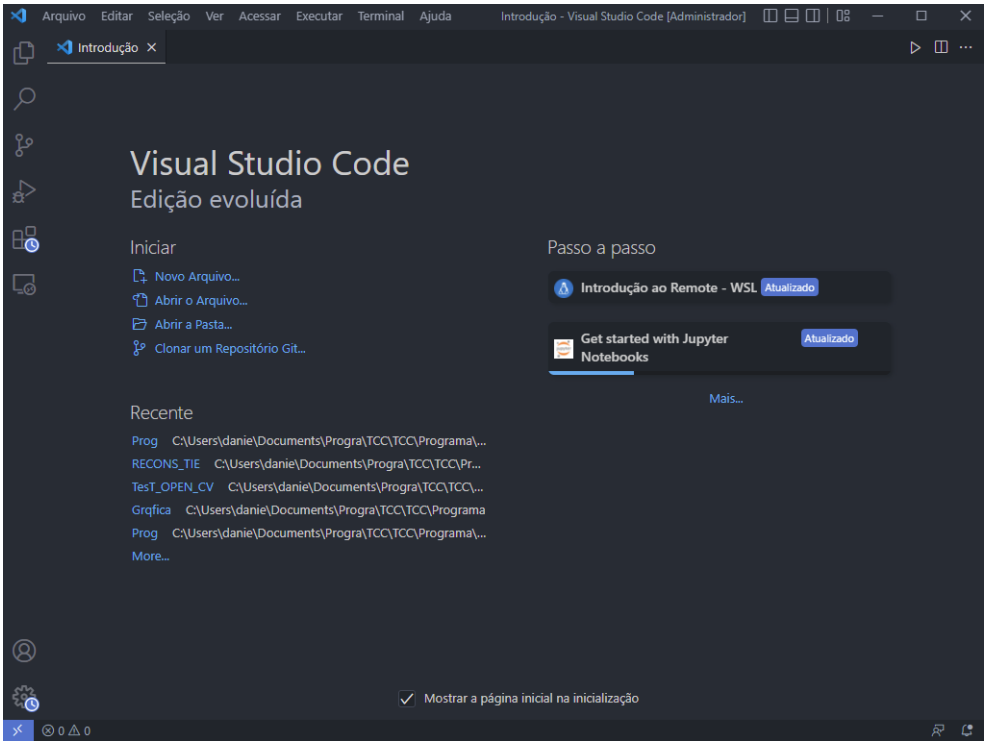
A segunda biblioteca é a *OpenCV*, desenvolvida para trabalhar com visão computacional e aprendizado de máquina, possuindo ferramentas para identificar objetos, classificar ações humanas, reproduzir nuvem de pontos e reconstruir imagens, sendo possível programar usando python (OPENCV, 2022). A *openCV* é a principal aplicação para processamento de imagens em tempo real (FARIA; FARIA, 2021).

Para a criação da aplicação da interface de usuário com o programa será utilizada a biblioteca *Qt Creator* e sua aplicação visual, com o intuito de agilizar e fornecer ferramentas necessárias para a criação da mesma.

3.4 Ambiente de desenvolvimento

Para realizar o desenvolvimento foi optado pela utilização da ferramenta de edição de texto Visual Studio Code, VSCode, que possui uma série de plugins que aceleram desenvolvimentos de *software* em diversas linguagens de programação, entre elas o python, linguagem principal de implementação do código. É possível ver o ambiente de desenvolvimento na Figura 4.

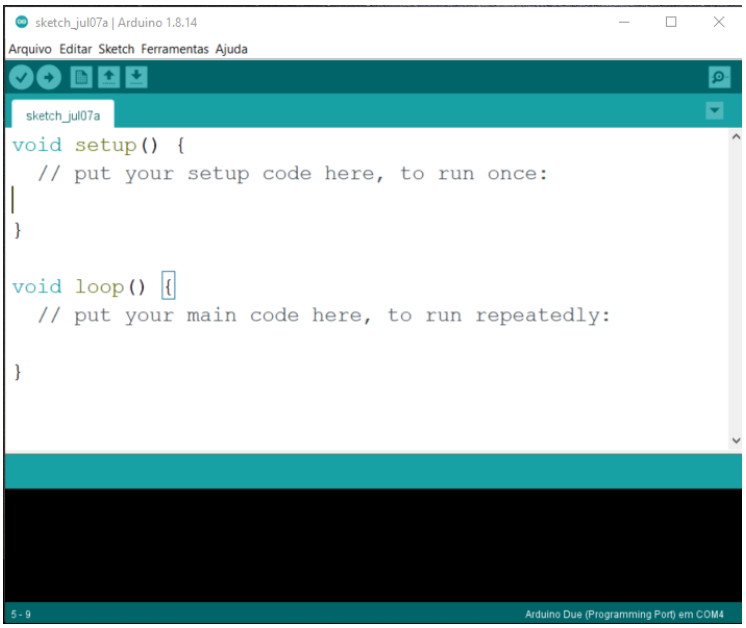
Figura 4 – Ambiente de desenvolvimento do VSCode



Fonte:Autor

A programação da placa Arduino due, foi por meio da IDE, *Integrated Development Environment* ou traduzindo Ambiente de Desenvolvimento Integrado, nativa disponibilizada pela própria empresa desenvolvedora da placa, no caso a Arduino IDE mostrada na Figura 5, que já possui ferramentas de desenvolvimento do programa para esse dispositivo.

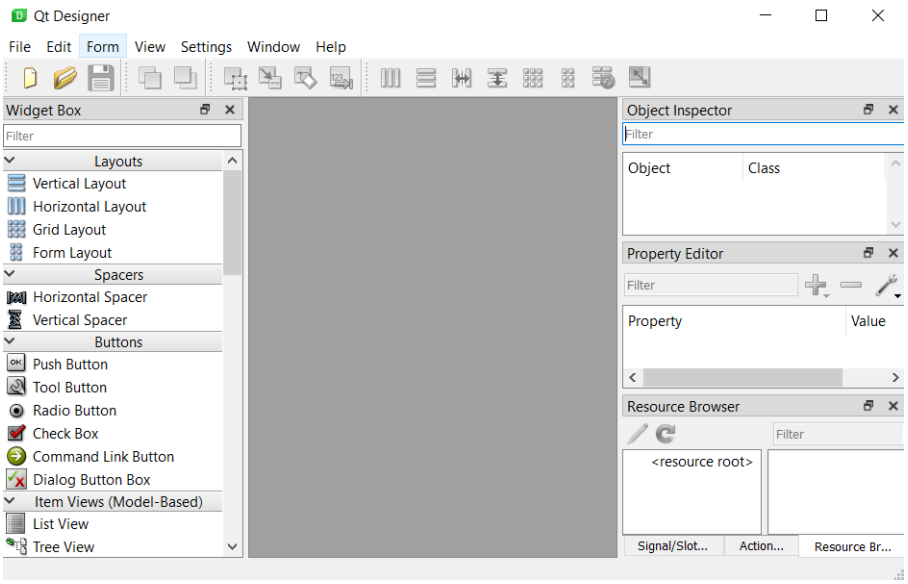
Figura 5 – Ambiente de desenvolvimento do Arduino IDE



Fonte:Autor

Como já comentado, para a criação da interface, foi escolhida a biblioteca Qt creator que possui uma interface de criação gráfica para auxiliar e agilizar a criação, essa interface é possível ver na Figura 6 .

Figura 6 – Ambiente de desenvolvimento do QTcreator livre



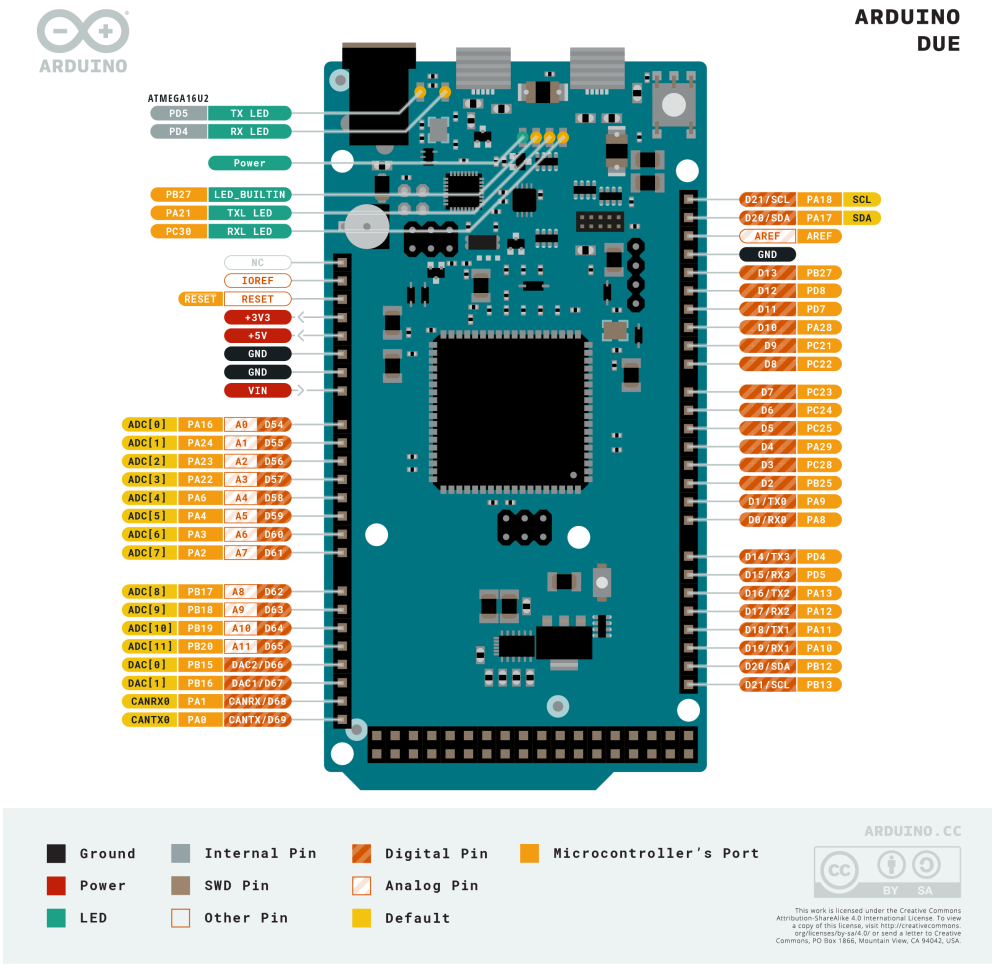
Fonte:Autor

3.5 Arduino due

O Arduino due é uma placa de desenvolvimento construída em torno do microcontrolador Atmel SAM3X8E ARM Cortex-M3, sendo um processador ARM de 32 bits com 54 pinos GPIO que podem ser programadas como entrada ou saídas, 12 entradas analógicas com um conversor AD de 12 bits trabalhando com variações de tensões de 0 à 3,3 volts e suporte a protocolos de comunicação como SPI , I2C e suporte ao barramento CAN (ARDUINO, 2022).

A placa Arduino due é uma das placas da linha de prototipagem e desenvolvimento Arduino, sendo descrito por PALLARO (2019) como um dos modelos com maior desempenho de processamento. A Figura 7 mostra à placa junto a especificação de cada pino.

Figura 7 – GPIO Arduino due



Fonte:arduino (2022)

3.6 SPI

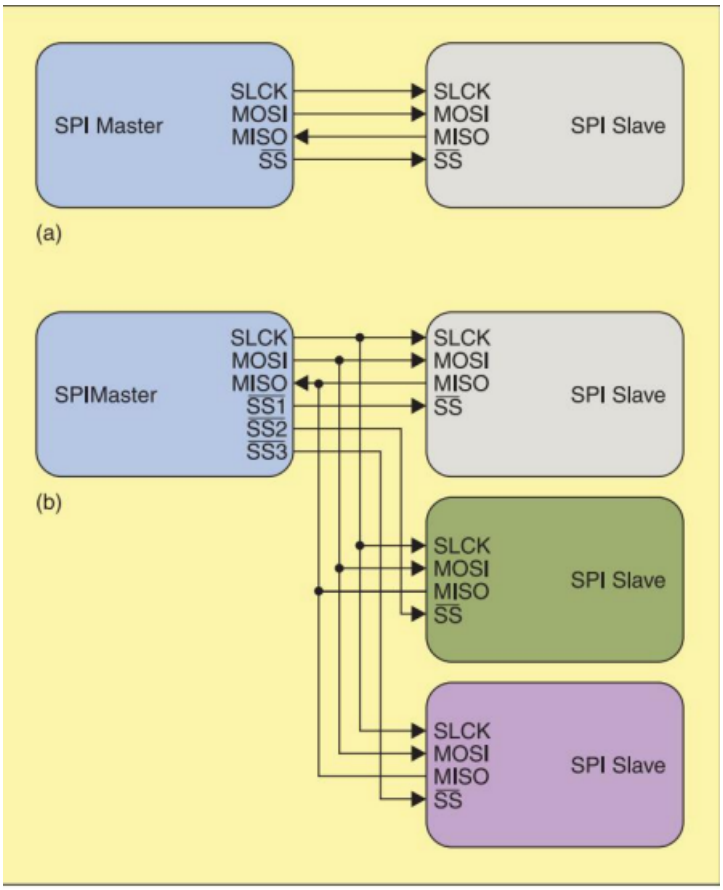
Para a aquisição de dados entre o Raspberry pi e o Arduino due, foi utilizado a comunicação SPI, um protocolo serial comumente utilizado para realizar a transferência de dados, sendo um protocolo de transmissão síncrona, transmite e recebe dados simultaneamente, baseado em interface mestre escravo, em que ambos recebem e encaminham mensagens. Nesse protocolo o mestre pode se comunicar com vários escravos, fazendo o uso de um sinal de clock para realizar o deslocamento de bits. O protocolo SPI utiliza 3 ligações comuns, sendo necessário mais uma conexão por dispositivo escravo conectado ao mestre, sendo as conexões (ANAND *et al.*, 2014):

- SCLK: O sinal de *clock*, responsável por sincronizar os dispositivos.
- MOSI: Linha de dados que transmite informações do bit do mestre para o escravo.
- MISO: Linha de dados de transmissão do escravo para o mestre.

- SS: *Slave select*, responsável por habilitar o escravo para receber e transmitir dados.

O autor Leens (2009) mostra a utilização de um mestre com somente um escravo, e com múltiplos escravos junto a suas ligações necessárias na Figura 8.

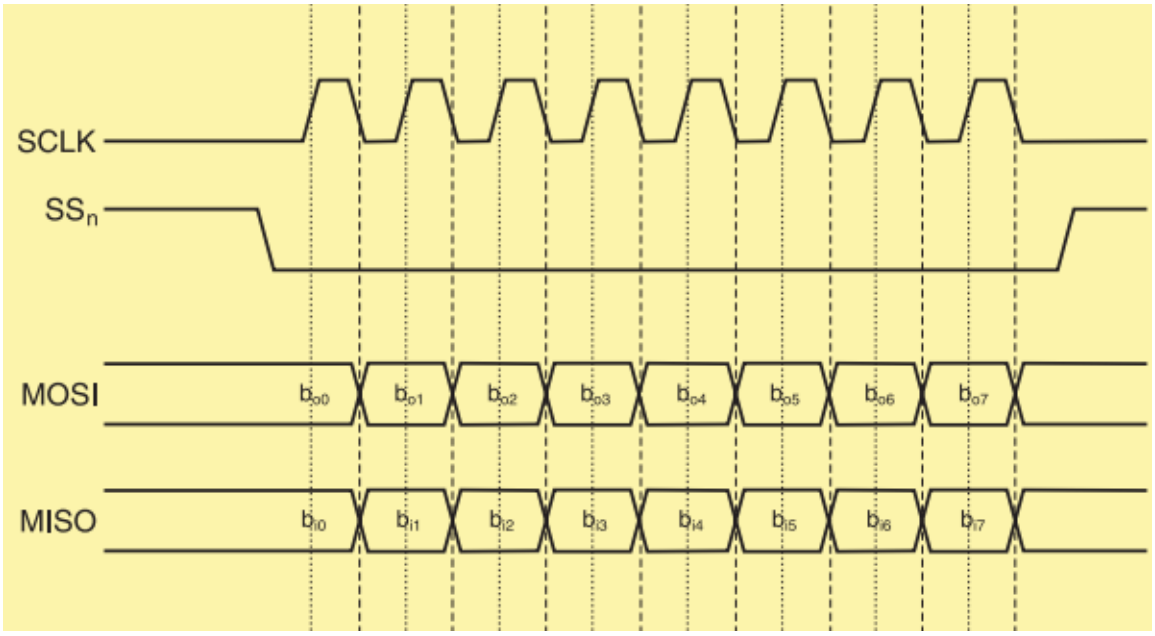
Figura 8 – Conexão mestre escravos SPI



Fonte: Leens (2009)

O barramento de dados ainda pode ser configurado para obter os dados por borda de subida ou descida do SCLK nas linhas MISO e MOSI, e estados da linha enquanto o escravo estiver desabilitado. A Figura 9 monstra o barramento de sinais de comunicação (LEENS, 2009).

Figura 9 – Sinais SPI simples comunicação



Fonte: Leens (2009)

3.7 *Hardware*

Para o sistema de aquisição, foi utilizada a placa desenvolvida com as noções da placa de aquisição de Schluchter (2020), sendo desenvolvida para realizar a aquisição de um conjunto de 16 eletrodos dispostos de maneira circular. O sistema de aquisição é fundamentado em eletrodos e multiplexadores para gerir de forma efetiva qual dos eletrodos será responsável por fornecer corrente, assim como, qual será conectado a fim de se obter a diferença de potencial.

O circuito utiliza um conjunto de multiplexadores capazes de comutar uma saída ou entrada analógica cada multiplexador com uma entrada de 4 bits consegue endereçar até 16 portas, dessa forma foram utilizados 4 multiplexadores modelo 74HC4067, sendo cada um responsável por uma tarefa, apresentada na Tabela 1 (SCHLUCHTER, 2020).

Tabela 1 – Funções dos multiplexadores

Multiplexador	Função
MUX1	Emissor de corrente
MUX2	Dispensar corrente
MUX3	Leitor de tensão
MUX4	Tensão referência

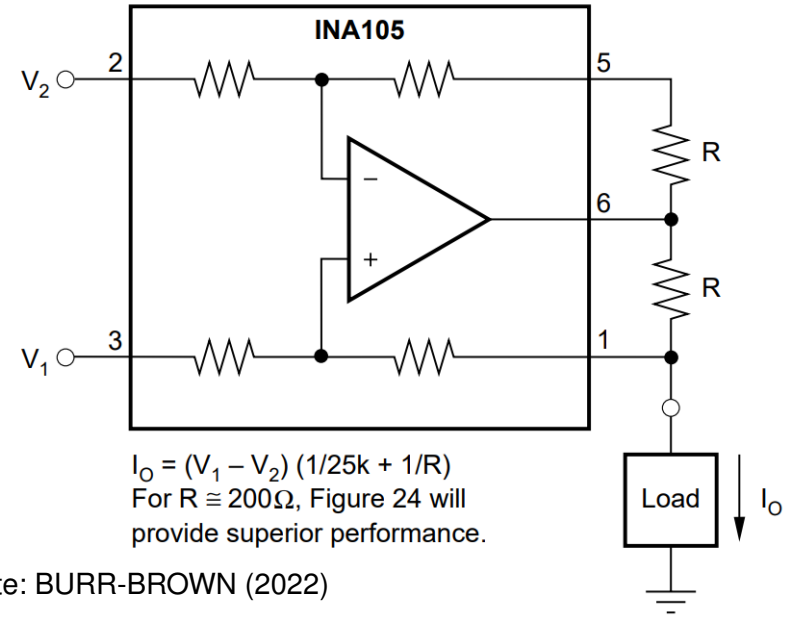
Fonte: Adaptado de (SCHLUCHTER, 2020)

A excitação do sistema de Schluchter (2020), utilizando um gerador de funções com o objetivo de converter a saída de tensão em corrente.

3.7.1 Circuito de excitação

Com a utilização de uma amplificador de instrumentação INA105 da marca Burr-Brown, a tensão proveniente do gerador de funções é convertido em corrente, sendo utilizado a própria configuração recomendada pelo *datasheet* do fabricante como mostrado na Figura 10 (SCHLUCHTER, 2020).

Figura 10 – Fonte de corrente com INA105



Fonte: BURR-BROWN (2022)

Segundo o próprio fabricante para se obter o valor da corrente Io, é necessário utilizar a Equação 13.

$$I_o = (V_1 - V_2) * (\frac{1}{25k} + \frac{1}{R}) \tag{13}$$

Os trabalhos de Schluchter (2020) estipulam como corrente desejada 141μ A com variação de tensão de 0,55V, necessita de uma resistores de 4,6k Ohms obtendo uma corrente de 100μ A.

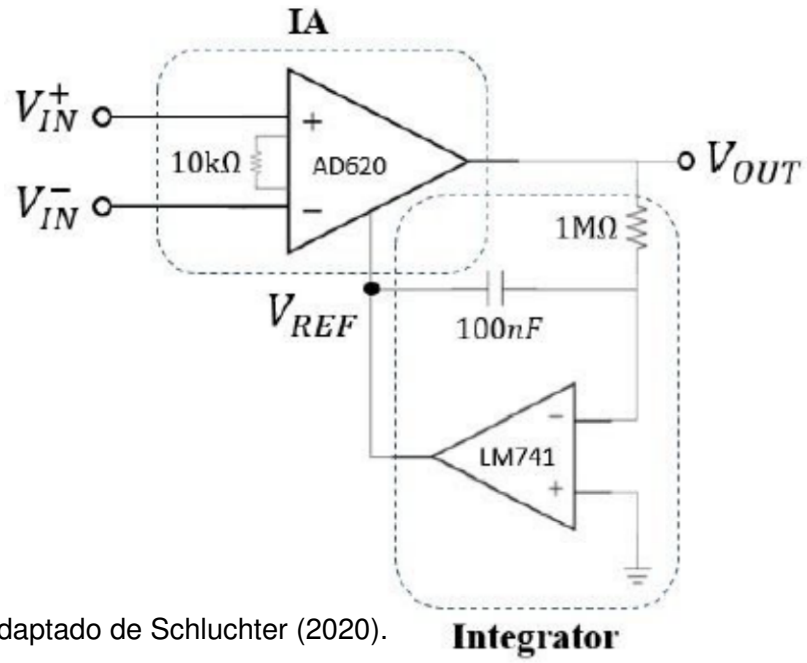
3.7.2 Aquisição de sinais

Nos trabalhos de Schluchter (2020), foi desenvolvido o circuito com um amplificador de instrumentação, filtro passa- alta, filtro passa-baixo e um amplificador AD620. O amplificador utilizado na entrada do circuito foi um AD620 com ganho estipulado pela Equação 14.

$$G = \frac{R}{Rg - 1} \tag{14}$$

Ao estipular R = 49,4 KΩ , definindo o resistor de ganho como Rg = 10KΩ, se consegue um ganho de 5,94 (SCHLUCHTER, 2020). Como o AD620 possui um ganho de corrente contínua, a saída foi conectada um circuito integrador com um TL081, subtraindo a corrente contínua como mostrado na Figura 11.

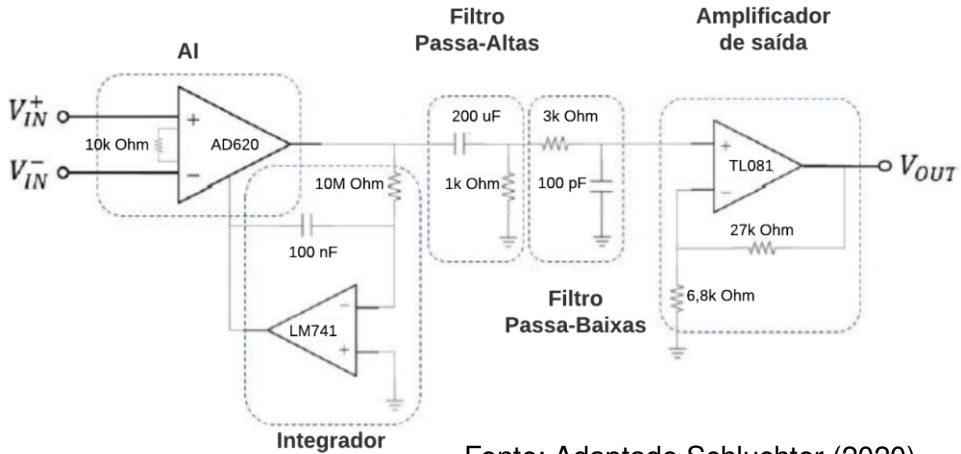
Figura 11 – Amplificador de instrumentação com integrador5



Fonte:Adaptado de Schluchter (2020).

Após essa primeira etapa, o sinal passa pelos filtros de passa-baixa e passa-altas ficando com uma frequência que varia de 0,8Hz e 117Khz (SCHLUCHTER, 2020). Por fim passa por um circuito amplificador utilizando um TL081, tendo como circuito de aquisição de sinal descrito na Figura 12 .

Figura 12 – Circuito de aquisição de sinais do sensor



Fonte: Adaptado Schluchter (2020).

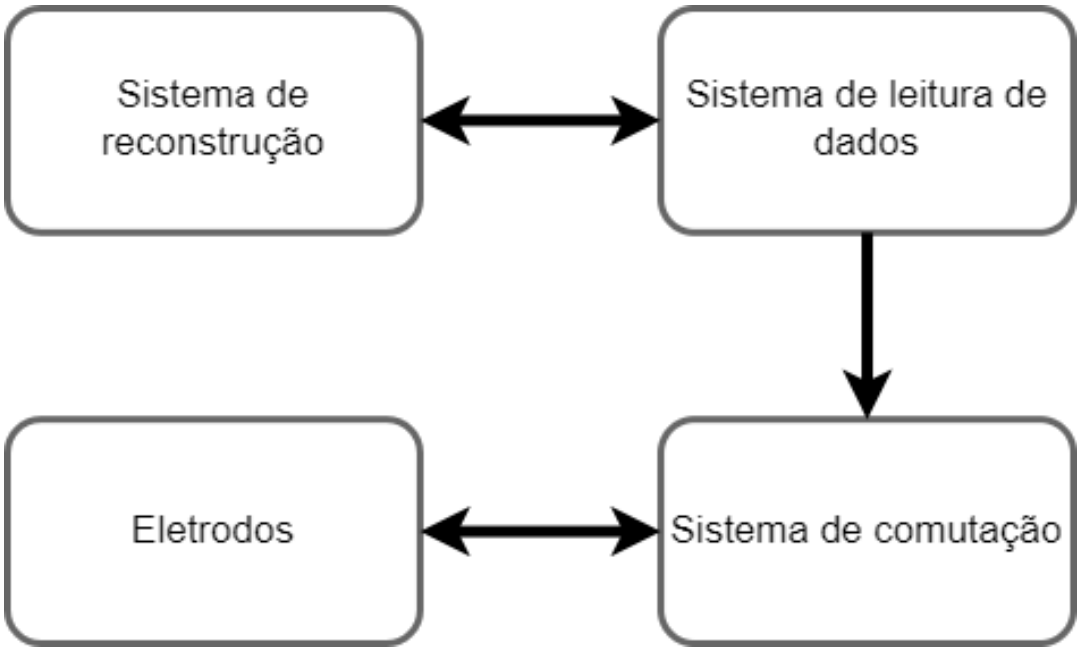
3.8 Arquitetura

No sistema TIE, uma série de componentes é utilizado, sendo eles responsáveis por realizar a comutação entre os eletrodos, aquisições de dados e interpretação dos dados. Podemos subdividir os módulos da seguinte forma (BARROS, 2011):

- Eletrodos : responsável por realizar a leitura de tensão e realizar a passagem de corrente pelo meio;
- Sistema de comutação: responsável por comutar qual eletrodo realizará a etapa de leitura e qual o de emissor de corrente;
- Sistema de leitura de dados: responsável por ler o valor da variação de tensão;
- Sistema de reconstrução: responsável por gerir a aquisições de dados, e realizar a reconstrução da imagem.

Utilizando esses módulos, foi realizado o desenvolvimento do programa e o fluxo de etapas para a comutação de eletrodos, aquisição de sinal e por fim a reconstrução da imagem. Na Figura 13 é mostrado como funciona o fluxo de dados assim como a comunicação existente entre os módulos.

Figura 13 – modelo de fluxo de dados

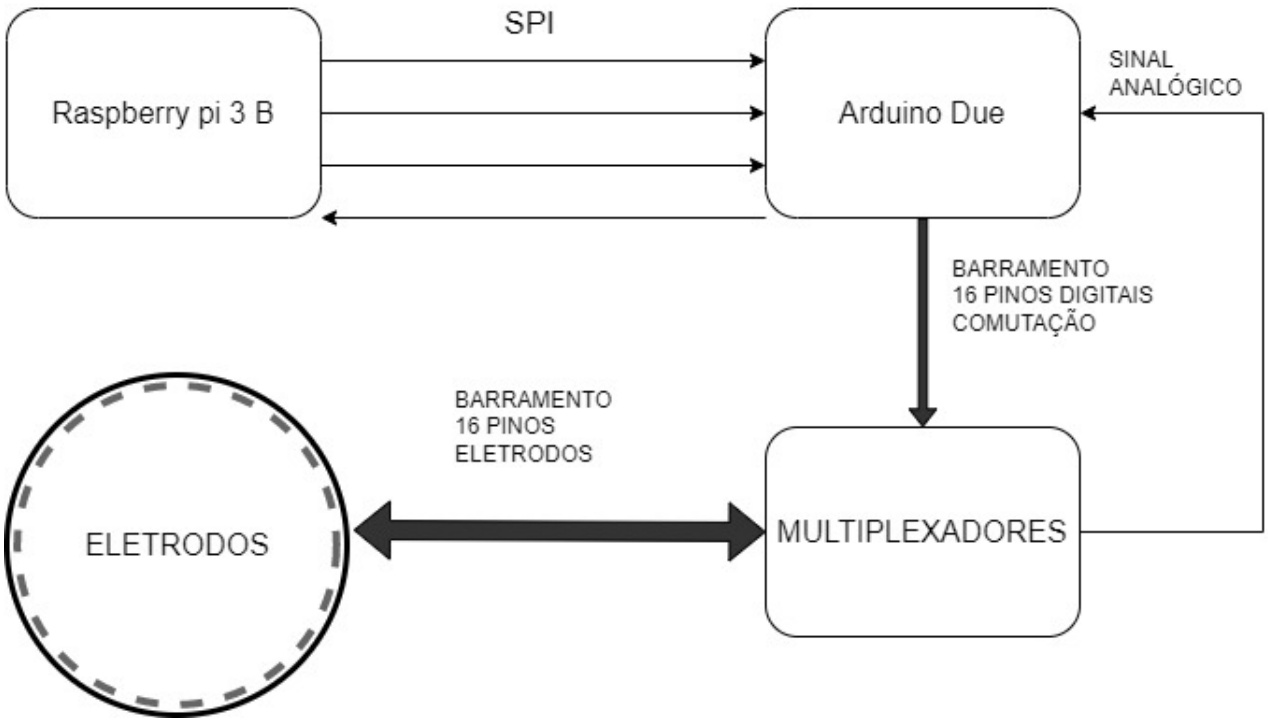


Fonte: O autor.

Para a realização de cada módulo foi proposta a utilização dos seguintes equipamentos: O Raspberry pi 3B como sistema de reconstrução de imagem; Arduino due para a leitura de dados analogicas de 12bits e para as comutações foi utilizado

como base a placa com multiplexadores descrita por Schluchter (2020) , considerado um sistema com 16 eletrodos. Na figura 14 é mostrada a montagem com os barramentos e conexões.

Figura 14 – Modelo de fluxo de dados utilizado



Fonte: O autor.

3.9 Aquisição de dados

A aquisição de dados realiza a integração de todos os módulos, sendo nessa etapa realizado o controle dos pares de eletrodos que serão acionado como emissores de corrente e qual responsável por realizar a leitura da variação de tensão. Cada componente do sistema atua com uma funcionalidade para realizar a leitura, sendo as seguintes definições dadas a cada um:

1. Raspberry pi: gerencia qual dos eletrodos será emissor de corrente e qual realizará a leitura transmitindo o comando por meio da comunicação SPI.

2. Arduino Due: responsável por receber e interpretar o comando recebido por SPI, gerando sinais lógicos em seus pinos com o intuito de setar os multiplexadores. Além disso, é o Arduino que interpreta o valor analógico recebido dos eletrodos após passar pelo sistema de aquisição e encaminha ao Raspberry, também por comunicação SPI.

3. Multiplexadores: ao receberem os sinais digitais, realizam a conexão dos eletrodos seguindo as ligações lógicas do Arduino.

3.9.1 Comandos SPI

Com a utilização do protocolo SPI, foi realizada a integração por meio de um conjunto de comandos utilizando hexadecimais (HEX). Na tabela 2 temos as listas em comandos e suas aplicações no Arduino.

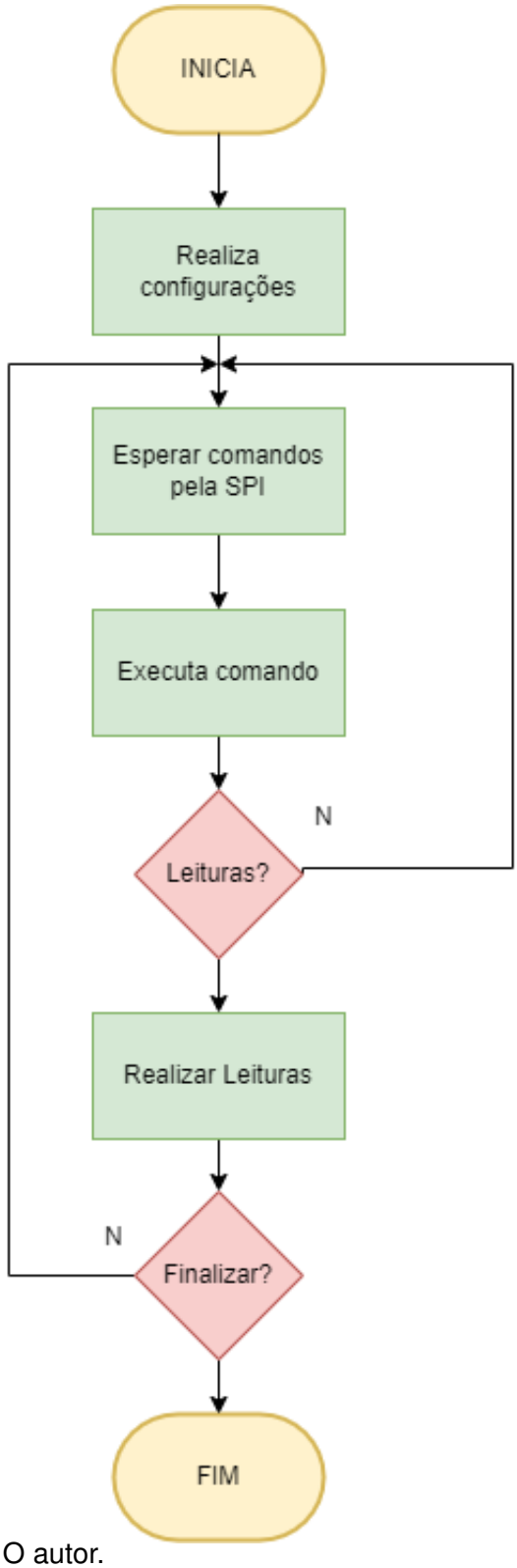
Tabela 2 – Comando o protocolo SPI

Comando	Função
0x11	Setar saída dos Emissores
0x22	Setar eletrodo de Leitura
0x33	Reencaminhar último dado enviado
0x44	Retorna os eletrodos de emissores e corrente
0x55	Valores hexadecimais dos eletrodos de Leitura
0x99	Solicita a leitura do dado da aquisição
0xAA	Inicia aquisição de dados.
0xCC	Confirmação de finalização de aquisição de dados

Fonte: O autor.

Entre os comandos, vale ressaltar o comando 0xCC, ao ser encaminhado pode vir a retornar 2 mensagem sendo 0xBB quando a aquisição ainda estiver sendo realizada, e 0xDD ao finalizar a leitura do eletrodo. O Arduino realiza não somente uma medição, mas sim uma série de medidas a fim de realizar a média de seus valores para obter uma maior precisão de suas leituras. O fluxograma da figura 15 estrutura como o programa do Arduino due atua junto aos comandos SPI.

Figura 15 – Fluxograma programa Arduino due



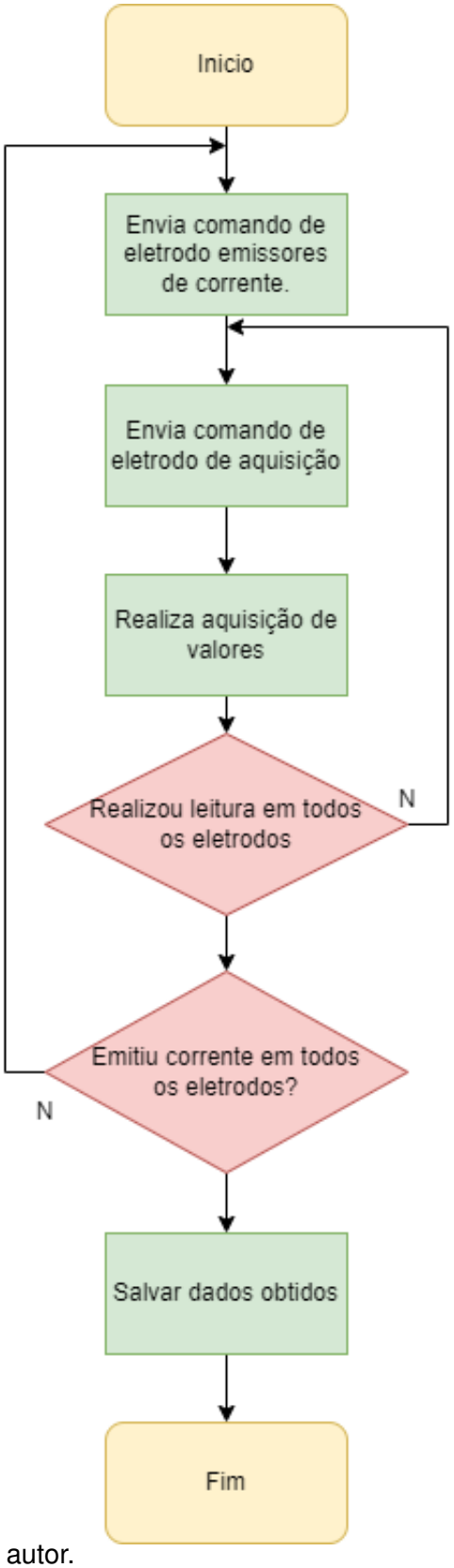
Fonte: O autor.

3.9.2 Gerenciamento eletrodos via Raspberry pi

A indexação dos eletrodos, assim como dar início às leituras de dados, ocorre por meio de um código feito em python rodando no Raspberry PI. O modelo de

indexação e de ordem de comandos é demonstrado no fluxograma da figura 16

Figura 16 – Fluxograma aquisição de dados Raspberry



Fonte: O autor.

O número total de leituras é obtido por meio da quantidade de eletrodos e

métodos de leitura. No processo descrito, foi utilizado o método de eletrodos vizinhos, que tem como objetivo a medição e emissão de corrente por eletrodos em pares adjacentes; a quantidade de leituras pode ser obtida através da Equação 15, sendo que L é a quantidade de leituras e n a quantidade de eletrodos.

$$L = n(n - 3) \tag{15}$$

Como o sistemas de eletrodos possuem no caso da aplicação 16 eletrodos, o processo realiza ao todos 208 leituras dos pares de eletrodos. É importante salientar que muitas vezes será realizada leituras similares, mas que garante a verossimilhança das aquisições

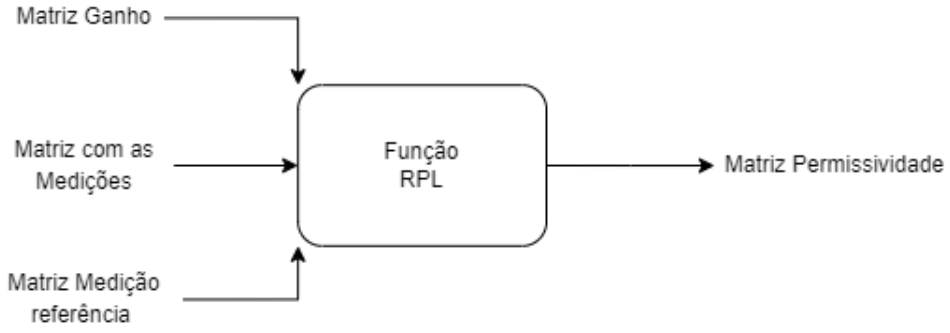
3.10 Algoritmo de resconstrução

Como mostrado na Equação 3 é preciso calcular a variação do meio com base na permissividade, indutância do meio, para isso foram implementados dois algoritmos, o da retroprojeção linear e de Landweber.

3.10.1 Retroprojeção linear

O método da retroprojeção linear, para ser implementado, necessita da entrada de duas matrizes como mostrado na Equação 4, sendo a entrada da matriz sensibilidade e a matriz com os sinais oriundos dos eletrodos, porém na prática foi realizada uma função com entradas de três parâmetros, sendo o terceiro o valor do meio sem a existência de um material que provoca a variação de impedância, os valores da variações de tensão em seu estado natural. A Figura 17 mostra o fluxo de entrada de dados.

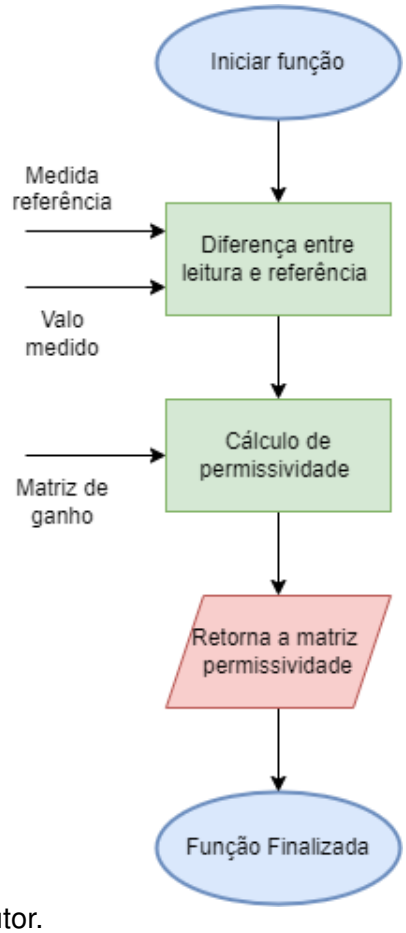
Figura 17 – Entrada de dados LBP



Fonte: O autor.

Para a realização do cálculo foi utilizado a biblioteca *numpy*, que realiza operações entre matrizes de forma dinâmica por meio dos métodos nela implementados. O fluxo interno nessa função foi realizado como mostrado na figura 18.

Figura 18 – Fluxograma cálculo Retroprojeção linear

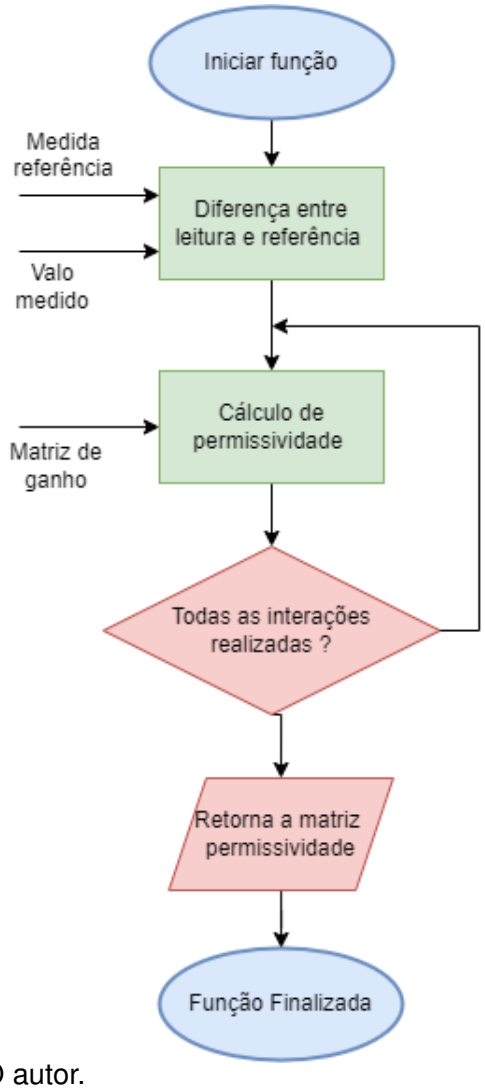


Fonte: O autor.

3.10.2 Método interativo de Landwaber

A implementação do algoritmo iterativo de Landweber utilizou as mesmas bibliotecas da retroprojeção linear. A Figura 19 mostra o fluxograma da implementação do algoritmo iterativo de Landweber descrito da Equação 12.

Figura 19 – Fluxograma cálculo Landweber



Fonte: O autor.

3.11 Reconstrução da imagem

A reconstrução da imagem se dá pela entrada da matriz permissividade. Essa matriz é associada uma lista de triângulos dicretizada por seus vértices, sendo assim cada triângulo recebe um valor de permissividade que corresponde ao ponto do meio onde foi realizada a leitura.

Nessa etapa já é possível dizer que possui todos os aspectos da tomografia, porém não fica clara o entendimento somente por valores números, para isso foi adaptado o algoritmo de Fan, Gao e Wang (2011). Inicialmente utilizado para a criação de representações gráficas de tomografia por capacitância, porém será utilizado nesse contexto para a representação da indutância.

O algoritmo prevê cálculos diferentes dos valores obtidos normalizados de permissividade presentes na imagem, sendo que nesta abordagem será tomada a impedância como valor normalizado, transformado em valores para cores RGB (*Red*

Green Blue) , vermelho, verde e azul respectivamente.

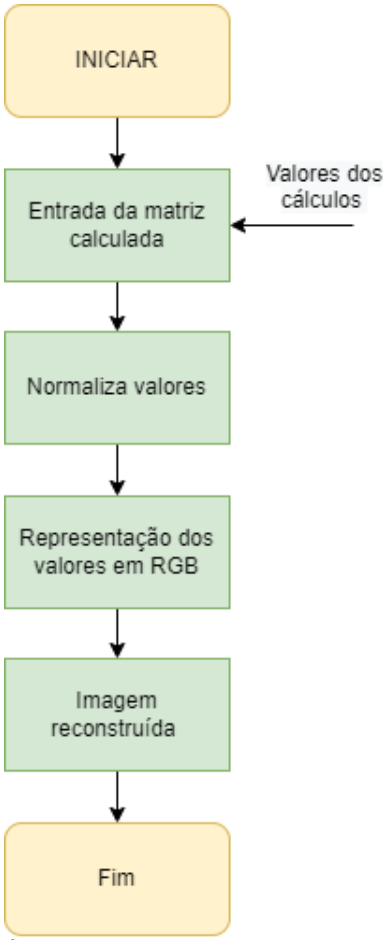
$$\begin{cases} R = 255 \\ G = \frac{1-\sigma}{1-0.55}, \ se \ 0.55 \leq \sigma \leq 1 \\ B = 0 \end{cases} \tag{16}$$

$$\begin{cases} R = \frac{\sigma-0.45}{0.55-0.45} \cdot 255 \\ G = 255, \ se \ 0.45 \leq \sigma \leq 0.55 \\ B = \frac{0.55-\sigma}{0.55-0.45} \cdot 255 \end{cases} \tag{17}$$

$$\begin{cases} R = 0 \\ G = \frac{\sigma}{0.45} \cdot 255, \ se \ \sigma \leq 0.45 \\ B = 255. \end{cases} \tag{18}$$

Com os valores de cores seguindo o programa por meio da biblioteca *Opencv* realiza a reconstrução da imagem associando a cada triângulo o valor de cor RGB com base em seus cálculos previamente realizados.O fluxograma da figura 20 mostra as etapas até a reconstrução.

Figura 20 – Fluxograma reconstrução da imagem



Fonte: O autor.

3.12 Discretização da imagem

Para realizar a reconstrução da imagem, será preciso discretizar o espaço a ser realizada a reconstrução da imagem em espaços menores a fim de representar os objetos, sendo utilizada malha de triângulos interligados com o objetivo de representar o meio de interesse .

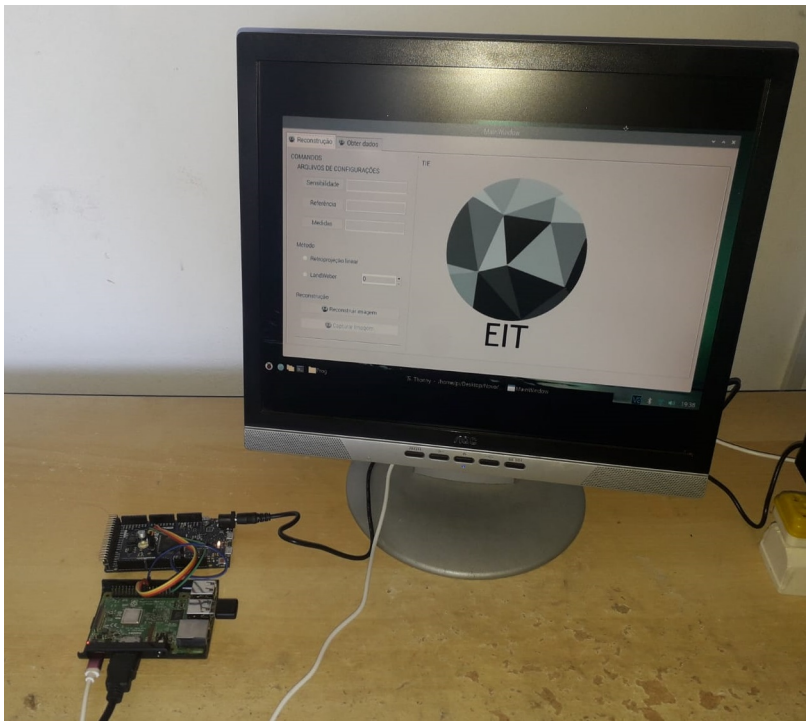
A discretização de triângulos é carregada por um conjunto de 3 pontos interligados para criar um triângulo, sendo que esse triângulo possui pontos compartilhados com os triângulos ao seu redor, construindo uma representação 2D do meio em que estão sendo realizadas as leituras. O espaço a ser discretizado será circular a fim de representar um sistema de aquisição de tomografia por impedância circular.

4 APRESENTAÇÃO DOS RESULTADOS

Neste capítulo serão apresentados os resultados do desenvolvimento do código descrito nos capítulos anteriores. Será apresentado o resultado final da implementação assim como os testes de validação executados e os resultados obtidos.

Para a realização dos testes foi realizada a seguinte montagem dos componentes, no caso, o Raspberry Pi, Arduino e uma tela para apresentar o resultados. A montagem é mostrada na Figura 21.

Figura 21 – Montagem para testes



Fonte: O autor.

4.1 Interface com o usuário

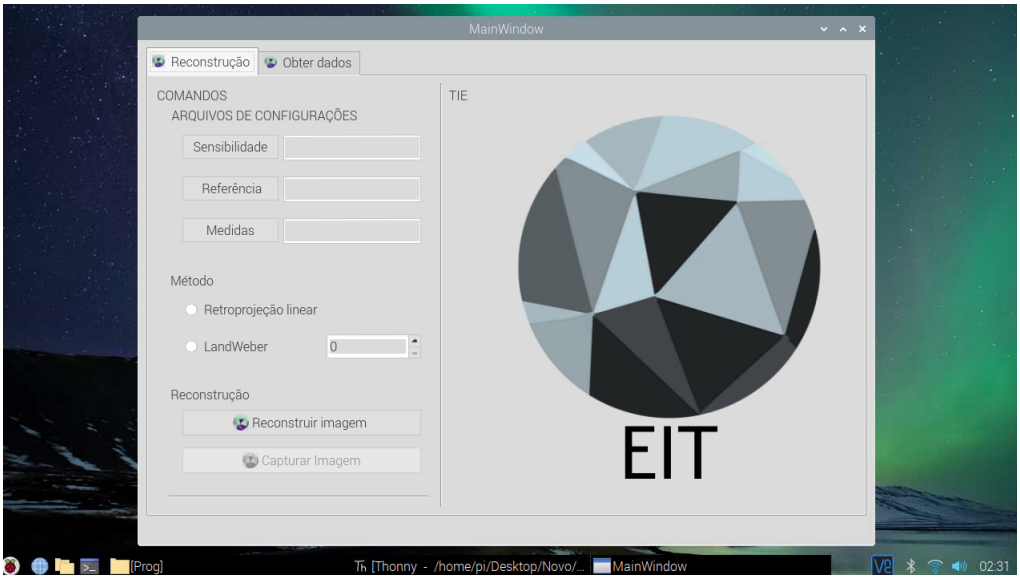
A interface do usuário foi construída levando em conta as necessidades envolvidas para a reconstrução da imagem, podendo citar os seguintes pontos, lembrando que deveria ser compatível com o sistema operacional do Raspberry pi 3 B fundamentado em linux:

- Entrada de matriz sensibilidade;
- Entrada vetor de tensão referência;
- Entrada vetor de tensão medidas;
- Possibilitar salvar a imagem obtida;

- Escolha do método de reconstrução (Retroprojeção ou Landweber);
- Realizar aquisição de sinais e salvar arquivos CSV.

Com esses aspectos, foi desenvolvida a seguinte interface com o usuário que é apresentada na Figura 22 com o programa rodando já no sistema operacional do Raspberry pi, no caso em questão no sistema raspbian.

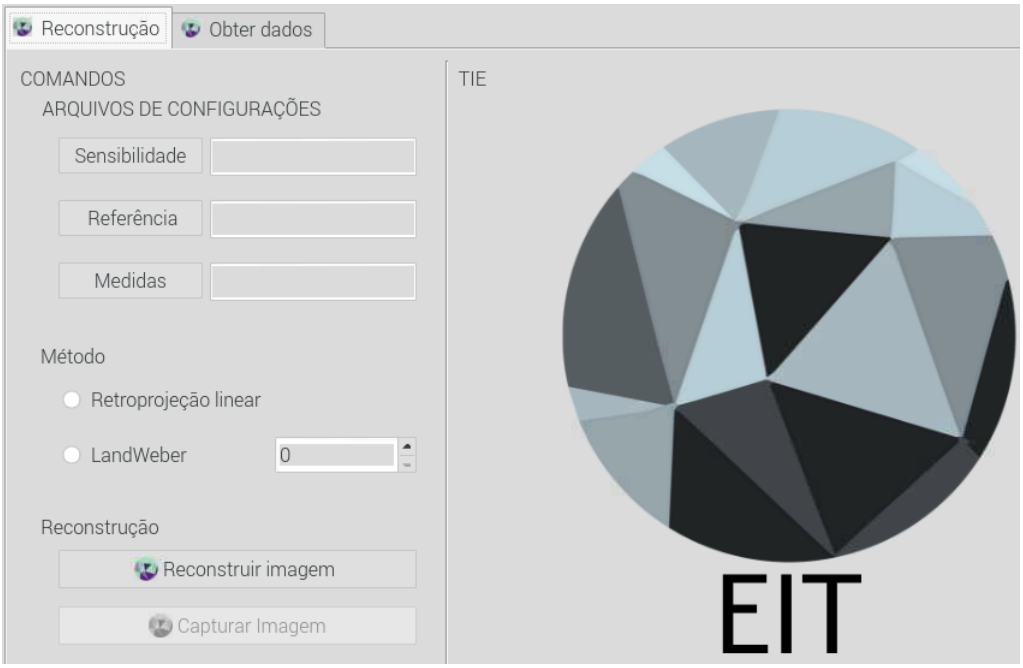
Figura 22 – Interface em funcionamento no raspbian



Fonte: O autor.

De forma mais detalhada, a interface foi dividida em duas abas, sendo a mostrada na Figura 22 a principal, onde oé mostrado ao usuário a imagem reconstruida. De forma mais detalhada, a Figura 23, mostra a aplicação possibilitando ver os pontos para carregar as matrizes e vetores necessários, assim como espaço para escolher o método desejado, espaço para mostrar a imagem e ferramenta de captura para salvar a imagem.

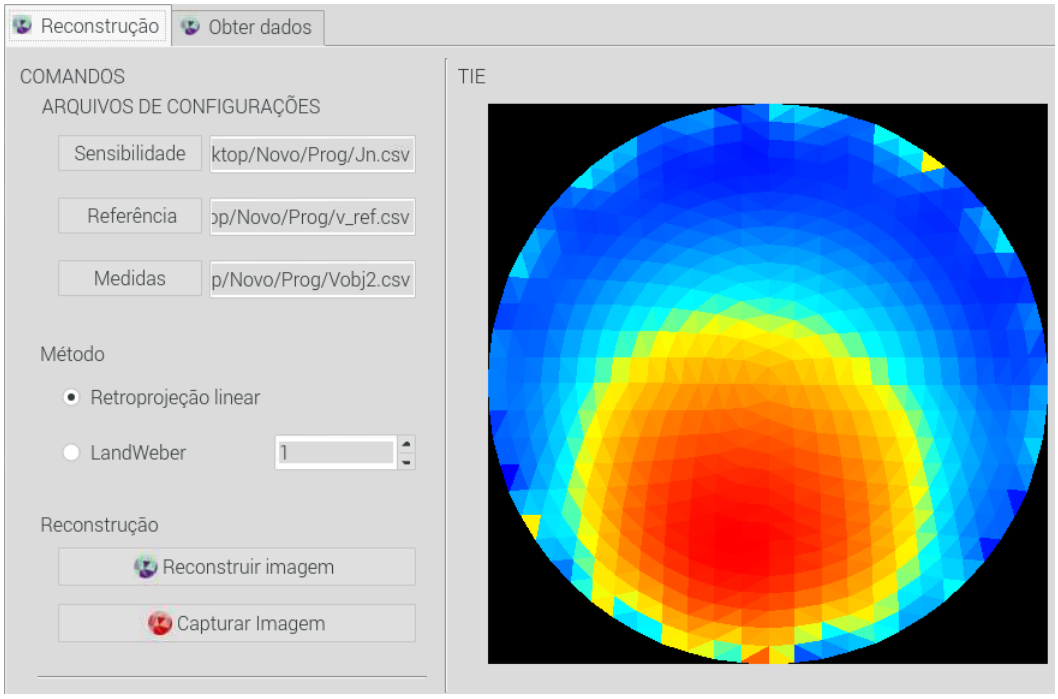
Figura 23 – Interface principal



Fonte: O autor.

Ao realizar todas as configurações e escolher o método de reconstrução, a imagem, mostrada na tela. A Figura 24 mostra a atuação do programa com a reconstrução da imagem com o método da retroprojeção linear.

Figura 24 – Demonstração de reconstrução

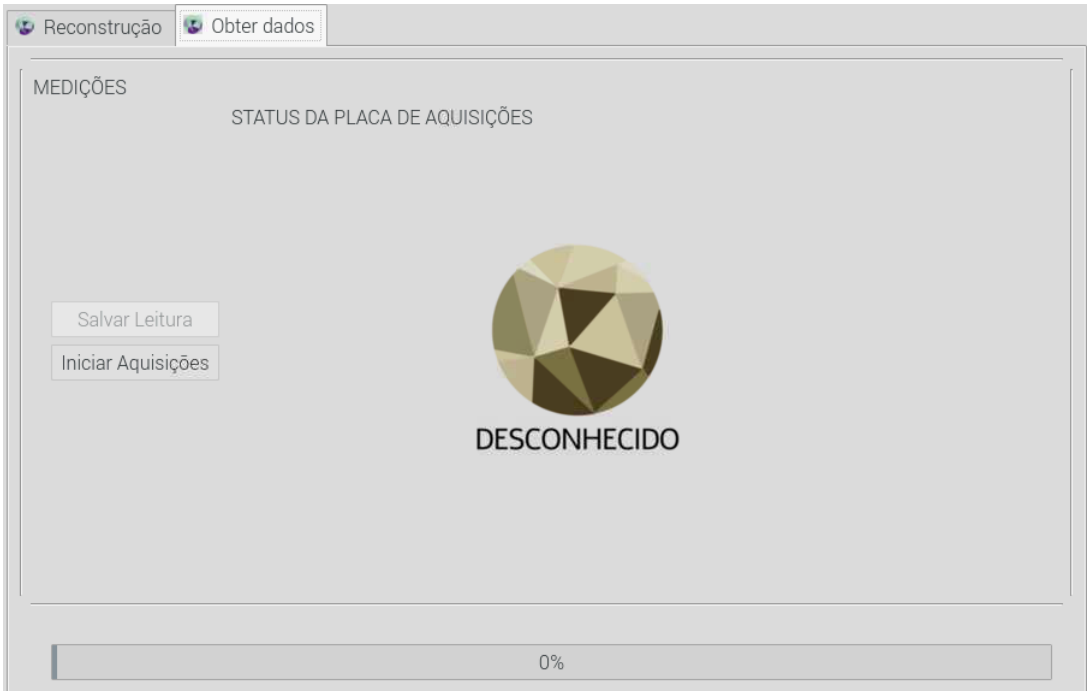


Fonte: O autor.

A aba "Obter dados", é responsável por realizar as aquisições das leituras do Arduino due, após receber todos os dados salva os valores em um vetor usando o

arquivo CSV. A Figura 25 mostra a aba de obter dados.

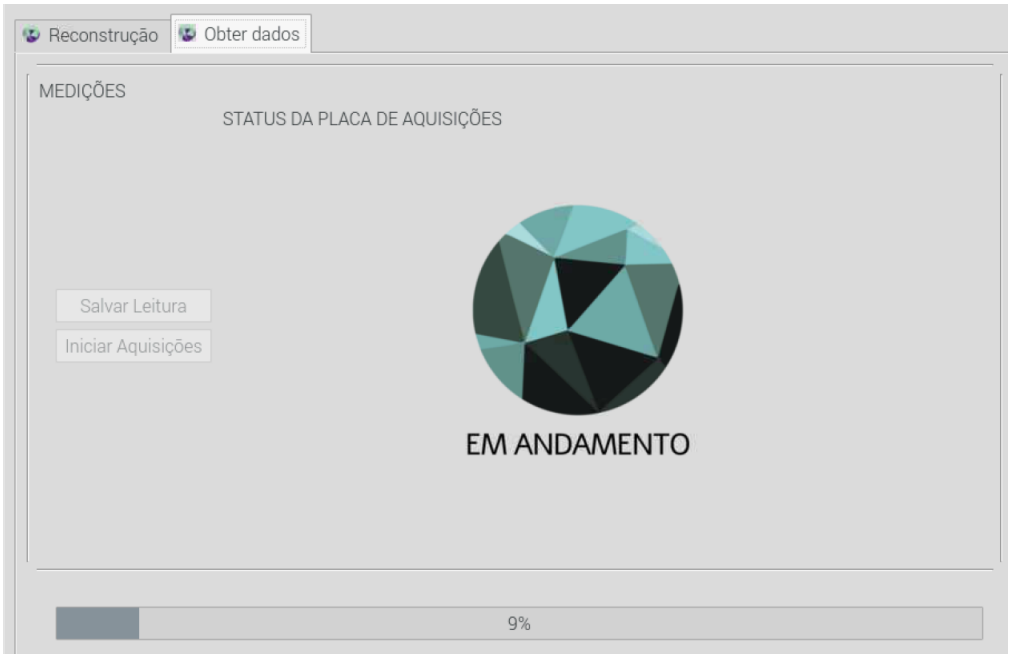
Figura 25 – Interface Obter dados



Fonte: O autor.

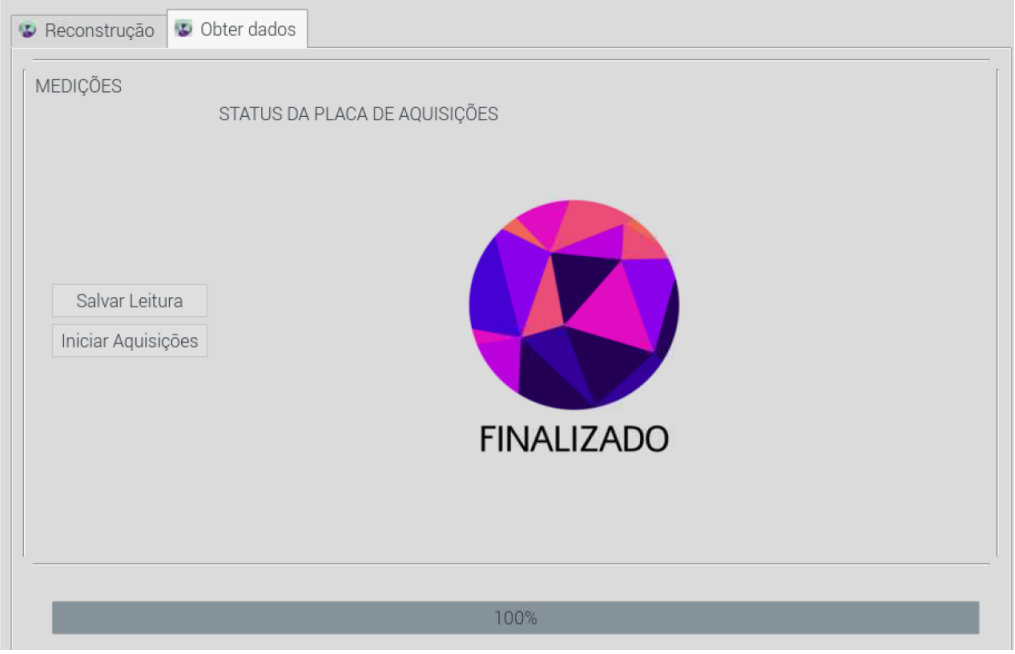
Essa aba possui variações ao iniciar a realização de aquisições para auxiliar o usuário a identificação, sendo mostrado na tela 2 variações: uma ao iniciar as leituras , mostrado na Figura 26; e outra ao finalizar as leituras mostrado na Figura 27, liberando a possibilidade de salvar o arquivo de aquisições.

Figura 26 – Leitura em andamento



Fonte: O autor.

Figura 27 – Leitura finalizada



Fonte: O autor.

4.2 Teste de acionamento dos multiplexadores

Para realizar a leitura de dados como descrito nos capítulos 3 e 4, foi implementada a lógica de acionamento dos eletrodos por meio da comutação das saídas.

Para esses testes, foi utilizado um multímetro em escala de tensão de 20 volts, a fim de identificar a comutação de dados nos pinos de saída do Arduino, sendo que desses testes foi obtida a comutação dos pinos, tendo em vista que durante os testes os pinos comutam de 0 a 3,0 volts já que o Arduino due opera com níveis de tensão lógica de acionamento de 3,3 volts.

4.3 Testes de aquisição

Como intuito de validar o sistema de aquisições que interage entre Raspberry e Arduino due, foi realizado os testes de aquisição, tendo em vista que os valores seriam fornecidos com sinais 12 bits, sendo assim teria composto por 4096 valores possíveis de nível lógico de tensão, variando entre os valores inteiros de 0 a 4095, fazendo correlação entre os valores de tensão de 0 a 3,3 volts. Dessa forma para chegar ao valor de tensão dado a entrada desses valores lógicos é necessário utilizar a Equação 19.

$$V(x) = \frac{3,3}{4095} \cdot x$$

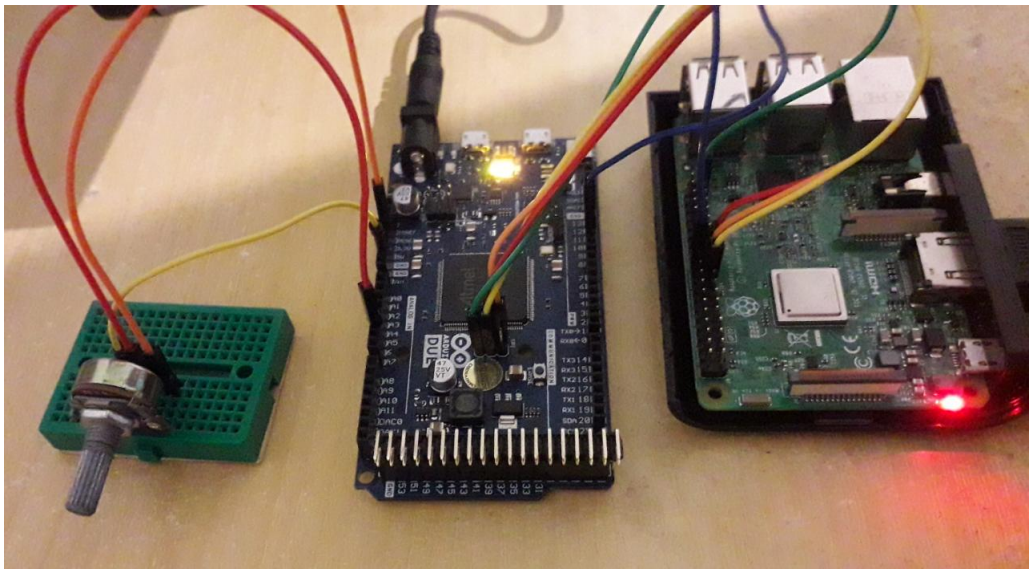
(19)

Sendo assim, foi realizado testes de aquisição realizando a leitura de tensão conectados ao GND, sendo esperado o nível lógico de tensão esperado igual a 0. Como resultado foi obtidas 208 leituras com nível lógico 0, logo temos 0 V

Em seguida, foi conectado a leitura a tensão de 3,3 V, dessa forma obtendo como resultado os valores de 208 leituras, no entanto, com o nível lógico de 4095 e algumas medições com o valor de 4094.

Para validar outros valores, foi conectado um potenciômetro ao circuito, com o intuito de variar a tensão na entrada analogica como mostrada na Figura 28. Com o auxílio de um multímetro da marca minipa modelo ET-1002 , foi lida a variação de tensão fornecida a entrada analógica pelo potenciômetro sendo lido como variação de tensão o valor de 2,00V, esperando assim a leitura de níveis lógicos de tensão igual 2482 segundo a Equação 19.

Figura 28 – Montagem tensão variável



Fonte: O autor.

Como resultado das medidas, foram obtidos 208 valores. Esses valores no teste sofreram uma pequena variação sendo obtido valores que variam entre medidas lógicas entre 2540 e 2542, variação pequena, porém com um valor diferente de nível lógico de tensão esperado considerando o valor medido pelo multímetro. Na tabela 3 é mostrado os valores de tensão segundo a Equação 19, para os niveis lógicos de 2540,2541 e 2542 obtidos pelas leituras.

O erro existente pode ter sido ocasionado pela matriz de contato, onde foram realizadas a leitura de entradas do sinal analógico. Como foram realizadas 208 leituras todos elas esperando obter o mesmo valor, foi calculado o desvio padrão com o intuito

Tabela 3 – Tabela de tensões lidas

Tensão medida	Tensão referência	erro
2,046	2,00	0,046
2,047	2,00	0,047
2,048	2,00	0,048

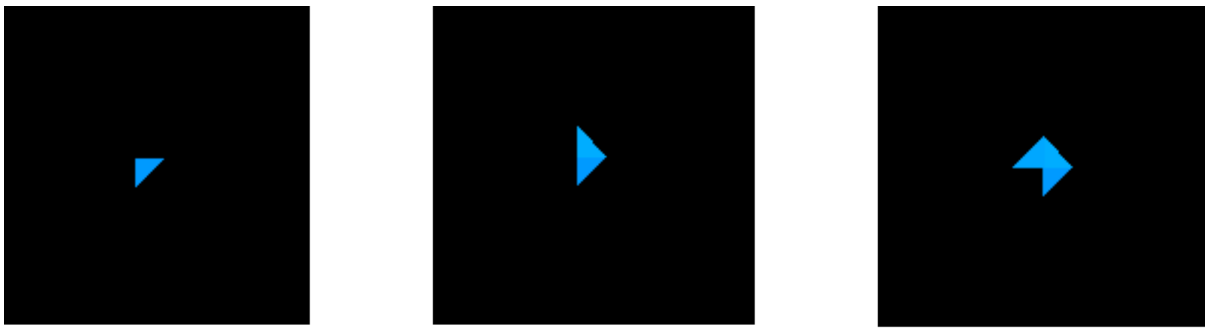
Fonte:O autor

de verificar a uniformidade dos dados, efetuando o desvio padrão das 208 medidas com os níveis lógicos.

4.4 Representação tomográfica

A representação da imagem tomográfica será por meio da imagem discretizada de uma circunferência, utilizando de 1024 triângulos, cada triângulos recebe uma codificação de cor RGB , com o intuito de representar a impedância. O conjunto de triângulos é construído por meio de 3 pontos, sendo que esses pontos são carregados ao programa por meio de uma arquivo CSV, a imagem é construída por meio construção da imagem ao carregar os pontos.

Figura 29 – Etapas de criação da malha (a) Primeiro triângulo (b) Segundo triângulo (c) Terceiro triângulo



Fonte: O autor.

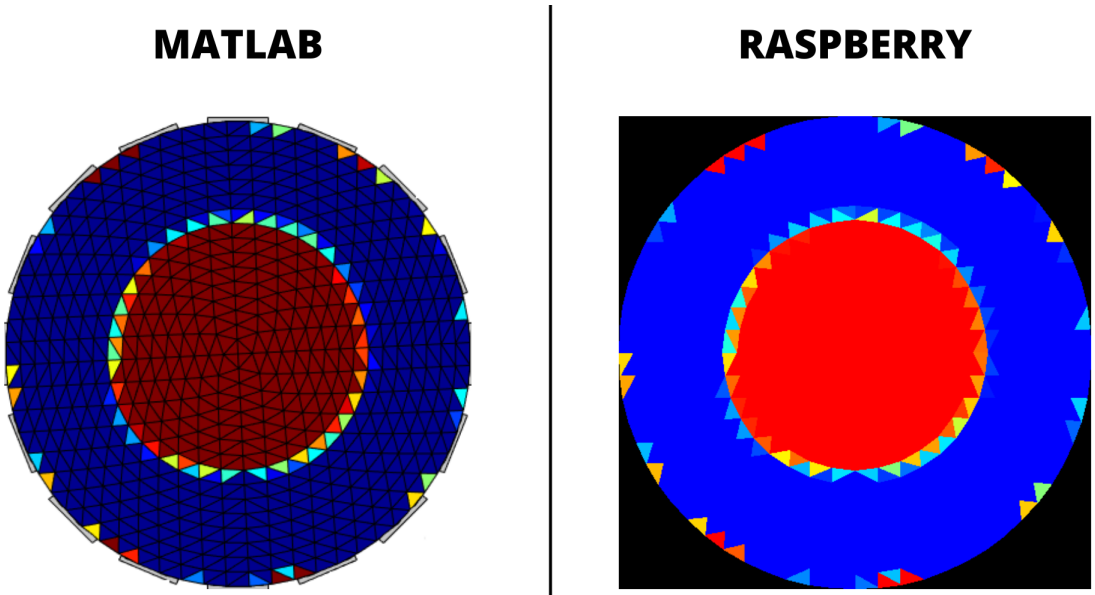
A malha será a responsável por realizar a representação do espaço e das variações de impedância pela variação de cor de cada um dos triângulos.

4.5 Reconstrução da imagem

Os testes de reconstrução da imagem foram executados por meio da utilização de 2 vetores de medidas obtidos anteriormente por meio de experimentação, sendo que a fim de comparação foram implementados os algoritmos de retroprojeção linear e de landweber no matlab, a fim de realizar a comparação dos resultados alcançados com a implementação do programa. Nas Figuras 30 até 33 temos a comparação da

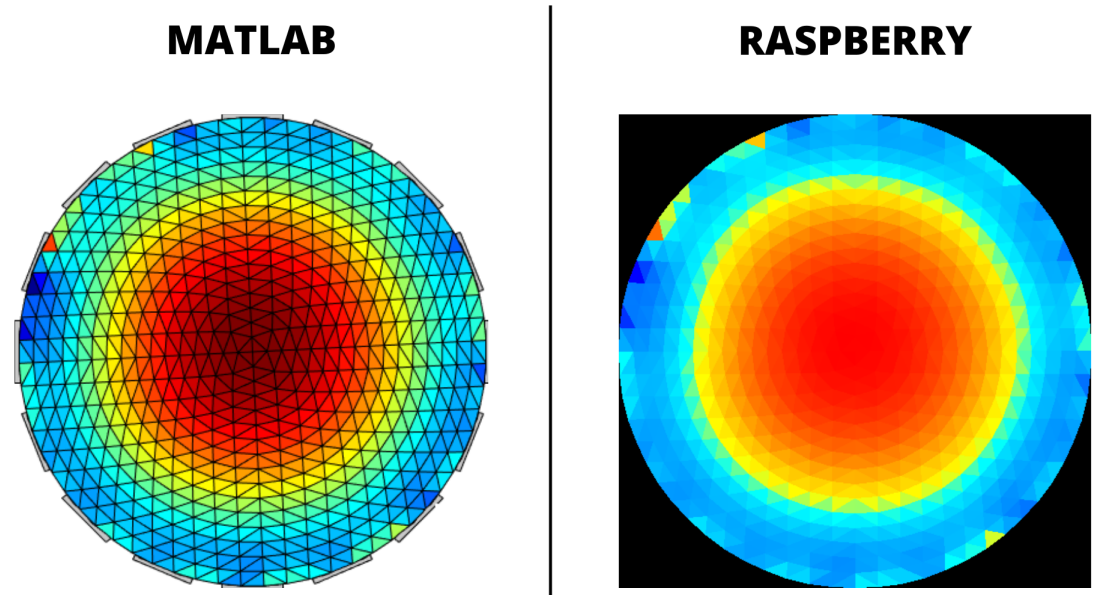
implementação dos métodos e das imagens obtidas por meio do matlab e as geradas pelo programa embarcado ao Raspberry, para 2 vetores distintos.

Figura 30 – Comparação vetor objeto 1 com Landwaber 100 interações



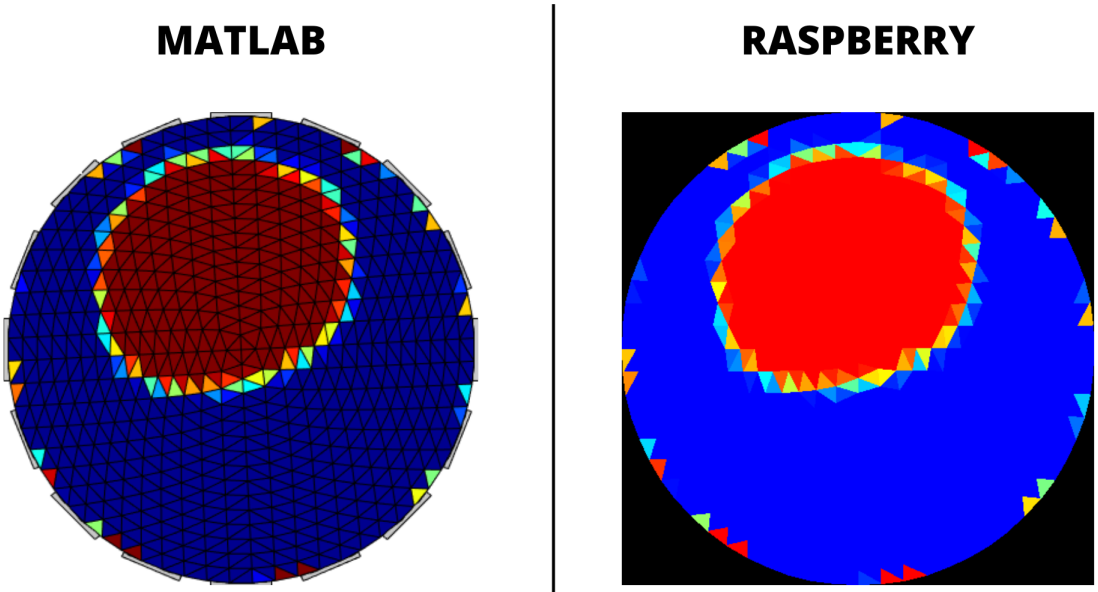
Fonte: O autor.

Figura 31 – Comparação vetor objeto 1 com retroprojeção linear



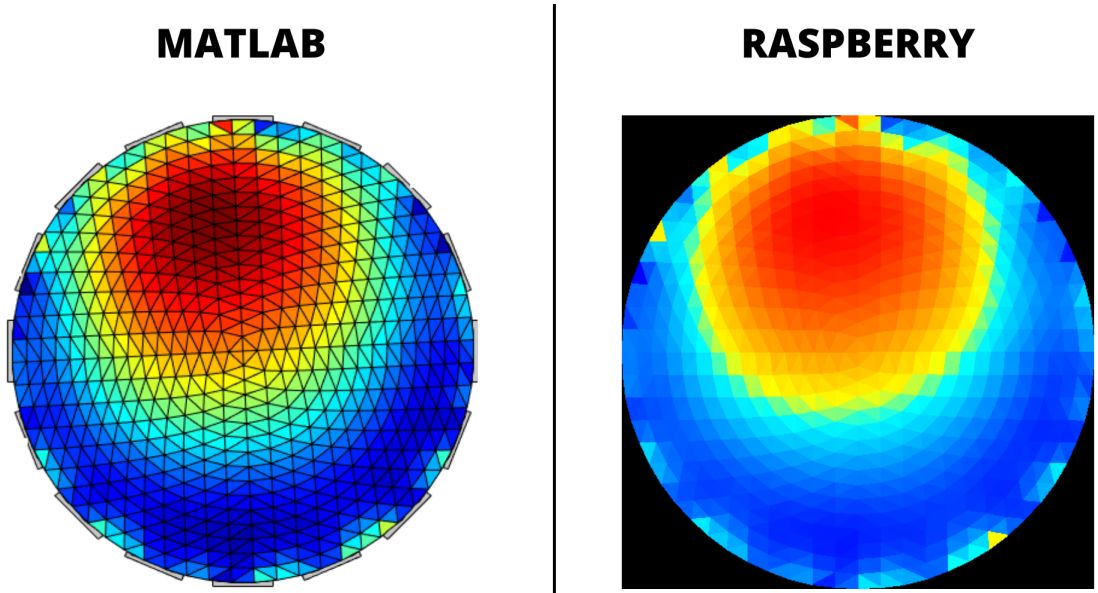
Fonte: O autor.

Figura 32 – Comparação vetor objeto 2 com Landwaber 100 interações



Fonte: O autor.

Figura 33 – Comparação vetor objeto 2 com retroprojeção linear



Fonte: O autor.

4.6 Análise e discussão dos resultados

Ao analisar as imagens é possível identificar a verossimilhança entre a aplicação no matlab e no programa embarcado no Raspberry para a reconstrução da imagem, mostrando a capacidade do programa em realizar a reconstrução da imagem.

Vale ressaltar que para a aplicação dos métodos, nos testes de execução com o programa embarcado no Raspberry pi , o método com maior velocidade de

execução foi o método da retroprojeção linear. As imagens geradas por esse métodos acabam por não gerar uma convergência na imagem, mostrando espaços externos com as variações de impedância, em comparação com o método de Landweber, as Figuras 33 e 32 mostram de maneira bem acentuada essa variação, enquanto no método de Landweber o objeto possui uma forma mais definida em comparação com a retroprojeção linear.

5 CONSIDERAÇÕES FINAIS

No presente trabalho, foi descrito o método de tomografia por impedância elétrica TIE, trazendo modelos de aquisição, circuitos de aquisição e o foco central do trabalho, os algoritmos utilizados para realizar a reconstrução da imagem, sendo detalhados os passos e conhecimentos necessários, tendo em vista o resultado final, o desenvolvimento de um *software* capaz de realizar a aquisição e reconstrução de imagem de forma a possibilitar sua utilização em um sistema embarcado, no caso em especial no computador de placa única Raspberry pi B 3.

Como resultado, foi desenvolvido um *software* dedicado que possui de forma integrada a capacidade de realizar a aquisição de dados, assim como, a reconstrução de imagens de tomografia TIE, sendo que, é possível optar entre dois métodos de reconstrução de imagem, retroprojeção linear e algoritmo iterativo de LandWeber. Esses dois métodos possuem suas singularidades quanto à resposta final e ao tempo de operação para retorno de respostas. Ambas as propostas apresentaram resultados satisfatórios, tendo em comparação algoritmo implementado em Matlab em um computador pessoal à parte.

Além da reconstrução, foi desenvolvido um método de aquisição utilizando um arduino integrado ao sistema por meio de interface de comunicação SPI. A integração desta segunda placa permite a conexão do programa a placas de aquisição que utilizam sistemas de multiplexação para 16 eletrodos que realizam o processo de aquisição dos eletrodos vizinhos.

Esse desenvolvimento valida a possibilidade de desenvolver sistemas que possam auxiliar em pesquisas ligadas a tomografia TIE, assim como, levanta a possibilidade da utilização de placas como a descrita, para implementação no âmbito da medicina.

5.1 Sugestões para trabalhos futuros

Para estudos e trabalhos de desenvolvimento futuros que abordem o assunto de forma similar, como sugestão, trazer a implementação de ferramentas de análise da imagem com os métodos presentes na biblioteca OpenCV, de forma a além de realizar a reconstrução da imagem, possibilitar análises mais complexas.

Além disso, sugere-se a implementação de outros modelos de aquisição, como o modelo de aquisição do eletrodo oposto, assim como, desenvolver tanto uma aplicação de *software* como *hardware* que possibilite a utilização de forma paralelamente, ou seja, recolher as informações e reconstruir na tela de forma continua, sem a necessidade de salvar um arquivo CSV previamente.

REFERÊNCIAS

ZHAOYAN Fan and Robert X. Gao and Jinjiang Wang. 22

ANAND *et al.* Design and implementation of a high speed serial peripheral interface. In: . IEEE, 2014. p. 1–3. ISBN 978-1-4799-3543-7. Disponível em: http://ieeexplore.ieee.org/document/6838431/. 29

ARDUINO. *Pinout Diagram*. 2022. Disponivel em https://docs.arduino.cc/. Acesso em 26 de junho de 2022. 28, 29

ARELLANO, F. Z. *et al.* Development of a portable, reliable and low-cost electrical impedance tomography system using an embedded system. *Electronics (Switzerland)*, MDPI AG, v. 10, p. 1–24, 1 2021. ISSN 20799292. 18, 20, 21

ARIS, W. Design of low-cost and high-speed portable two-dimensional electrical impedance tomography (eit). *International Journal of Engineering Technology*, v. 7, p. 6458–6463, 2018. Disponível em: www.sciencepubco.com/index.php/IJET. 21

BARROS, T. R. de. *Simulação numérica de um sensor de tomografia capacitiva para análise de escoamento bifásico ar-água*. 2011. Disponível em: http://acervus.unicamp.br/index.asp?codigo_sophia=795792. 34

BOLFE, V. *et al.* Comportamento da impedância elétrica dos tecidos biológicos durante estimulação elétrica transcutânea. *Revista Brasileira de Fisioterapia*, v. 11, 4 2007. ISSN 1413-3555. Disponível em: http://www.scielo.br/scielo.php?script=sci_arttext&pid=S1413-35552007000200011&lng=pt&nrm=iso&tlng=pt. 19

BURR-BROWN. *Precision Unity Gain DIFFERENTIAL AMPLIFIER INA105 DESCRIPTION*. 2022. Disponivel em https://www.raspberrypi.com/documentation/computers/os.html. Acesso em 26 de junho de 2022. Disponível em: https://pdf1.alldatasheetpt.com/datasheet-pdf/view/56670/BURR-BROWN/INA105.html. 32

CARVALHO, A. C. P. *et al.* História da tomografia computadorizada história da radiologia descritores. *ACP Rev Imagem*, v. 29, p. 61–66, 2007. 17

DAVIDSON, J. L. *et al.* Three-dimensional electrical impedance tomography applied to a metal-walled filtration test platform. *Measurement Science and Technology*, Institute of Physics Publishing, v. 15, p. 2263–2274, 2004. 22

DEVELOPERS, N. *What is NumPy?* 2022. Disponivel em https://numpy.org/doc/stable/user/whatisnumpy.html. Acesso em 26 de junho de 2022. 26

FAN, Z.; GAO, R. X.; WANG, J. Virtual instrument for online electrical capacitance tomography. In: SILVIU, F. (Ed.). *Practical Applications and Solutions Using LabVIEW Software*. Rijeka: IntechOpen, 2011. cap. 1. Disponível em: https://doi.org/10.5772/19523. 41

FARIA, J. J. R.; FARIA, A. R. de. Exploring the opencv library for image processing in long-pulse thermography. *Materials Research*, Universidade Federal de Sao Carlos, v. 24, 2021. ISSN 19805373. 26

INCA. *Exposição à Radiação*. 2022. Disponivel em https://www.inca.gov.br/causas-e-prevencao/prevencao-e-fatores-de-risco/exposicao-a-radiacao. Acesso em 25 de agosto de 2022. 15

JIA, J.; WANG, H.; MILLINGTON, D. Electrical resistance tomography sensor for highly conductive oil-water two-phase flow measurement. *IEEE Sensors Journal*, Institute of Electrical and Electronics Engineers Inc., v. 17, p. 8224–8233, 12 2017. ISSN 1530437X. 17

JIA, X. *et al.* The capacitance tomography image reconstruction algorithm based on sensitivity field sensitivity matrix. In: . [S.l.]: IOP Publishing Ltd, 2021. v. 2005. ISSN 17426596. 21, 22

LANDWEBER, L. An iteration formula for fredholm integral equations of the first kind. *American Journal of Mathematics*, v. 73, p. 615, 7 1951. ISSN 00029327. Disponível em: https://www.jstor.org/stable/2372313?origin=crossref. 22

LEENS, F. An introduction to i ² c and spi protocols. *IEEE Instrumentation Measurement Magazine*, v. 12, p. 8–13, 2 2009. ISSN 1094-6969. Disponível em: http://ieeexplore.ieee.org/document/4762946/. 30, 31

MORSCH, J. A. *COMO É FEITO UMA TOMOGRAFIA? PARA QUE SERVE?* 2022. Disponivel em https://telemedicinamorsch.com.br/blog/como-e-feito-uma-tomografia. Acesso em 17 de janeiro de 2022. 15

MOTA, F. R. M. D. *TOMOMETRIA CAPACITIVA APLICADA À MEDIÇÃO DE FRAÇÃO DE ÁGUA EM ESCOAMENTOS BIFÁSICOS*. 2015. Disponível em: https://repositorio.ufsc.br/xmlui/handle/123456789/169641. 19, 22, 23

OPENCV. *About*. 2022. Disponivel em https://opencv.org/about/. Acesso em 26 de junho de 2022. 26

PALLARO, M. A. P. *AVALIAÇÃO DE DESEMPENHO DE ALGORITMOS CRIPTOGRÁFICOS EM MICROCONTROLADORES ARDUINO*. 2019. Disponível em: http://repositorio.utfpr.edu.br/jspui/handle/1/9263. 28

PERLIN, L. P.; PINTO, R. C. A. Tomografia ultrassônica em concreto. *Revista IBRACON de Estruturas e Materiais*, v. 6, p. 246–269, 4 2013. ISSN 1983-4195. Disponível em: http://www.scielo.br/scielo.php?script=sci_arttext&pid=S1983-41952013000200006&lng=pt&tlng=pt. 17

RASCHKA, S.; PATTERSON, J.; NOLET, C. Machine learning in python: Main developments and technology trends in data science, machine learning, and artificial intelligence. *Information (Switzerland)*, MDPI AG, v. 11, 4 2020. 26

RASPBERRYPI. *Raspberry Pi OS*. 2022. Disponivel em https://www.raspberrypi.com/documentation/computers/os.html. Acesso em 26 de junho de 2022. 25

RIBEIRO, D. E. Protótipo de um tomógrafo de impedância de baixo custo baseado em open-hardware. *Universidade Federal de Pernambuco*, 8 2017. Disponível em: https://repositorio.ufpe.br/handle/123456789/31146. 15

RYMARCZYK, T. Prototype of miniature electrical impedance tomograph smarteit cooperating with raspberry pi platform; prototype of miniature electrical impedance tomograph smarteit cooperating with raspberry pi platform. 2019. 20

SCHLUCHTER, A. *A 16-Channel Electrical Impedance Tomography System Using the Red Pitaya*. 2020. Disponível em: https://escholarship.org/uc/item/7969r8kb. 31, 32, 33, 35

SINGH, G. *et al.* A low-cost portable wireless multi-frequency electrical impedance tomography system. *Arabian Journal for Science and Engineering*, Springer Verlag, v. 44, p. 2305–2320, 3 2019. ISSN 21914281. 15, 17, 21, 25

SOHAL, H. *et al.* Electrical impedance imaging system using fpgas for flexibility and interoperability. *BioMedical Engineering OnLine*, v. 13, p. 126, 2014. Disponível em: http://www.biomedical-engineering-online.com/content/13/126. 21

TAKA, E. N. Transvarredura por bioimpedância: uma ferramenta para a detecção precoce do câncer de mama em mulheres jovens. 2008. Disponível em: http://www.peb.ufrj.br/teses/Tese0082_2008_12_18.pdf. 24

VALLAT, R. statistics in python. *Journal of Open Source Software*, The Open Journal, v. 3, p. 1026, 11 2018. 26

YANG, W. Q.; PENG, L. *Image reconstruction algorithms for electrical capacitance tomography*. 2003. 1-13 p. Disponível em: http://iopscience.iop.org/0957-0233/14/1/201. 21, 22, 23

APÊNDICES

APÊNDICE A – CÓDIGO RASPBERRY PI

```
1
2 from PyQt5 import QtCore, QtGui, QtWidgets
3 from EIT.AppTie import Ui_MainWindow
4
5
6 if __name__ == "__main__":
7     import sys
8     app = QtWidgets.QApplication(sys.argv)
9     QMainWindow = QtWidgets.QMainWindow()
10    ui = Ui_MainWindow()
11    ui.setupUi(QMainWindow)
12    QMainWindow.show()
13    sys.exit(app.exec_())
```

```
16
17 from PyQt5 import QtCore, QtGui, QtWidgets
18 from PyQt5.QtWidgets import QMessageBox
19 import numpy
20 import EIT.eit as TIE
21 from EIT import Comunica
22 import time
23 import cv2
24 import csv
25 import threading
26
27
28
29 class Ui_MainWindow(object):
30     def setupUi(self, MainWindow):
31         MainWindow.setObjectName("MainWindow")
32         MainWindow.resize(917, 629)
33         MainWindow.setStyleSheet("background-color: rgb(217, 217, 217);")
34         self.centralwidget = QtWidgets.QWidget(MainWindow)
35         self.centralwidget.setObjectName("centralwidget")
36         self.horizontalLayout_2 = QtWidgets.QHBoxLayout(self.centralwidget)
37         self.horizontalLayout_2.setObjectName("horizontalLayout_2")
38         self.Aplic = QtWidgets.QTabWidget(self.centralwidget)
39         self.Aplic.setContextMenuPolicy(QtCore.Qt.PreventContextMenu)
40         self.Aplic.setAutoFillBackground(False)
41         self.Aplic.setObjectName("Aplic")
42         self.tab = QtWidgets.QWidget()
43         self.tab.setObjectName("tab")
44         self.horizontalLayout = QtWidgets.QHBoxLayout(self.tab)
45         self.horizontalLayout.setObjectName("horizontalLayout")
46         self.groupBox = QtWidgets.QGroupBox(self.tab)
47         self.groupBox.setMinimumSize(QtCore.QSize(350, 0))
48         self.groupBox.setMaximumSize(QtCore.QSize(300, 16777215))
49         self.groupBox.setObjectName("groupBox")
50         self.verticalLayout = QtWidgets.QVBoxLayout(self.groupBox)
51         self.verticalLayout.setObjectName("verticalLayout")
52         self.groupBoxArquivos = QtWidgets.QGroupBox(self.groupBox)
```

```
53          sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Fixed, QtWidgets.
    QSizePolicy.Fixed)
54          sizePolicy.setHorizontalStretch(0)
55          sizePolicy.setVerticalStretch(0)
56          sizePolicy.setHeightForWidth(self.groupBoxArquivos.sizePolicy().
    hasHeightForWidth())
57          self.groupBoxArquivos.setSizePolicy(sizePolicy)
58          self.groupBoxArquivos.setMinimumSize(QtCore.QSize(300, 200))
59          self.groupBoxArquivos.setMaximumSize(QtCore.QSize(500, 200))
60          self.groupBoxArquivos.setObjectName("groupBoxArquivos")
61          self.formLayout = QtWidgets.QFormLayout(self.groupBoxArquivos)
62          self.formLayout.setObjectName("formLayout")
63          spacerItem = QtWidgets.QSpacerItem(20, 40, QtWidgets.QSizePolicy.Minimum,
    QtWidgets.QSizePolicy.Expanding)
64          self.formLayout.setItem(0, QtWidgets.QFormLayout.LabelRole, spacerItem)
65          self.Bt_sensibilidade = QtWidgets.QPushButton(self.groupBoxArquivos)
66          self.Bt_sensibilidade.setMinimumSize(QtCore.QSize(120, 0))
67          self.Bt_sensibilidade.setObjectName("Bt_sensibilidade")
68          self.formLayout.setWidget(1, QtWidgets.QFormLayout.LabelRole, self.
    Bt_sensibilidade)
69          self.Line_sensibilidade = QtWidgets.QLineEdit(self.groupBoxArquivos)
70          self.Line_sensibilidade.setObjectName("Line_sensibilidade")
71          self.formLayout.setWidget(1, QtWidgets.QFormLayout.FieldRole, self.
    Line_sensibilidade)
72          spacerItem1 = QtWidgets.QSpacerItem(20, 40, QtWidgets.QSizePolicy.Minimum,
    QtWidgets.QSizePolicy.Expanding)
73          self.formLayout.setItem(2, QtWidgets.QFormLayout.LabelRole, spacerItem1)
74          self.Bt_tense_ref = QtWidgets.QPushButton(self.groupBoxArquivos)
75          self.Bt_tense_ref.setMinimumSize(QtCore.QSize(120, 0))
76          self.Bt_tense_ref.setMaximumSize(QtCore.QSize(200, 16777215))
77          self.Bt_tense_ref.setObjectName("Bt_tense_ref")
78          self.formLayout.setWidget(3, QtWidgets.QFormLayout.LabelRole, self.
    Bt_tense_ref)
79          self.Line_tense_ref = QtWidgets.QLineEdit(self.groupBoxArquivos)
80          self.Line_tense_ref.setObjectName("Line_tense_ref")
81          self.formLayout.setWidget(3, QtWidgets.QFormLayout.FieldRole, self.
    Line_tense_ref)
82          spacerItem2 = QtWidgets.QSpacerItem(20, 40, QtWidgets.QSizePolicy.Minimum,
    QtWidgets.QSizePolicy.Expanding)
83          self.formLayout.setItem(4, QtWidgets.QFormLayout.LabelRole, spacerItem2)
84          self.Bt_medidas = QtWidgets.QPushButton(self.groupBoxArquivos)
85          self.Bt_medidas.setMinimumSize(QtCore.QSize(120, 0))
86          self.Bt_medidas.setObjectName("Bt_medidas")
87          self.formLayout.setWidget(5, QtWidgets.QFormLayout.LabelRole, self.
    Bt_medidas)
88          self.Line_medida = QtWidgets.QLineEdit(self.groupBoxArquivos)
89          self.Line_medida.setObjectName("Line_medida")
90          self.formLayout.setWidget(5, QtWidgets.QFormLayout.FieldRole, self.
    Line_medida)
91          spacerItem3 = QtWidgets.QSpacerItem(20, 40, QtWidgets.QSizePolicy.Minimum,
    QtWidgets.QSizePolicy.Expanding)
92          self.formLayout.setItem(6, QtWidgets.QFormLayout.LabelRole, spacerItem3)
93          self.verticalLayout.addWidget(self.groupBoxArquivos)
94          self.groupBoxMetodo = QtWidgets.QGroupBox(self.groupBox)
95          self.groupBoxMetodo.setObjectName("groupBoxMetodo")
96          self.gridLayout_4 = QtWidgets.QGridLayout(self.groupBoxMetodo)
97          self.gridLayout_4.setObjectName("gridLayout_4")
98          self.radio_landwaber = QtWidgets.QRadioButton(self.groupBoxMetodo)
```

```
99          self.radio_landwaber.setObjectName("radio_landwaber")
100         self.gridLayout_4.addWidget(self.radio_landwaber, 2, 0, 1, 1)
101         self.landIntera = QtWidgets.QSpinBox(self.groupBoxMetodo)
102         self.landIntera.setMaximum(10000)
103         self.landIntera.setObjectName("landIntera")
104         self.gridLayout_4.addWidget(self.landIntera, 2, 1, 1, 1)
105         self.Radio_projeo = QtWidgets.QRadioButton(self.groupBoxMetodo)
106         self.Radio_projeo.setObjectName("Radio_projeo")
107         self.gridLayout_4.addWidget(self.Radio_projeo, 0, 0, 1, 1)
108         self.verticalLayout.addWidget(self.groupBoxMetodo)
109         self.groupBoxReconstru = QtWidgets.QGroupBox(self.groupBox)
110         self.groupBoxReconstru.setObjectName("groupBoxReconstru")
111         self.gridLayout_3 = QtWidgets.QGridLayout(self.groupBoxReconstru)
112         self.gridLayout_3.setObjectName("gridLayout_3")
113         self.Bt_reconstruirImagem = QtWidgets.QPushButton(self.groupBoxReconstru)
114         icon = QtGui.QIcon()
115         icon.addPixmap(QtGui.QPixmap("Img_Eit/ImgTab.png"), QtGui.QIcon.Normal,
    QtGui.QIcon.Off)
116         self.Bt_reconstruirImagem.setIcon(icon)
117         self.Bt_reconstruirImagem.setObjectName("Bt_reconstruirImagem")
118         self.gridLayout_3.addWidget(self.Bt_reconstruirImagem, 1, 0, 1, 1)
119         self.Bt_capturarImagem = QtWidgets.QPushButton(self.groupBoxReconstru)
120         icon1 = QtGui.QIcon()
121         icon1.addPixmap(QtGui.QPixmap("Img_Eit/ImgRec.png"), QtGui.QIcon.Normal,
    QtGui.QIcon.Off)
122         self.Bt_capturarImagem.setIcon(icon1)
123         self.Bt_capturarImagem.setObjectName("Bt_capturarImagem")
124         self.gridLayout_3.addWidget(self.Bt_capturarImagem, 2, 0, 1, 1)
125         self.verticalLayout.addWidget(self.groupBoxReconstru)
126         self.line = QtWidgets.QFrame(self.groupBox)
127         self.line.setFrameShape(QtWidgets.QFrame.HLine)
128         self.line.setFrameShadow(QtWidgets.QFrame.Sunken)
129         self.line.setObjectName("line")
130         self.verticalLayout.addWidget(self.line)
131         self.horizontalLayout.addWidget(self.groupBox)
132         self.line_12 = QtWidgets.QFrame(self.tab)
133         self.line_12.setFrameShape(QtWidgets.QFrame.VLine)
134         self.line_12.setFrameShadow(QtWidgets.QFrame.Sunken)
135         self.line_12.setObjectName("line_12")
136         self.horizontalLayout.addWidget(self.line_12)
137         self.groupBox_2 = QtWidgets.QGroupBox(self.tab)
138         sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Preferred,
    QtWidgets.QSizePolicy.MinimumExpanding)
139         sizePolicy.setHorizontalStretch(0)
140         sizePolicy.setVerticalStretch(0)
141         sizePolicy.setHeightForWidth(self.groupBox_2.sizePolicy().hasHeightForWidth
    ())
142         self.groupBox_2.setSizePolicy(sizePolicy)
143         self.groupBox_2.setMinimumSize(QtCore.QSize(300, 300))
144         self.groupBox_2.setAlignment(QtCore.Qt.AlignCenter)
145         self.groupBox_2.setObjectName("groupBox_2")
146         self.gridLayout = QtWidgets.QGridLayout(self.groupBox_2)
147         self.gridLayout.setObjectName("gridLayout")
148         spacerItem4 = QtWidgets.QSpacerItem(10, 10, QtWidgets.QSizePolicy.Minimum,
    QtWidgets.QSizePolicy.Minimum)
149         self.gridLayout.addItem(spacerItem4, 2, 1, 1, 1)
150         spacerItem5 = QtWidgets.QSpacerItem(10, 10, QtWidgets.QSizePolicy.Minimum,
    QtWidgets.QSizePolicy.Minimum)
```

```
151         self.gridLayout.addItem(spacerItem5, 1, 2, 1, 1)
152         spacerItem6 = QtWidgets.QSpacerItem(10, 10, QtWidgets.QSizePolicy.Minimum,
    QtWidgets.QSizePolicy.Minimum)
153         self.gridLayout.addItem(spacerItem6, 1, 0, 1, 1)
154         spacerItem7 = QtWidgets.QSpacerItem(10, 10, QtWidgets.QSizePolicy.Minimum,
    QtWidgets.QSizePolicy.Minimum)
155         self.gridLayout.addItem(spacerItem7, 0, 1, 1, 1)
156         self.Imagem_TIE = QtWidgets.QLabel(self.groupBox_2)
157         self.Imagem_TIE.setContextMenuPolicy(QtCore.Qt.NoContextMenu)
158         self.Imagem_TIE.setPixmap(QtGui.QPixmap("Img_Eit/LogoEit.png"))
159         self.Imagem_TIE.setObjectName("Imagem_TIE")
160         self.gridLayout.addWidget(self.Imagem_TIE, 1, 1, 1, 1)
161         self.horizontalLayout.addWidget(self.groupBox_2)
162         icon2 = QtGui.QIcon()
163         icon2.addPixmap(QtGui.QPixmap("Img_Eit/ImgTab.png"), QtGui.QIcon.Normal,
    QtGui.QIcon.Off)
164         self.Aplic.addTab(self.tab, icon2, "")
165         self.tab_2 = QtWidgets.QWidget()
166         self.tab_2.setObjectName("tab_2")
167         self.gridLayout_2 = QtWidgets.QGridLayout(self.tab_2)
168         self.gridLayout_2.setObjectName("gridLayout_2")
169         self.line_11 = QtWidgets.QFrame(self.tab_2)
170         self.line_11.setFrameShape(QtWidgets.QFrame.VLine)
171         self.line_11.setFrameShadow(QtWidgets.QFrame.Sunken)
172         self.line_11.setObjectName("line_11")
173         self.gridLayout_2.addWidget(self.line_11, 1, 2, 1, 1)
174         self.line_9 = QtWidgets.QFrame(self.tab_2)
175         self.line_9.setFrameShape(QtWidgets.QFrame.VLine)
176         self.line_9.setFrameShadow(QtWidgets.QFrame.Sunken)
177         self.line_9.setObjectName("line_9")
178         self.gridLayout_2.addWidget(self.line_9, 1, 0, 1, 1)
179         self.groupBox_3 = QtWidgets.QGroupBox(self.tab_2)
180         self.groupBox_3.setMaximumSize(QtCore.QSize(16777215, 100))
181         self.groupBox_3.setTitle("")
182         self.groupBox_3.setObjectName("groupBox_3")
183         self.gridLayout_5 = QtWidgets.QGridLayout(self.groupBox_3)
184         self.gridLayout_5.setObjectName("gridLayout_5")
185         self.progressBar = QtWidgets.QProgressBar(self.groupBox_3)
186         self.progressBar.setProperty("value", 0)
187         self.progressBar.setObjectName("progressBar")
188         self.gridLayout_5.addWidget(self.progressBar, 0, 0, 1, 1)
189         self.gridLayout_2.addWidget(self.groupBox_3, 3, 1, 1, 1)
190         self.line_2 = QtWidgets.QFrame(self.tab_2)
191         self.line_2.setFrameShape(QtWidgets.QFrame.HLine)
192         self.line_2.setFrameShadow(QtWidgets.QFrame.Sunken)
193         self.line_2.setObjectName("line_2")
194         self.gridLayout_2.addWidget(self.line_2, 2, 1, 1, 1)
195         self.line_6 = QtWidgets.QFrame(self.tab_2)
196         self.line_6.setWindowModality(QtCore.Qt.WindowModal)
197         self.line_6.setFrameShape(QtWidgets.QFrame.HLine)
198         self.line_6.setFrameShadow(QtWidgets.QFrame.Sunken)
199         self.line_6.setObjectName("line_6")
200         self.gridLayout_2.addWidget(self.line_6, 0, 1, 1, 1)
201         self.MedidasTens = QtWidgets.QGroupBox(self.tab_2)
202         self.MedidasTens.setMaximumSize(QtCore.QSize(16777215, 16777215))
203         self.MedidasTens.setAlignment(QtCore.Qt.AlignCenter)
204         self.MedidasTens.setObjectName("MedidasTens")
205         self.gridLayout_6 = QtWidgets.QGridLayout(self.MedidasTens)
```

```
206         self.gridLayout_6.setObjectName("gridLayout_6")
207         self.grupoStatus = QtWidgets.QGroupBox(self.MedidasTens)
208         self.grupoStatus.setLayoutDirection(QtCore.Qt.LeftToRight)
209         self.grupoStatus.setAlignment(QtCore.Qt.AlignCenter)
210         self.grupoStatus.setObjectName("grupoStatus")
211         self.gridLayout_7 = QtWidgets.QGridLayout(self.grupoStatus)
212         self.gridLayout_7.setObjectName("gridLayout_7")
213         spacerItem8 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding
    , QtWidgets.QSizePolicy.Minimum)
214         self.gridLayout_7.addItem(spacerItem8, 1, 2, 1, 1)
215         self.Img_status = QtWidgets.QLabel(self.grupoStatus)
216         self.Img_status.setMinimumSize(QtCore.QSize(300, 300))
217         self.Img_status.setText("")
218         self.Img_status.setPixmap(QtGui.QPixmap("Img_Eit/desconhecido.png"))
219         self.Img_status.setObjectName("Img_status")
220         self.gridLayout_7.addWidget(self.Img_status, 1, 1, 1, 1)
221         spacerItem9 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding
    , QtWidgets.QSizePolicy.Minimum)
222         self.gridLayout_7.addItem(spacerItem9, 1, 0, 1, 1)
223         spacerItem10 = QtWidgets.QSpacerItem(20, 40, QtWidgets.QSizePolicy.Minimum,
     QtWidgets.QSizePolicy.Expanding)
224         self.gridLayout_7.addItem(spacerItem10, 2, 1, 1, 1)
225         spacerItem11 = QtWidgets.QSpacerItem(20, 40, QtWidgets.QSizePolicy.Minimum,
     QtWidgets.QSizePolicy.Expanding)
226         self.gridLayout_7.addItem(spacerItem11, 0, 1, 1, 1)
227         self.gridLayout_6.addWidget(self.grupoStatus, 1, 1, 5, 1)
228         self.pushButton = QtWidgets.QPushButton(self.MedidasTens)
229         self.pushButton.setMaximumSize(QtCore.QSize(200, 16777215))
230         self.pushButton.setObjectName("pushButton")
231         self.gridLayout_6.addWidget(self.pushButton, 2, 0, 1, 1)
232         self.pushButton_2 = QtWidgets.QPushButton(self.MedidasTens)
233         self.pushButton_2.setMaximumSize(QtCore.QSize(200, 16777215))
234         self.pushButton_2.setObjectName("pushButton_2")
235         self.gridLayout_6.addWidget(self.pushButton_2, 3, 0, 1, 1)
236         spacerItem12 = QtWidgets.QSpacerItem(20, 40, QtWidgets.QSizePolicy.Minimum,
     QtWidgets.QSizePolicy.Expanding)
237         self.gridLayout_6.addItem(spacerItem12, 1, 0, 1, 1)
238         spacerItem13 = QtWidgets.QSpacerItem(20, 40, QtWidgets.QSizePolicy.Minimum,
     QtWidgets.QSizePolicy.Expanding)
239         self.gridLayout_6.addItem(spacerItem13, 4, 0, 1, 1)
240         self.gridLayout_2.addWidget(self.MedidasTens, 1, 1, 1, 1)
241         self.Aplic.addTab(self.tab_2, icon2, "")
242         self.horizontalLayout_2.addWidget(self.Aplic)
243         MainWindow.setCentralWidget(self.centralwidget)
244         self.menubar = QtWidgets.QMenuBar(MainWindow)
245         self.menubar.setGeometry(QtCore.QRect(0, 0, 917, 26))
246         self.menubar.setObjectName("menubar")
247         MainWindow.setMenuBar(self.menubar)
248         self.statusbar = QtWidgets.QStatusBar(MainWindow)
249         self.statusbar.setObjectName("statusbar")
250         MainWindow.setStatusBar(self.statusbar)
251         self.retranslateUi(MainWindow)
252         self.Aplic.setCurrentIndex(0)
253         QtCore.QMetaObject.connectSlotsByName(MainWindow)
254 
255 
256         #Variaveis utilizadas no sistema.
257         self.routeSensibilidade = None
```

```
258         self.routeTenseRef = None
259         self.routeMeasure = None
260         self.imagem = None
261         self.listaLeituras = list()
262 
263 
264         #set eventos de click
265         self.Bt_sensibilidade.clicked.connect(self.setSensibilidade)
266         self.Bt_medidas.clicked.connect(self.setMeasure)
267         self.Bt_tense_ref.clicked.connect(self.setTenseRef)
268 
269         self.Bt_reconstruirImagem.clicked.connect(self.rebuidIMG)
270 
271         self.Bt_capturarImagem.clicked.connect(self.saveIMG)
272 
273         self.pushButton.clicked.connect(self.saveLeituras)
274 
275         self.pushButton_2.clicked.connect(self.realizaAquis)
276 
277 
278 
279 
280         #desabilitar botões
281         self.Bt_capturarImagem.setEnabled(False)
282 
283         self.pushButton.setEnabled(False)
284 
285         self.progressBar.setValue(0)
286 
287 
288 
289     def retranslateUi(self, MainWindow):
290         _translate = QtCore.QCoreApplication.translate
291         MainWindow.setWindowTitle(_translate("MainWindow", "MainWindow"))
292         self.groupBox.setTitle(_translate("MainWindow", "COMANDOS"))
293         self.groupBoxArquivos.setTitle(_translate("MainWindow", "ARQUIVOS DE
    CONFIGURAÇÕES"))
294         self.Bt_sensibilidade.setText(_translate("MainWindow", "Sensibilidade"))
295         self.Line_sensibilidade.setText(_translate("MainWindow", ""))
296         self.Bt_tense_ref.setText(_translate("MainWindow", " Referência"))
297         self.Line_tense_ref.setText(_translate("MainWindow", ""))
298         self.Bt_medidas.setText(_translate("MainWindow", "Medidas"))
299         self.Line_medida.setText(_translate("MainWindow", ""))
300         self.groupBoxMetodo.setTitle(_translate("MainWindow", "Método"))
301         self.radio_landwaber.setText(_translate("MainWindow", "LandWeber"))
302         self.Radio_projeo.setText(_translate("MainWindow", "Retroprojeção linear"))
303         self.groupBoxReconstru.setTitle(_translate("MainWindow", "Reconstrução"))
304         self.Bt_reconstruirImagem.setText(_translate("MainWindow", "Reconstruir
    imagem "))
305         self.Bt_capturarImagem.setText(_translate("MainWindow", "Capturar Imagem"))
306         self.groupBox_2.setTitle(_translate("MainWindow", "TIE"))
307         self.Aplic.setTabText(self.Aplic.indexOf(self.tab), _translate("MainWindow
    ", "Reconstrução"))
308         self.MedidasTens.setTitle(_translate("MainWindow", "MEDIÇÕES"))
309         self.grupoStatus.setTitle(_translate("MainWindow", "STATUS DA PLACA DE
    AQUISIÇÕES"))
310         self.pushButton.setText(_translate("MainWindow", "Salvar Leitura"))
311         self.pushButton_2.setText(_translate("MainWindow", "Iniciar Aquisições"))
```

```
312         self.Aplic.setTabText(self.Aplic.indexOf(self.tab_2), _translate("
    MainWindow", "Obter dados"))
313 
314 
315     def setSensibilidade(self):
316         self.Line_sensibilidade.setText(f"{QtWidgets.QFileDialog.getOpenFileName()
    [0]}")
317 
318 
319     def setMeasure(self):
320         self.Line_medida.setText(f"{QtWidgets.QFileDialog.getOpenFileName()[0]}")
321 
322 
323     def setTenseRef(self):
324         self.Line_tense_ref.setText(f"{QtWidgets.QFileDialog.getOpenFileName()
    [0]}")
325 
326     def saveIMG(self):
327         arquivo = QtWidgets.QFileDialog.getSaveFileName()[0]
328 
329         if(arquivo != ""):
330             cv2.imwrite(arquivo + '.png', self.imagem)
331 
332 
333 
334     def rebuid(self):
335         width = self.Imagem_TIE.width()
336         height = self.Imagem_TIE.height()
337 
338         if width <= height:
339             height = width
340         else:
341             width = height
342 
343         intera  = int(self.landIntera.text()) #retorna o valor colocado de interaçõ
    es em landwaber
344 
345         imagem = TIE.rebuildImg(width,
346                                 height,
347                                 RouteFilejn=self.Line_sensibilidade.text(),
348                                 RouteFileVobj=self.Line_medida.text(),
349                                 RouteFileVref=self.Line_tense_ref.text(),
350                                 method = self.checkedMethod(),
351                                 intera_LDWBR = intera)
352 
353 
354 
355 
356         self.imagem = cv2.cvtColor(imagem, cv2.COLOR_RGB2BGR)
357         imagem = QtGui.QImage(imagem,width,height,imagem.strides[0],QtGui.QImage.
    Format_RGB888) # reformula o vetor de imagem para aparever na interface visual
358 
359 
360 
361         self.Imagem_TIE.setPixmap(QtGui.QPixmap.fromImage(imagem))
362 
363         self.Bt_capturarImagem.setEnabled(True)
364         self.Bt_reconstruirImagem.setEnabled(True)
```

```
365
366
367     def rebuidIMG(self):
368
369         if(self.activeRebuidImg()):
370             self.Bt_reconstruirImagem.setEnabled(False)
371             threa = threading.Thread(target = self.rebuid)
372             threa.start()
373
374
375     def activeRebuidImg(self):
376
377         if self.existsRouteSensebilidade():
378             QMessageBox.about(self.centralwidget,"ERRO","Sem arquivo sensibilidade
    ")
379             return False
380
381         if self.existRouteMeasure():
382             QMessageBox.about(self.centralwidget,"ERRO","Sem arquivo de Medidas")
383             return False
384
385
386         if self.existRouteVref():
387             QMessageBox.about(self.centralwidget,"ERRO","Sem arquivo de Tensão
    referência")
388             return False
389
390         if self.existRadioMethod():
391             QMessageBox.about(self.centralwidget,"ERRO","Escolha entre um dos
    modelos")
392             return False
393
394         if self.checkInteraLandweber():
395             QMessageBox.about(self.centralwidget,"ERRO","Valor de interações deve
    ser maior que zero")
396             return False
397         return True
398
399
400     def existsRouteSensebilidade(self):
401         return (self.Line_sensibilidade.text() == "")
402
403
404     def existRouteMeasure(self):
405         return (self.Line_medida.text() == "")
406
407
408     def existRouteVref(self):
409         return (self.Line_tense_ref.text() == "")
410
411
412     def existRadioMethod(self):
413         return not (self.radio_landwaber.isChecked() or self.Radio_projeo.isChecked
    ())
414
415
416     def checkInteraLandweber(self):
417         return (self.landIntera.text() == '0')
```

```
418
419
420     def checkedMethod(self):
421
422         if self.radio_landwaber.isChecked():
423             return "LDWBR"
424         if self.Radio_projeo.isChecked():
425             return "RPLIN"
426
427
428     def realizaAquis(self):
429         self.progressBar.setValue(0)
430         self.Img_status.setPixmap(QtGui.QPixmap("Img_Eit/conectado.png"))
431         threa = threading.Thread(target = self.aquisiSPI)
432         threa.start()
433
434
435     def saveLeituras(self):
436         route = QtWidgets.QFileDialog.getSaveFileName()[0]
437
438         if(route != ""):
439             with open(route +'.csv', 'w') as f:
440                 w = csv.writer(f)
441
442                 w.writerow(self.listaLeituras)
443
444                 f.close()
445             self.listaLeituras = []
446             self.pushButton.setEnabled(False)
447             self.progressBar.setValue(0)
448
449
450
451
452     def aquisiSPI(self):
453         self.pushButton_2.setEnabled(False)
454         arduinoSPI = Comunica.ComunicaSPI()
455         arduinoSPI.initSPI()
456
457         valor = 0
458
459
460         numero= 0
461
462         for PrimeiroEmissor in range(0,16):
463
464             if(PrimeiroEmissor == 15):
465                 SegundoEmissor = 0
466             else:
467                 SegundoEmissor = PrimeiroEmissor + 1
468
469
470             if(arduinoSPI.correntePin(PrimeiroEmissor,SegundoEmissor)):
471
472                 time.sleep(0.1)
473
474                 for Leituras in range(1,14):
475                     PrimeiroEletrodoTensao = SegundoEmissor + Leituras
```

```
476
477                     if(PrimeiroEletrodoTensao>14):
478                         PrimeiroEletrodoTensao = PrimeiroEletrodoTensao - 15
479
480                     SegundoEletrodoTensao = PrimeiroEletrodoTensao + 1
481
482                     if(arduinoSPI.tensePin(PrimeiroEletrodoTensao,
    SegundoEletrodoTensao)):
483                         arduinoSPI.ativaLeituras()
484                         valor = arduinoSPI.Leitura()
485                         self.listaLeituras.append(valor)
486
487                         numero += 1
488                         progresso = int(numero*100/208)
489                         self.progressBar.setValue(progresso)
490                         time.sleep(0.1)
491
492         if(len(self.listaLeituras)>=208):
493             self.pushButton.setEnabled(True)
494             self.pushButton_2.setEnabled(True)
495             self.Img_status.setPixmap(QtGui.QPixmap("Img_Eit/desconectado.png"))
496
497
```

```
498
499 import csv
500 from turtle import st
501 from typing import ValuesView
502 import numpy
503 import cv2
504 import csv
505
506
507
508 def color_normalized(valor_normalized:float):
509     '''
510     valor_normalized:float
511
512     Ao entrar com um valor normalizado, de 0 a 1, a função retorna uma lista com
    valores RGB
513     sendo que na lista retorna os valores de (RED,GREEN,BLUE)
514
515     '''
516     if valor_normalized > 0.55 and valor_normalized <= 1.0:
517         red = 255
518         gree = ((1 - valor_normalized)/(1- 0.55))*255
519         blue = 0
520         return [int(red),int(gree),int(blue)]
521     elif valor_normalized > 0.45 and valor_normalized<= 0.55:
522         red = ((valor_normalized - 0.45)/(0.55 - 0.45))*255
523         gree = 255
524         blue = ((0.55 - valor_normalized)/(0.55 - 0.45))*255
525         return [int(red),int(gree),int(blue)]
526     elif valor_normalized >= 0 and valor_normalized<= 0.45:
```

```
527         red = 0
528         gree = (valor_normalized/0.45)*255
529         blue = 255
530         return [int(red),int(gree),int(blue)]
531     else:
532         return [0,0,0]
533
534
535
536
537
538 def color_triangles_normalized( *args):
539     '''
540     args:list
541
542     Entre com um vetor normalizado (valores de 0 a 1) para obter a lista de lista
        com as cores.
543
544     '''
545     color_list = list()
546
547     max_value = float(max(args[0]))
548     min_value = float(min(args[0]))
549
550     valores_imagem = list()
551
552     if len(args) > 1:
553         return[]
554     for normalized_valor in args[0]:
555
556         valor = float(normalizedValue(float(normalized_valor),min_value,max_value))
557         valores_imagem.append(valor)
558         color_list.append(color_normalized(valor))
559
560     with open('/home/pi/Desktop/matrixImagem.csv', 'w') as f:
561         w = csv.writer(f)
562         w.writerow(valores_imagem)
563         f.close()
564     return color_list
565
566
567 def normalizedValue(value,min_value,max_value):
568     '''
569     '''
570     diff = float(max_value) - float(min_value)
571     ret  = (value - min_value)/diff
572     return ret
573
574
575
576 def open_csv(*args):
577     '''
578     args:list
579
580     entrada de uma lista com o caminho para a pasta desejada com as informações.
581     a lista deve contar com o caminho do arquivo em formator de string separado
582     por virgulas
583     '''
```

```
584     #route = os.path.join(os.getcwd(),args[0])
585     route = args[0]
586     with open(f"{route}",'r') as points:
587         reader_points = list(csv.reader(points, delimiter=';',quoting= csv.
        QUOTE_NONNUMERIC))
588     return reader_points
589
590
591 def shapes_points(step:int = 3,width_window:int = 400,
592                   height_window:int = 400,points_shapes:list = []):
593     '''
594     step:int
595     width_window:int
596     height_window:int
597     points_shapes:list
598
599     Gerador de lista com os pontos de cada conjunto de poligonos desejados sendo
        que no minimo
600     para formar um poligono é necessario 3 pontos.
601
602     '''
603
604     #Separar em uma lista de++ tuplas onde cada tupla recebera  outras 3 tuplas
        sendo que representarão um dos vetices dos triangulos
605     range_points = range(0,len(points_shapes))
606     shape_list_points = list()
607
608     for i in range_points[::step]:
609
610         #lista de repasse para a lista principal que sera acessada e realizara a
        reconstrução da imagem
611         lista_repas = list()
612
613         #laço de repetição para a criação das tupla interna a lista de repas
614         for p in range(0,step):
615             #calculo realizado para da posição do novo ponto, levando em consideraç
        ão o tamanho da tela criada
616             lista_repas.append((int(points_shapes[i+p][0]*width_window/2+
        width_window/2),
617                                 int(points_shapes[i+p][1]*height_window/2+
        height_window/2)))
618
619         #Adicionando na lista principal a tupla de posições com os vertices do
        triangulo
620         shape_list_points.append(lista_repas)
621
622
623     return shape_list_points
624
625
626
627 def retVobj(lista1,lista2):
628
629
630     retorno = list()
631
632     if(len(lista1)==len(lista2)):
633         for i in range(0,len(lista1)):
```

```
634             retorno.append(((lista1[i]-lista2[i])/lista1[i]))
635
636     with open('/home/pi/Desktop/matrixDifere.csv', 'w') as f:
637         w = csv.writer(f)
638         w.writerow(retorno)
639         f.close()
640
641     return retorno
642
643
644
645 def projeLiner(Sensiblidade:list = [],Vobj:list = [],Vref:list = []):
646     """
647     Realiza o calculo por meio do metodo retroprojeção linear
648
649     Args:
650         Sensiblidade (list): matriz sensibilidade
651         Vobj (list): Matriz com os valores de medições das variações de tensão
652         Vref (list): Matriz com os valores de tensão referência
653
654     Returns:
655         list: Matriz resultado
656     """
657
658     Vobj = numpy.array(Vobj).T
659     Vref = numpy.array(Vref).T
660
661     Lamb = retVobj(Vref,Vobj)
662     matrix_Lamb = numpy.array(Lamb)
663     matrix_S = numpy.array(Sensiblidade)
664     matrix_St = matrix_S.T
665
666     Gk = matrix_St.dot(matrix_Lamb)
667
668     return list(Gk)
669
670 def landweber(Sensiblidade:list = [],Vobj:list = [],Vref:list = [],intera:int = 0):
671     """
672     Realiza o calculo por meio do metodo interativo de LandWaber
673
674     Args:
675         Sensiblidade (list): matriz sensibilidade
676         Vobj (list): Matriz com os valores de medições das variações de tensão
677         Vref (list): Matriz com os valores de tensão referência
678         intera (int): Nuemro de interações que será realizado no metodo de
    LandWaber
679
680     Returns:
681         list: Matriz resultado da interação de LandWaber
682     """
683     Vobj = numpy.array(Vobj).T
684     Vref = numpy.array(Vref).T
685
686     Lamb = retVobj(Vref,Vobj)
687     #Lamb = numpy.array(((Vobj-Vref)/Vref))
688     matrix_Lamb = numpy.array(Lamb)
689     matrix_S = numpy.array(Sensiblidade)
690     matrix_St = matrix_S.T
```

```
691
692     #StS = matrix_St.dot(matrix_S)
693     #autovalor,autovetores = numpy.linalg.eig(StS)
694
695
696     #max_auto = max(autovalor)
697
698     #Gk = matrix_St.dot(matrix_Lamb)
699     Gk = numpy.zeros((1024,1))
700
701     #mi = 2/(max_auto.real)
702     mi = 1.5926
703
704     listaR = list()
705     r = matrix_Lamb-matrix_S.dot(Gk)
706
707     quantidade = 0
708
709     while(quantidade<intera):
710
711         quantidade += 1
712
713         Go = Gk + mi*matrix_St @ r
714
715         Go = numpy.array(convergeF(Go))
716
717         r = matrix_Lamb-matrix_S @ Go
718
719         Gk = Go
720
721
722     return list(Gk)
723
724
725 def convergeF(lista:list = []):
726     for i,argumento in enumerate(lista):
727         if(argumento<0):
728             lista[i] = 0
729
730         elif(argumento>1):
731             lista[i] = 1
732
733     return lista
734
735
736 def rebuildImg(Rwidth:int = 500,
737                Rheight:int = 500,
738                RouteFileVobj:str = '',
739                RouteFileVref:str = '',
740                RouteFilejn:str = '',
741                method:str = '',
742                intera_LDWBR:int = 0):
743
744     imagem = numpy.zeros((Rwidth,Rheight,3), numpy.uint8)
745
746     point_list = open_csv('Cord.csv')
747     shapes = shapes_points(step=3, width_window = Rwidth,height_window= Rheight,
    points_shapes= point_list)
```

```
748     color = list()
749 
750     Vref = open_csv(RouteFileVref)
751 
752     jn = open_csv(RouteFilejn)
753 
754     Vobj = open_csv(RouteFileVobj)
755 
756     if method == "LDWBR":
757         total = landweber(jn,Vobj,Vref,intera_LDWBR)
758     if method == "RPLIN":
759         total = projeLiner(jn,Vobj,Vref)
760 
761 
762 
763     color = color_triangles_normalized(total)
764 
765     for i in range(0,len(shapes)):
766         pts = numpy.array(shapes[i],numpy.int32)
767         cv2.fillConvexPoly(imagem,pts,color[i])
768 
769     return imagem
```

```
771 from cmath import pi
772 import re
773 import spidev
774 import RPi.GPIO as GPIO
775 import time
776 import csv
777 
778 
779 class ComunicaSPI:
780     def initSPI(self):
781         self.spi = spidev.SpiDev(0,0) # conectar a interface /dev/spidev0.1
782         self.spi.max_speed_hz = 250000
783 
784 
785     def tensePin(self,pinoA,pinoB):
786         """
787         Seta e confere se os pinos de tensão setados estão corretos.
788 
789         Args
790             pinoA: Pino de medição
791             pinoB: Pino de referencia
792         Returns
793             bool: retorna de verdadeiro se conseguiu realializar e setar o pino de
    tensão
794 
795         """
796         pino = int(self.concatenaPins(int(pinoA),int(pinoB)))
797         self.setPinTense(pino)
798         numberIntera = 0
799         while ((self.getPinTense() != pino) and (numberIntera < 50)):
800             self.setPinTense(pinTense = pino)
```

```
801             numberIntera += 1
802             time.sleep(0.1)
803 
804         if numberIntera >= 50:
805             return False
806 
807         return True
808 
809 
810     def correntePin(self,pinoA:int,pinoB:int):
811         """
812         Seta e confere se os pinos de corrente setados estão corretos.
813 
814         Args
815             pinoA: Pino de emissão
816             pinoB: Pino de entrada
817         Returns
818             bool: retorna de verdadeiro se conseguiu realializar e setar o pino de
    corrente
819 
820         """
821         pino = int(self.concatenaPins(pinoA,pinoB))
822         self.setPinCorrente(pino)
823         numberIntera = 0
824         while ((self.getPinCorrente() != pino) and (numberIntera < 50)):
825             self.spi.writebytes([0x00])
826             self.setPinCorrente(pino)
827             numberIntera += 1
828             time.sleep(0.1)
829 
830 
831         if numberIntera >= 50:
832             return False
833 
834         return True
835 
836 
837     def ativaLeituras(self):
838         self.spi.writebytes([0xAA])
839 
840 
841     def Leitura(self):
842         self.spi.writebytes([0xCC])
843         while(self.spi.readbytes(1)[0] != 0xDD):
844             time.sleep(0.5)
845             self.spi.writebytes([0x00])
846             self.spi.writebytes([0xCC])
847         valor = self.getLeitura()
848         return valor
849 
850 
851 
852 
853 
854     def concatenaPins(self,firstPin,secondPin):
855         """
856         Args
857             firstPin: Pino A
```

```
858             secondPin: Pino B
859         Returns
860             str: retorna de forma concatenada o valor em hexadecimal dos dosi
    valores sendo 0X(firstPin)(secondPin)
861         """
862         return  str((firstPin <<4) + secondPin)
863 
864 
865     def setPinTense(self,pinTense:int):
866         self.spi.writebytes([0x22,pinTense]) #Seta o pino de tensão com o valor hex
     pinTense com o comando 0x22
867 
868 
869     def setPinCorrente(self,pinCorrente:int):
870         self.spi.writebytes([0x11,pinCorrente])#Seta o pino de Corrente com o valor
     hex pinCorrente com o comando 0x11
871 
872 
873     def getPinTense(self):
874         """
875         Retorna o eletrodo de leitura de tensão es carregado no arduino, lembrando
    que esse valor sera numerico mas
876         sera preciso de sua forma hexecimal para compreender qual eletrodo esta
    setado.
877 
878         Args:
879 
880         Return:
881             str: Valor em hexadecimal dos pinos de tensão
882         """
883         self.spi.writebytes([0x55])
884         return self.spi.readbytes(1)[0]
885 
886 
887     def getPinCorrente(self):
888         """
889         Retorna o eletrodo de corrente carregado no arduino, lembrando que esse
    valor sera numerico mas sera preciso
890         de sua forma hexecimal para compreender qual eletrodo esta setado.
891 
892         Args:
893 
894         Return:
895             str: Valor em hexadecimal dos pinos de tensão
896         """
897 
898         self.spi.writebytes([0x44])
899         return self.spi.readbytes(1)[0]
900 
901 
902 
903     def getLeitura(self):
904 
905         leitura = 0
906 
907         self.spi.writebytes([0x99])
908         time.sleep(0.1)
909         leitura1 = self.spi.readbytes(1)
```

```
910
911         leitura2 = self.spi.readbytes(1)
912
913
914         leitura = int((leitura2[0] << 8)+ leitura1[0])
915         return leitura
```

APÊNDICE B – CÓDIGO ARDUINO

```
919
920
921 #include "eit.h"
922
923
924 /*
925  * Código de comunicação e comutação entre eletrodos utilizando SPI
926  *
927  * MISO -> nativo do arduino due
928  * MOSI -> Nativo do arduino due
929  * SCK -> nativo do arduino due
930  * SS -> pino 10 do arduino due
931  *
932 */
933
934
935 //~---------------------------Definições-----------------------------
936
937 #define dataOk 3
938
939 int stepPin = 3;
940
941 int initPin = 23;
942
943 int finalPin = 53;
944
945 int tensao = 0;
946
947
948 //-----------------Definirescolpo de funções----------
949
950 //config
951
952
953 //func
954 void configPinDigital(int inicialPin = 0,int finalPin = 0, int stepPin = 0, bool
        mode = false);
955 void digitalWritePinEIT();
956 byte find_Action(byte receive);
957 byte interpretAction(byte action,int receive);
958 int tens();
959
960
961 //convert
962 int convertHexToInt();
963 String ValueIntVectByte();
964
965
966
967 //_________________variaveis globais_________________
968
969 bool flag=false;
970 bool flagDataConplete = false;
971
972 uint16_t DataReceived;
```

```
973
974 uint16_t data;
975
976 uint16_t eletrodoEmissor;
977 uint16_t eletrodoLeitor;
978
979 enum NexAction {SetZero,SetWriteEIT,SetReadEIT,SetReLoadRasberry};
980
981 byte DataReceivedIndex = 0;
982
983 eit_eletrodo eletrodos(23,53,2);
984
985 void setup() {
986   Serial.begin(9600);
987   while(!Serial);
988   //SPI serial recieve
989   REG_PMC_PCER0 |= PMC_PCER0_PID24;   // Power up SPI clock
990   REG_SPI0_WPMR = 0<<SPI_WPMR_WPEN;   //Unlock user interface for SPI
991
992   //Instance SPI0, MISO: PA25, (MISO), MOSI: PA26, (MOSI), SCLK: PA27, (SCLK), NSS:
        PA28, (NPCS0)
993   REG_PIOA_ABSR &= ~PIO_ABSR_P25;     // Transfer Pin control from PIO to SPI
994   REG_PIOA_PDR |= PIO_PDR_P25;      // disable pio to control this pin (MISO)
995
996   REG_PIOA_ABSR &= ~PIO_ABSR_P26;     // Transfer Pin control from PIO to SPI
997   REG_PIOA_PDR |= PIO_PDR_P26;      // disable pio to control this pin (MOSI)
998
999   REG_PIOA_ABSR &= ~PIO_ABSR_P27;     // Transfer Pin control from PIO to SPI
1000  REG_PIOA_PDR |= PIO_PDR_P27;      // disable pio to control this pin (SCLK)
1001
1002  REG_PIOA_ABSR &= ~PIO_ABSR_P28;     // Transfer Pin control from PIO to SPI
1003  REG_PIOA_PDR |= PIO_PDR_P28;      //disable pio to control this pin (NSS)
1004
1005  REG_SPI0_CR = 1;            // Enable SPI
1006  REG_SPI0_MR = 0;            // Slave mode
1007
1008  SPI0->SPI_IER = SPI_IER_RDRF;     // Receive Data Register Full Interrupt
1009  NVIC_EnableIRQ(SPI0_IRQn);
1010
1011  SPI0->SPI_CSR[0] = SPI_CSR_NCPHA|SPI_CSR_BITS_8_BIT;  // Shift on falling edge
        and transfer 8 bits. SPI_CSR_BITS_16_BIT  SPI_CSR_BITS_8_BIT
1012  pinMode(3,INPUT); // Seta o pino 3 como sendo o chip select do SPI
1013
1014  Serial.println("START");
1015
1016
1017 }
1018
1019
1020
1021 bool flagData = false;
1022 bool flagReturn = false;
1023 bool flagEletrodoCorrente = false;
1024 bool flagEletrodoLeitorTensao = false;
1025 bool flagExecutar = false;
1026 bool flagReturnValueEletrodoTense = false;
1027 bool flagReturnValueEletrodoCorrent = false;
1028 bool flagReturnValueEletrodotens = false;
```

```
1029 bool leituraAnalogica = false;
1030 bool flagEnvioDeDadosOk = false;
1031 bool flagEnvioDeDados = false;
1032 bool flagAtivarLeituras = false;
1033 bool flagParteDoDado = false;
1034 bool flagReceberEletrodoTensao = false;
1035
1036 byte contador = 0;
1037
1038 int emissor;
1039
1040 byte action = 0; //Selecção de qual comanda sera executado na proximo envio ou
         leitura;
1041
1042
1043
1044 void SPI0_Handler()
1045 {
1046   uint32_t status = SPI0->SPI_SR;
1047
1048
1049   if (status & SPI_SR_RDRF & !flagParteDoDado){
1050
1051     DataReceived = SPI0->SPI_RDR & SPI_RDR_RD_Msk;
1052
1053     if(DataReceived){
1054
1055       if(flagEletrodoCorrente){
1056         setWritePins(DataReceived);
1057         flagEletrodoCorrente = false;
1058       }
1059       else if(flagReceberEletrodoTensao){
1060         setReadTens(DataReceived);
1061         flagReceberEletrodoTensao = false;
1062       }
1063       else if(DataReceived == 0x11 & !flagEletrodoCorrente){
1064
1065         flagEletrodoCorrente = true;
1066       }
1067       else if(DataReceived == 0x22 & !flagReceberEletrodoTensao){
1068         flagReceberEletrodoTensao = true;
1069       }
1070       else if(DataReceived == 0x33){
1071         flagReturn = true;
1072       }
1073       else if(DataReceived == 0x44){
1074         flagReturnValueEletrodoCorrent = true;
1075       }
1076       else if(DataReceived == 0x55){
1077         flagReturnValueEletrodotens  = true;
1078       }
1079       else if(DataReceived == 0xAA){
1080         leituraAnalogica = true;
1081       }
1082       else if(DataReceived == 0xCC){
1083         flagEnvioDeDadosOk = true;
1084       }
1085       else if(DataReceived == 0xEE){
```

```
1086          flagAtivarLeituras = true;
1087        }
1088        else if(DataReceived == 0X99){
1089          flagParteDoDado = true;
1090        }
1091        else if(DataReceived){
1092          data = DataReceived;
1093        }
1094      }
1095
1096    }
1097    if (status & SPI_SR_TDRE){
1098
1099      if(flagReturn){
1100       SPI0->SPI_TDR = data  & SPI_RDR_RD_Msk;
1101       flagReturn = false;
1102       }
1103      else if(flagReturnValueEletrodoCorrent){
1104        SPI0->SPI_TDR = eletrodos.getValorEmissorCorrente();
1105        flagReturnValueEletrodoCorrent = false;
1106      }
1107      else if(flagReturnValueEletrodotens){
1108        SPI0->SPI_TDR = eletrodos.getValorLeituraTens();
1109        flagReturnValueEletrodotens = false;
1110      }
1111      else if(flagEnvioDeDadosOk){
1112        if(flagEnvioDeDados){
1113          SPI0->SPI_TDR = 0xDD;
1114        }
1115        else{
1116          SPI0->SPI_TDR = 0xBB;
1117        }
1118        flagEnvioDeDadosOk = false;
1119      }
1120      else if (flagParteDoDado){
1121        if(contador == 0){
1122          SPI0->SPI_TDR = tensao & 0b0000000011111111;
1123
1124          contador++;
1125        }
1126        else if(contador == 1){
1127          SPI0->SPI_TDR = tensao >> 8;
1128          contador++;
1129        }
1130        else if(contador == 2){
1131        contador = 0;
1132        tensao = 0;
1133        flagParteDoDado = false;
1134        flagEnvioDeDados = false;
1135        }
1136
1137      }
1138     }
1139
1140
1141  }
1142
1143  byte indice = 0;
```

```
1144
1145 void loop() {
1146
1147   if(leituraAnalogica & !flagEnvioDeDados){
1148     eletrodos.writeData();
1149
1150     delayMicroseconds(500);
1151     analogReadResolution(12);
1152     for(int p = 0; p<100;p++){
1153       tensao = tensao + analogRead(5);
1154       Serial.println(tensao);
1155       }
1156
1157     tensao = tensao/100;
1158     delayMicroseconds(200);
1159     flagEnvioDeDados = true;
1160     leituraAnalogica = false;
1161     }
1162
1163 }
1164
1165
1166 //----------- funções ---------
1167
1168 void setWritePins(int WritePins){
1169   eletrodos.setEmissorCorrente(WritePins);
1170 }
1171
1172 void setReadTens(int WritePins){
1173   eletrodos.setLeituraTens(WritePins);
1174 }
```

```
1176
1177 #include "Arduino.h"
1178 #include "eit.h"
1179
1180 eit_eletrodo::eit_eletrodo(int pinInit,int pinFinal,int stepPin)
1181 {
1182   // define os pinos de saída do arduino due como output;
1183   for(int i = pinInit; i<=pinFinal ;i+=stepPin){
1184     pinMode(i,OUTPUT);
1185     digitalWrite(i,LOW);
1186   }
1187 }
1188
1189 void eit_eletrodo::writeData(){
1190   // set os pinos dos multiplexadores
1191
1192   writeMultiPlex(_pin_emissor_corrente.saida,23,29);
1193   writeMultiPlex(_pin_emissor_corrente.entrada,31,37);
1194   writeMultiPlex(_pin_leitura_tensao.saida,39,45);
1195   writeMultiPlex(_pin_leitura_tensao.entrada,47,53);
1196
1197 }
```

```
1198
1199
1200 void eit_eletrodo::writeMultiPlex(byte values,byte initPin,byte finalPin){
1201   // Seta os pinos do multiplexador
1202   String setPin = String(values,BIN);
1203   byte i = 0;
1204   for(int pin = initPin; pin <= initPin +(finalPin - initPin) ;pin+=2){
1205     if(digitalRead(pin) != setPin[i]){
1206       if(setPin[i] == '0'){
1207         digitalWrite(pin,LOW);
1208       }
1209       else{
1210         digitalWrite(pin,HIGH);
1211       }
1212
1213     }
1214     i+=1;
1215   }
1216
1217 }
1218
1219 //------------------Set variaveis------------------
1220 void eit_eletrodo::setPinEmissorCorrenteSaida(byte pin_emissor_corrente_saida){
1221   _pin_emissor_corrente_saida = pin_emissor_corrente_saida;
1222 }
1223
1224 void eit_eletrodo::setPinEmissorCorrenteEntrada(byte pin_emissor_corrente_entrada){
1225   _pin_emissor_corrente_entrada = pin_emissor_corrente_entrada;
1226 }
1227
1228 void eit_eletrodo::setPinLeitorTensao(byte pin_leitor_tensao){
1229   _pin_leitor_tensao = pin_leitor_tensao;
1230 }
1231
1232 void eit_eletrodo::setPinLeitorTensaoReferencia(byte pin_leitor_tensao_referencia){
1233   _pin_leitor_tensao_referencia = pin_leitor_tensao_referencia;
1234 }
1235
1236 void eit_eletrodo::setEmissorCorrente(int eletrodosEmissoresCorrente){
1237   //Salva os eletros responsaveis por efetuar injetar corrente no sistemas
1238   _pin_emissor_corrente.valor_salvo = eletrodosEmissoresCorrente;
1239   _pin_emissor_corrente.saida = (eletrodosEmissoresCorrente >> 4);
1240   _pin_emissor_corrente.entrada = (eletrodosEmissoresCorrente & 0b00001111);
1241
1242 }
1243
1244 void eit_eletrodo::setLeituraTens(int eletrodosLeitorTensao){
1245   //Salva os eletros responsaveis por efetuar a leitura da variação de tensão
1246   _pin_leitura_tensao.valor_salvo = eletrodosLeitorTensao;
1247   _pin_leitura_tensao.saida = (eletrodosLeitorTensao >> 4);
1248   _pin_leitura_tensao.entrada = (eletrodosLeitorTensao & 0b00001111);
1249 }
1250
1251
1252
1253 //------------------Get variaveis------------------
1254
1255
```

```
1256 int eit_eletrodo::getValorLeituraTens(){
1257   return _pin_leitura_tensao.valor_salvo;
1258 }
1259
1260 int eit_eletrodo::getValorEmissorCorrente(){
1261   return _pin_emissor_corrente.valor_salvo;
1262 }
1263
1264 byte eit_eletrodo::getPinEmissorCorrenteSaida(){
1265   return _pin_emissor_corrente_saida;
1266 }
1267
1268 byte eit_eletrodo::getPinEmissorCorrenteEntrada(){
1269   return _pin_emissor_corrente_entrada;
1270 }
1271
1272 byte eit_eletrodo::getPinLeitorTensao(){
1273   return _pin_leitor_tensao;
1274 }
1275
1276 byte eit_eletrodo::getPinLeitorTensaoReferencia(){
1277   return _pin_leitor_tensao_referencia;
1278 }
```

```
1280
1281 #ifndef eit_eletrodo_H
1282 #define eit_eletrodo_H
1283
1284 #include "Arduino.h"
1285
1286 struct pinosAtivos{
1287   int valor_salvo;
1288   byte saida;
1289   byte entrada;
1290 };
1291
1292
1293 class eit_eletrodo
1294 {
1295   public:
1296       eit_eletrodo(int pinInit,int pinFinal,int stepPin);
1297
1298       //Metodos
1299
1300       void writeData();
1301       void writeMultiPlex(byte values,byte initPin,byte finalPin);
1302
1303       // Set variaveis
1304       void setPinEmissorCorrenteSaida(byte pin_emissor_corrente_saida);
1305       void setPinEmissorCorrenteEntrada(byte pin_emissor_corrente_entrada);
1306       void setPinLeitorTensao(byte pin_leitor_tensao);
1307       void setPinLeitorTensaoReferencia(byte pin_leitor_tensao_referencia);
1308
1309       void setEmissorCorrente(int eletrodosEmissoresCorrente);
```

```
1310        void setLeituraTens(int eletrodosLeitorTensao);
1311
1312
1313        //Get Variaveis
1314
1315        int getValorLeituraTens();
1316        int getValorEmissorCorrente();
1317
1318        byte getPinEmissorCorrenteSaida();
1319        byte getPinEmissorCorrenteEntrada();
1320        byte getPinLeitorTensao();
1321        byte getPinLeitorTensaoReferencia();
1322
1323
1324    private:
1325
1326
1327        byte _pin_emissor_corrente_saida;
1328        byte _pin_emissor_corrente_entrada;
1329        byte _pin_leitor_tensao;
1330        byte _pin_leitor_tensao_referencia;
1331
1332        pinosAtivos _pin_emissor_corrente;
1333        pinosAtivos _pin_leitura_tensao;
1334
1335
1336 };
1337
1338 #endif
```

```
1340     import spidev
1341 import RPi.GPIO as GPIO
1342 import time
1343 import csv
1344 spi = spidev.SpiDev(0, 0) # create spi object connecting to /dev/spidev0.1
1345 spi.max_speed_hz = 250000 # set speed to 250 Khz
1346
1347
1348
1349 leituras = list()
1350
1351 for emissroCorrente in range(0,16):
1352
1353     if(emissroCorrente==15):
1354         emissroCorrente2 = 0
1355     else:
1356         emissroCorrente2 = emissroCorrente  + 1
1357
1358     print("Corrente")
1359     print(f"{emissroCorrente} {emissroCorrente2}")
1360
1361     corrente = (emissroCorrente << 4) + emissroCorrente2
1362
1363     spi.writebytes([0x11,corrente ]) # write one by
```

```
1364     spi.writebytes([0x44])
1365     while((spi.readbytes(1)[0]) != corrente ):
1366         spi.writebytes([0x00,0x00])
1367         spi.writebytes([0x11,corrente])
1368         spi.writebytes([0x44])
1369 
1370     time.sleep(0.1)
1371 
1372 
1373 
1374     for i in range(1,14):
1375 
1376         emissor = emissroCorrente2 + i
1377         if (emissor>14):
1378             emissor = emissor - 15
1379 
1380         emissor2 = emissor  + 1
1381 
1382         emissores = (emissor << 4) + emissor2
1383 
1384         spi.writebytes([0x22,emissores]) # write one by
1385         spi.writebytes([0x55])
1386         while((spi.readbytes(1)[0]) != emissores):
1387             spi.writebytes([0x00,0x00])
1388             spi.writebytes([0x22,emissores])
1389             spi.writebytes([0x55])
1390 
1391         time.sleep(0.5)
1392         spi.writebytes([0xAA])
1393         ##spi.writebytes([0xFF,0xFE]) # write one byte
1394         spi.writebytes([0xCC])
1395         valor = spi.readbytes(1)
1396 
1397         while(valor[0] != 0xDD):
1398 
1399             spi.writebytes([0xCC])
1400             valor = spi.readbytes(1)
1401             time.sleep(1)
1402 
1403         spi.writebytes([0x99])
1404         values = spi.readbytes(1)
1405         values2 = spi.readbytes(1)
1406         values = (values2[0]  << 8) + values[0]
1407 
1408         leituras.append(values)
1409 
1410         print(values)
1411 
1412 
1413 
1414 with open('leituras.csv', 'w') as f:
1415 
1416     w = csv.writer(f)
1417 
1418     w.writerow(leituras)
1419 
1420     f.close()
1421 
```

```
1422 print(leituras)
1423 print(len(leituras))
1424
1425 """
```

ANEXOS