SENAC CAS
Física Aplicada

Tomógrafo

Física Eletrica

Andre Luiz Veras / duverassusa@gmail.com
Andrew Araujo / andrewaraujoco@gmai.com
Gustavo Borges / guborges4789a@gmail.com
Ruan Gomes / ruanggs.05@gmail.com

Professor:
Dalke Meucci
Jorge Echeimberg

5 de dezembro de 2025

1 Resumo

Um Tomógrafo é um equipamento médico utilizado para a obtenção de imagens detalhadas do interior do corpo humano ou de objetos. Ele emprega técnicas (como as Tomografias 'Computadorizadas' e as por 'Emissão de pósitrons') para gerar imagens transversais de alta resolução dos órgãos, tecidos ou estruturas anatômicas em questão. A tomografia computadorizada utiliza raios-X para criar imagens detalhadas de cortes transversais do corpo, enquanto a tomografia por emissão de pósitrons utiliza radiofármacos para detectar processos metabólicos. A combinação de ambas, permite a visualização precisa de estruturas anatômicas juntamente com informações funcionais e metabólicas. Essas imagens são fundamentais para diagnósticos médicos precisos, planejamento de tratamentos e acompanhamento de doenças. Os tomógrafos modernos oferecem tecnologias avançadas, como reconstrução tridimensional, redução de doses de radiação e maior rapidez na obtenção das imagens, contribuindo significativamente para a prática médica atual.

Proposta: A ideia proposta era de que, os alunos se reunissem em grupos para que pudessem trabalhar na produção de um Tomógrafo de Luz, apresentando e solucionando os problemas durante o percurso e estudando a aplicação de técnicas de processamento de imagens na linguagem de programação Python, onde teria a aquisição de dados que seria utilizada para formar a imagem completa de um objeto, sendo ele translúcido ou opaco.

Palavras-Chave: Tomógrafo, Programação, Python.

2 Introdução

O regente relatório visa apresentar os resultados de um projeto focalizado na concepção de um sistema de tratamento de dados pela linguagem Python. Este projeto apresenta uma abordagem inovadora que entrelaça os fundamentos teóricos do curso Física Elétrica com uma aplicação prática tangível.

Um Tomógrafo de Luz representa um avanço significativo na tecnologia de imagem médica, desempenhando um papel crucial na obtenção de informações detalhadas sobre estruturas internas do corpo humano. Este equipamento utiliza princípios ópticos avançados para proporcionar imagens transversais de alta resolução, revelando com precisão a morfologia de órgãos, tecidos e estruturas anatômicas. Sua aplicação abrange diversas áreas da medicina, permitindo diagnósticos mais precisos e contribuindo para a monitorização eficaz de condições médicas. Ao explorar a interação da luz com os tecidos biológicos, o Tomógrafo de Luz representa uma ferramenta inovadora que otimiza a obtenção de informações clínicas cruciais, promovendo avanços significativos na prática médica e pesquisa biomédica.

Ao longo das páginas iremos explorar as distintas fases do projeto desde sua concepção inicial até a sua finalização, abordando temas que incluem o funcionamento do tomógrafo, a construção do sistema de tratamento de dados e seus resultados.

3 Fundamentos da Tomografia e Transformada de Radon

O processo de tomografia pode ser realizado através de amostragens paralelas de feixes de raios x ou raios gama, em diferentes ângulos em um determinado objeto de análise. Tais feixes de raios sofrem atenuações de acordo com a distribuição de densidade do objeto, todo esse conjunto de perfis é o resultado da transformada de Radon do objeto em análise. Os dados resultantes da transformada de Radon são chamados de Sinograma.

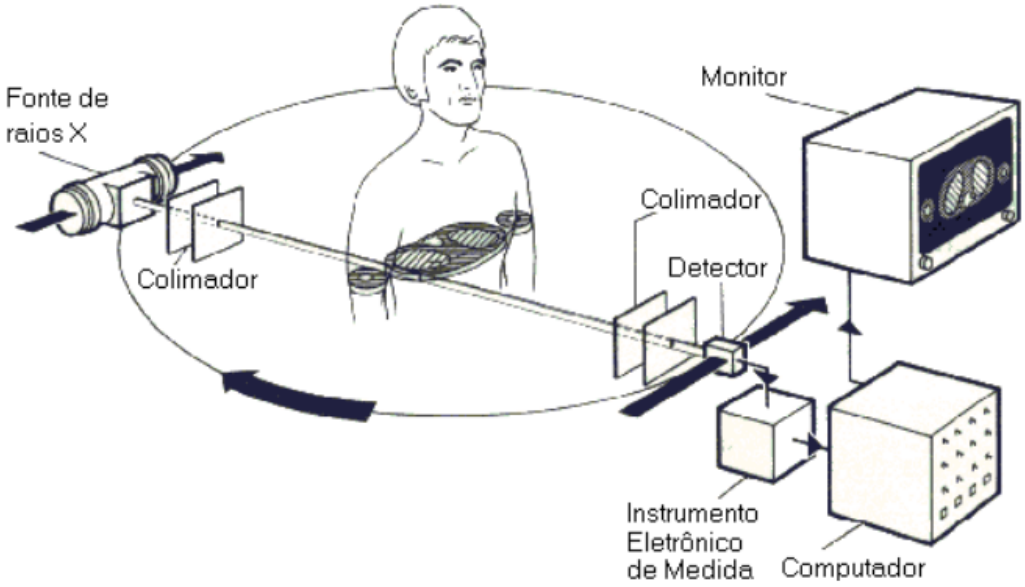

Figura 1: Ilustração do processo de tomografia, par emissor-detector rotaciona e translada realizando cortes em diferentes ângulos e trajetórias paralelas em torno do objeto.

3.1 Princípios Físicos

A trajetória do raio ionizante que atravessa o objeto pode ser aproximada por uma reta, e a intensidade desse raio é atenuada ao longo de uma linha com coeficiente de atenuação. Abaixo temos um exemplo de objeto com distribuição de atenuação não homogênea vista ao longo de uma trajetória, os intervalos no eixo u indicam o tamanho dos segmentos.

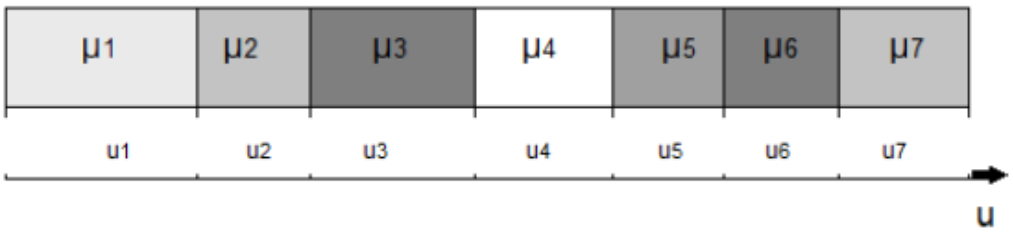

Figura 2: Distribuição da atenuação ao longo de uma trajetória do raio em um objeto.

A atenuação da intensidade obedece a lei de decaimento exponencial de Beer-Lambert. Para um objeto com distribuição da atenuação uniforme u1, e tamanho do segmento percorrido u1, tem-se:

$$\frac{I}{I_0} = e^{-u_1 \mu_1} \tag{1}$$

Aonde 1_0 é a intensidade máxima do raio ionizante, e 1 é a intensidade final do raio após atravessar o objeto. Para um objeto com distribuição não uniforme, similar ao objeto da figura 2, tem-se:

$$\frac{I}{I_0} = e^{-\sum_{i=1}^{n} u_i \mu_i} \tag{2}$$

Para se obter o análogo contínuo da equação 2, basta substituir o somatório dos elementos discretos pela integral, e tornar u como elemento de linha du, assim tem-se:

$$\frac{I}{I_0} = e^{-\int_a^b \mu(x,y)\, du} \tag{3}$$

Foi visto até então, como se obter os valores de intensidade para as trajetórias, agora há a necessidade de uma modelagem geométrica do problema e parâmetros que possam indicar a disposição desses raios.

3.2 Modelagem Matemáica

Em um tomógrafo de primeira geração, seu arranjo geométrico possui configuração que faz uso das projeções paralelas, cada trajetória de raio ionizante é parametrizado através da distância normal s à origem do sistema de coordenadas cartesiano e o ângulo θ da linha relativa ao primeiro eixo. Abaixo, na figura 3, uma ilustração de uma seção de um objeto sendo atravessado por uma trajetória do raio ionizante do tomógrafo.

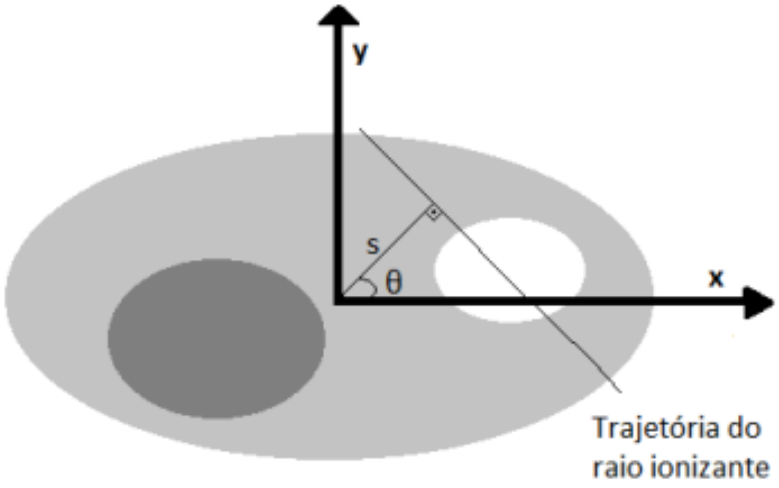

Figura 3: A trajetória pode ser descrita pelos parâmetros e θ.

Um conjunto de dos valores das intensidades das trajetórias com mesmo ângulo formam o que é chamado de projeção.
A equação 3 pode ser reescrita da seguinte forma:

$$\frac{I(s,\theta)}{I_0} = e^{-\int_a^b \mu(x,y)\,du} \tag{4}$$

Dessa forma a intensidade está relacionada aos parâmetros que identificam a trajetória.
A projeção g(s,θ) é definida como:

$$g(s,\theta) = ln\frac{I(s,\theta)}{I_0} \tag{5}$$

Assim, da equação 4, tem-se que:

$$g(s,\theta) = \int_{-\infty}^{\infty} \mu(x,y)\, du = \int_{-\infty}^{\infty} \mu(s\cos\theta - u\, sen\theta\ , s\, sen\theta + u\, cos\theta)\, du \tag{6}$$

A função de g(s,θ) é uma integral de linha em du de u(x,y), e também é a Transformada de Radon de u(x,y).

3.3 Transformada de Radon

A transformada de Radon converte uma função de um sistema de coordenadas espaciais (x,y) para outro sistema de coordenadas espaciais de Radon. O domínio de Radon descreve os valores das integrais de linha como da equação 6, através dos parâmetros de (s,θ). A equação de linha é expressa por xcosθ + ysenθ - s = 0. A expressão da linha pode também ser escrita na forma de reta paramétrica, onde é usado o parâmetro u. Abaixo, um exemplo de projeção, com uso desse parâmetro:

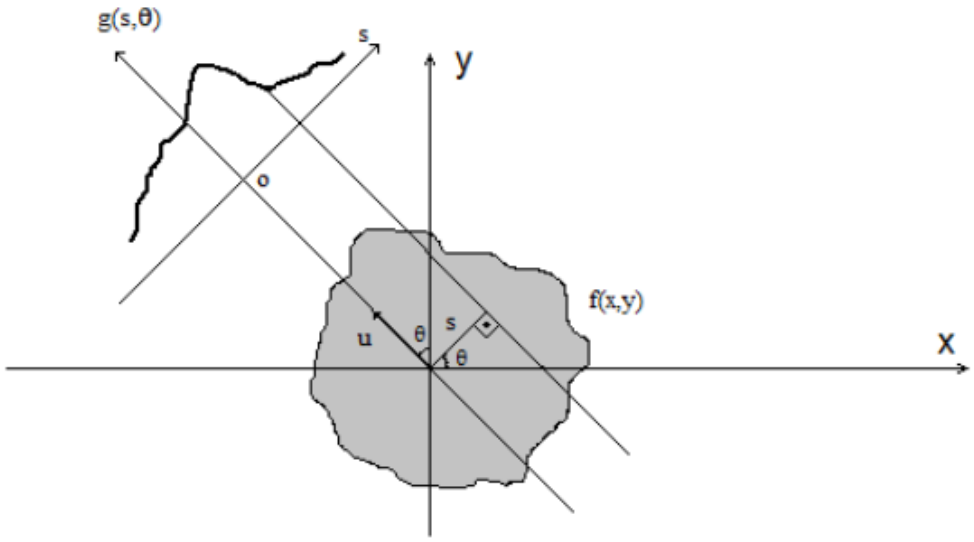

Figura 4: A trajetória pode ser parametrizada pelo vetor unitário de mesma direção.

Dessa forma a expressão que define a linha fica:

$$x = s\cos\theta - u\, sen\,\theta$$
$$y = s\, sen\theta + u\, cos\,\theta \tag{7}$$

Deixando em função de x e y:

$$s = x\,cos\theta + y\,sen\,\theta$$
$$u = -x\,sen\theta + y\,cos\,\theta \tag{8}$$

Assim, da equação 6 e 7, tem-se a Transformada de Radon para a função f(x,y) como:

$$g(s,\theta) = \int_{-\infty}^{\infty} f(s\,\cos\theta - u\,sen\theta\ , s\,sen\theta + u\,cos\theta)\,du \tag{9}$$

A projeção g(s,θ) é obtida por um conjunto de integrais de linha com mesmo ângulo θ. Abaixo, projeções nos ângulos θ_1 e θ_2:

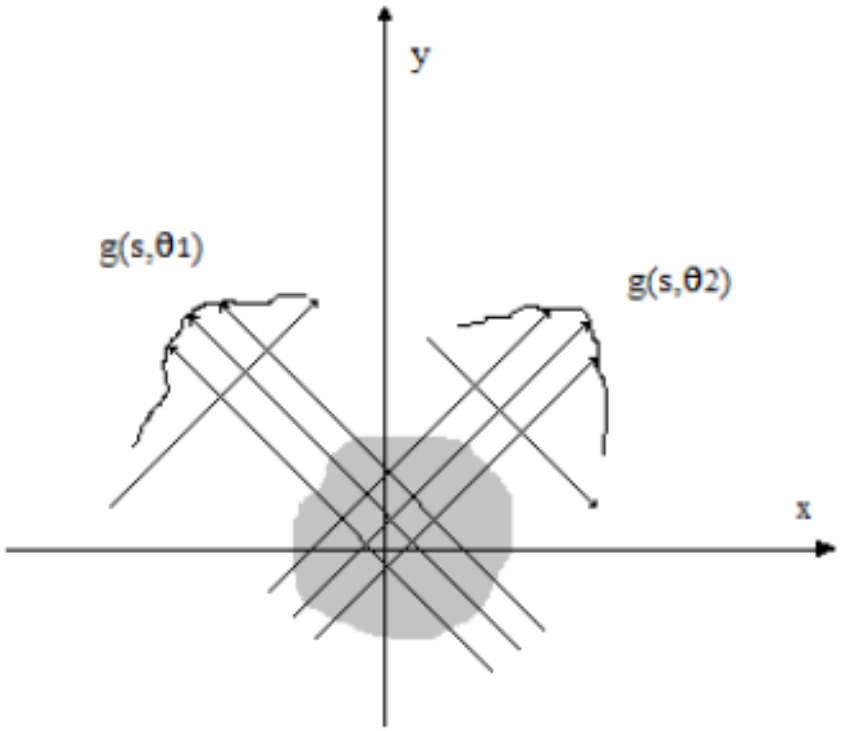

Figura 5: Projeções em dois diferentes ângulos.

A composição de diversas projeções forma o que é chamado de Sinograma, que é obtido no processo de tomografia, e esse conjunto de dados servem de ponto de partida para a reconstrução da imagem. A Transformada de Radon tem fundamental importância nesse processo, a tomografia nada mais é que a transformada de Radon aplicada.

4 Tomógrafo de Luz

O dispositivo apresentado é um sistema simplificado de digitalização 3D que incorpora tecnologias inspiradas na tomografia por emissão de luz e scanner 3D que utiliza a técnica de "digitalização por projeção de luz estruturada". Embora seja considerado "caseiro"e "simples", esse sistema oferece uma abordagem abrangente e precisa para a captura de objetos tridimensionais, destacando-se pela sua modularidade e flexibilidade. Apesar de alguma perda de precisão associada à sua natureza simplificada, essa característica é compensada pela adaptabilidade única do dispositivo. Adicionalmente, o aparelho incorpora os princípios da tomografia por emissão de luz, uma técnica que envolve a projeção de luz através do objeto para coletar informações sobre sua densidade e propriedades ópticas. Este método complementa eficazmente a tomografia por feixe cônico, proporcionando uma visão mais abrangente da superfície e composição do objeto.

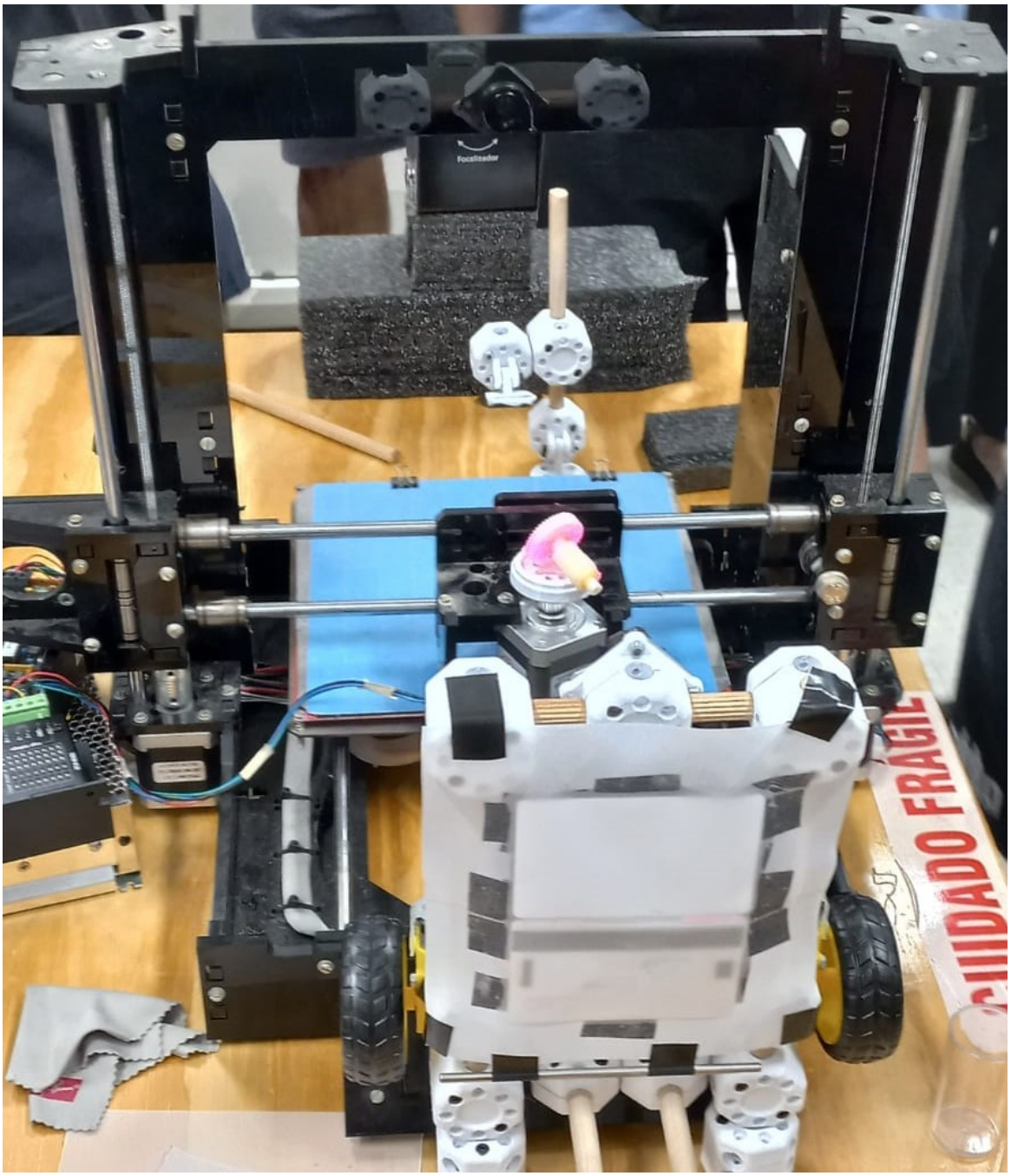

Figura 6: Tomógrafo de Luz

4.1 Geração do Sinal e Detecção

Os scanners 3D empregam a tecnologia de "varredura por projeção de luz estruturada", projetando padrões de luz conhecidos na superfície do objeto. Durante o processo de digitalização, o objeto é dividido em 15 camadas distintas, cada uma composta por 200 fotos. Mecanismos precisos iluminam cada camada à medida que a peça completa uma volta completa em incrementos precisos. A luz é então captada por câmeras estrategicamente posicionadas atrás da peça, registrando com precisão a deformação nos padrões de luz.

O mecanismo avança a cada conjunto de 200 imagens capturadas, movendo-se para uma nova camada e repetindo o processo até que 3.000 fotos sejam adquiridas. Essas imagens, após passarem por um processo digital, tornam-se vitais para a reconstrução detalhada e precisa do objeto em uma representação tridimensional. Esse processo meticuloso resulta na criação de modelos digitais altamente detalhados, conferindo grande valor em diversas aplicações, incluindo design industrial, engenharia reversa e análise dimensional avançada.

4.2 Construção da Imagem

O processo referido reflete a fase crítica de reconstrução tridimensional a partir das informações capturadas pelas imagens dos feixes gerados durante as coletas realizadas pela máquina. Cada foto, representando uma camada específica do objeto, serve como uma peça do quebra-cabeça digital, contribuindo para a recriação detalhada da superfície. Essa abordagem é meticulosamente executada por meio de codificação em Python, com ênfase na utilização da biblioteca Pillow para manipulação de imagens.

Durante a execução desse processo, o computador realiza uma análise passo a passo das imagens obtidas, implementando algoritmos que estruturam e processam cada camada individualmente. A biblioteca Pillow, conhecida por sua eficácia no processamento de imagens, desempenha um papel fundamental na manipulação e organização dessas informações visuais.

Ao seguir o fluxo de execução, cada imagem é cuidadosamente integrada no código, levando em consideração sua posição relativa e os dados capturados pelos feixes correspondentes. Esse processo iterativo e sequencial culmina na construção de um modelo tridimensional computadorizado da superfície do objeto.

Este método de reconstrução computadorizada não apenas evidencia a interseção entre hardware e software, mas também destaca a sinergia entre a captura de dados avançada e as capacidades de programação para transformar informações visuais em representações digitais tridimensionais precisas.

5 Tratamento de dados

O tratamento de dados na engenharia da computação e produção é uma prática essencial que desempenha um papel central na análise e otimização de processos, tomada de decisões estratégicas e avanço tecnológico. Envolve a coleta, limpeza e transformação de dados para extrair informações valiosas, impulsionando a inovação e a eficiência. Na engenharia da computação, a aplicação de algoritmos avançados e análise de dados é fundamental, contribuindo para a automação, monitoramento de desempenho e manutenção preditiva em ambientes industriais. A evolução constante de ferramentas e linguagens de programação, aliada ao crescente poder computacional, posiciona a engenharia da computação e produção como campos propícios para a aplicação inovadora de técnicas de tratamento de dados. Resumindo, o tratamento de dados é a base essencial para a integração entre tecnologia da informação e produção, capacitando a engenharia da computação a navegar, inovar e até orientar um mundo dominado por dados.

5.1 Estrutura Lógica

O tratamento de dados na engenharia da computação e produção é uma prática essencial que desempenha um papel central na análise e otimização de processos, tomada de decisões estratégicas e avanço tecnológico. Envolve a coleta, limpeza e transformação de dados para extrair informações valiosas, impulsionando a inovação e a eficiência. Na engenharia da computação, a aplicação de algoritmos avançados e análise de dados é fundamental, contribuindo para a automação, monitoramento de desempenho e manutenção preditiva em ambientes industriais. A evolução constante de ferramentas e linguagens de programação, aliada ao crescente poder computacional, posiciona a engenharia da computação e produção como campos propícios para a aplicação inovadora de técnicas de tratamento de dados. Resumindo, o tratamento de dados é a base essencial para a integração entre tecnologia da informação e produção, capacitando a engenharia da computação a navegar, inovar e até orientar um mundo dominado por dados.

5.2 Recorte de imagens

Após a conclusão da etapa de coleta de imagens tornou-se necessário ajustar suas dimensões para extrair apenas os filetes, utilizando a linguagem Python foi desenvolvido um código utilizando a biblioteca PIL (Pillow).

```
from PIL import Image
import os

def cortar_imagens(pasta_origem, pasta_destino):
    if not os.path.exists(pasta_destino):
        os.makedirs(pasta_destino)
    arquivos = os.listdir(pasta_origem)
    for arquivo in arquivos:
        caminho_completo = os.path.join(pasta_origem, arquivo)
        if os.path.isfile(caminho_completo) and arquivo.lower().
           endswith(('.png', '.jpg', '.jpeg')):
            # Abre a imagem
            imagem = Image.open(caminho_completo)
            largura_original, altura_original = imagem.size
            left = 1
            top = 190
            right = 399
            bottom = 203
            imagem_cortada = imagem.crop((left, top, right,
               bottom))
            caminho_destino = os.path.join(pasta_destino, arquivo
               )
            imagem_cortada.save(caminho_destino)
if __name__ == "__main__":
    pasta_origem = "/content/drive/MyDrive/conjunto3/"
    pasta_destino = "/content/drive/MyDrive/conjunto3/corteProntо
       "
    cortar_imagens(pasta_origem, pasta_destino)
```

Figura 7: Imagem cortada

O código possui o propósito de redimensionar imagens em uma pasta específica de origem, seguido pelo armazenamento das partes recortadas em uma pasta designada como destino. Tecnicamente, o código automatiza a segmentação de imagens através de um algoritmo que realiza o corte de imagens em um diretório de origem, com as partes recortadas sendo organizadas de forma sistemática em um diretório de destino predefinido. O algoritmo emprega coordenadas predefinidas como parâmetros para o processo de recorte, assegurando consistência e precisão na segmentação das imagens. Essa abordagem otimiza o fluxo de trabalho ao eliminar a necessidade de intervenção manual no processo de corte, resultando em uma manipulação automatizada de imagens mais eficiente e escalável.

5.3 Conversão para escala de cinza e Inversão da densidade óptica (Alta densidade - Brnaco)

A aplicação das técnicas de Conversão para Escala de Cinza e Inversão da Densidade Óptica (alta densidade representando a cor branca) proporciona uma significativa melhoria na visualização do objeto em questão. Este procedimento não apenas facilita a manipulação pelo programa, mas também promove uma visualização otimizada, permitindo uma observação mais clara de detalhes e variações na intensidade luminosa. Ademais, a utilização dessas técnicas simplifica o processamento computacional, contribuindo para uma análise mais eficiente e acessível do objeto, tornando-o mais facilmente perceptível e interpretável.

```
from PIL import Image
import os

def converter_tons_de_cinza_e_inverter_densidade(pasta_entrada,
   pasta_saida):
    if not os.path.exists(pasta_saida):
        os.makedirs(pasta_saida)
    arquivos = os.listdir(pasta_entrada)
    for arquivo in arquivos:
        if arquivo.lower().endswith(('.png', '.jpg', '.jpeg')):
            caminho_entrada = os.path.join(pasta_entrada, arquivo
               )
            imagem = Image.open(caminho_entrada)
            imagem_tons_de_cinza = imagem.convert('L')
            imagem_invertida = Image.eval(imagem_tons_de_cinza,
               lambda x: 255 - x)
            caminho_saida = os.path.join(pasta_saida, arquivo)
            imagem_invertida.save(caminho_saida)
if __name__ == "__main__":
    pasta_entrada = "/content/drive/MyDrive/conjunto3/corteProntо
       "
    pasta_saida = "/content/drive/MyDrive/conjunto3/Inversao"
    converter_tons_de_cinza_e_inverter_densidade(pasta_entrada,
       pasta_saida)
```

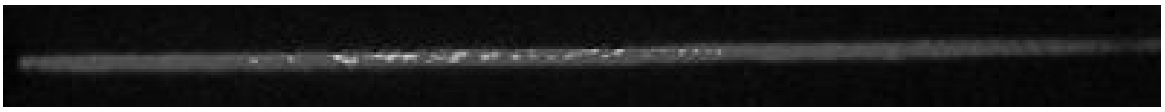

Figura 8: Conversão para escala cinza

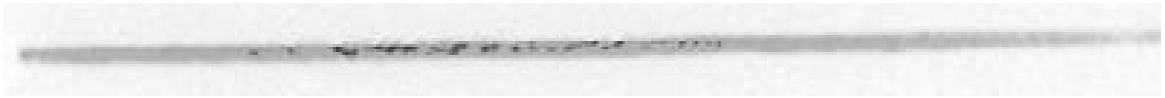

Figura 9: Inversão da densidade óptica

5.4 Rotação para 90 graus

Este script em Python utiliza a biblioteca PIL (Pillow) para rotacionar imagens contidas em um diretório de origem e salvar as versões rotacionadas em um diretório de destino. Aqui está uma explicação técnica do código:

```
import os
from PIL import Image

diretorio_origem = r'/content/drive/MyDrive/conjunto3/Inverse'
diretorio_destino = r'/content/drive/MyDrive/conjunto3/RotV3'
if not os.path.exists(diretorio_destino):
    os.makedirs(diretorio_destino)
arquivos = os.listdir(diretorio_origem)
imagens_com_erro = []
for arquivo in arquivos:
    if arquivo.lower().endswith(('.png', '.jpg', '.jpeg')):
        try:
            caminho_imagem_original = os.path.join(
               diretorio_origem, arquivo)

            imagem = Image.open(caminho_imagem_original)

            imagem_rotacionada = imagem.rotate(90, expand=True)

            caminho_imagem_rotacionada = os.path.join(
               diretorio_destino, arquivo)

            imagem_rotacionada.save(caminho_imagem_rotacionada)
        except Exception as e:
            print(f"Erro ao processar a imagem {arquivo}: {e}")
            imagens_com_erro.append(arquivo)


if not imagens_com_erro:
    print("As imagens foram salvas no diretorio.")
else:
    print("As imagens a seguir nao foram rotacionados:")
    for imagem_erro in imagens_com_erro:
        print(imagem_erro)
```

O código realiza uma tarefa simples de processamento em lote de imagens, rotacionando cada imagem em 90 graus e salvando as versões rotacionadas em um diretório diferente.

5.5 Geração de sinograma com 200 filetes (15 sinogramas)

O problema de reconstrução tomográfica na tomografia computadorizada consiste em derivar uma imagem tomográfica fatiada a partir de um conjunto de projeções. Cada projeção é formada pela integração do contraste do objeto ao longo de raios paralelos em 2D, resultando em um único pixel na projeção. Dado que uma única projeção é unidimensional, é necessário adquirir múltiplas projeções em diferentes ângulos em relação ao objeto para viabilizar a reconstrução. A compilação dessas projeções em vários ângulos é denominada sinograma, representando uma transformação linear da imagem original.

```
import os
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
diretorio_imagens = r'/content/drive/MyDrive/conjunto3/RotV3'
diretorio_salvar_sinogramas = r'/content/drive/MyDrive/conjunto3/
   sinogramV5/'
if not os.path.exists(diretorio_salvar_sinogramas):
    os.makedirs(diretorio_salvar_sinogramas)
intervalos = [(1, 200), (201, 400), (401, 600), (601, 800), (801,
    1000),
              (1001, 1200), (1201, 1400), (1401, 1600), (1601,
                 1800), (1801, 2000),
              (2001, 2200), (2201, 2400), (2401, 2600), (2601,
                 2800), (2801, 2999)]
for i, (inicio, fim) in enumerate(intervalos, start=1):
    imagens_arrays = []
    for num in range(inicio, fim + 1):
        nome_arquivo = f"{num:05d}.jpg"
        caminho_imagem = os.path.join(diretorio_imagens,
           nome_arquivo)
        if os.path.exists(caminho_imagem):
            imagem = Image.open(caminho_imagem).convert('L')
            imagens_arrays.append(np.array(imagem))
        else:
            print(f"Aviso: A imagem {nome_arquivo} nao foi
               encontrada.")
    if imagens_arrays:
        sinograma = np.stack(imagens_arrays, axis=0)
        plt.imshow(sinograma[:, :, 0], cmap='gray')
        plt.title(f"Sinograma {i}")
        plt.xlabel("Posicao do Detector")
        plt.ylabel("Angulo de Projecao")
        nome_arquivo_sinograma = f'sinograma{i}.png'
        plt.imsave(os.path.join(diretorio_salvar_sinogramas,
           nome_arquivo_sinograma), sinograma[:, :, 0], cmap='
           gray')
        plt.close()
        print(f"Sinograma {i} salvo com sucesso.")
    else:
        print(f"Erro: Nenhuma imagem foi encontrada para o
           intervalo {inicio}-{fim}. Sinograma {i} nao foi criado
           .")
```

O script automatizado realiza a aquisição das imagens rotacionadas, localizadas em um diretório de origem predefinido. A cada execução, seleciona 200 imagens e as combina para criar um sinograma. Este sinograma resultante é então armazenado no diretório previamente designado para tal finalidade.

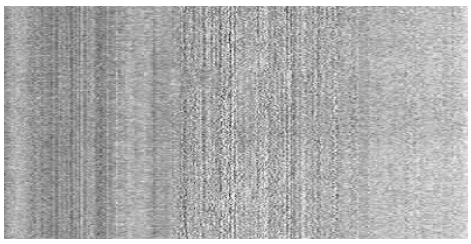

Figura 10: Sinograma 1

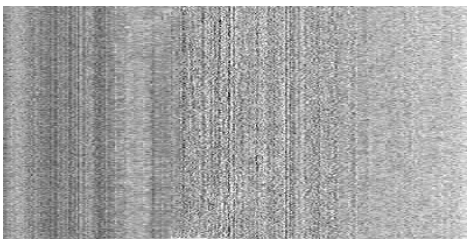

Figura 11: Sinograma 2

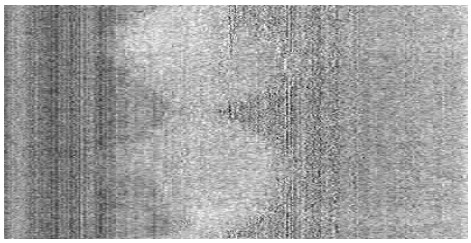

Figura 12: Sinograma 3

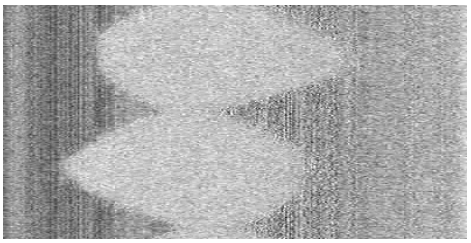

Figura 13: Sinograma 4

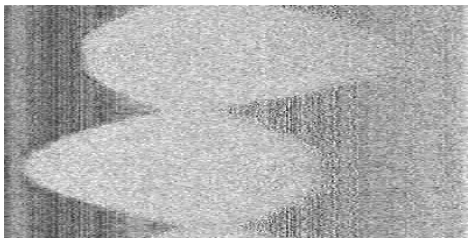

Figura 14: Sinograma 5

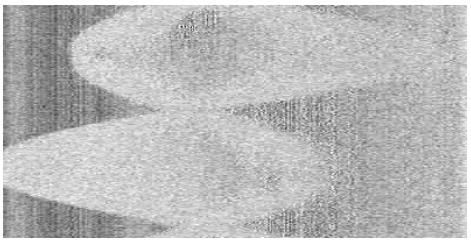

Figura 15: Sinograma 6

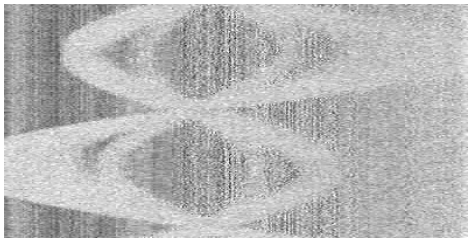

Figura 16: Sinograma 7

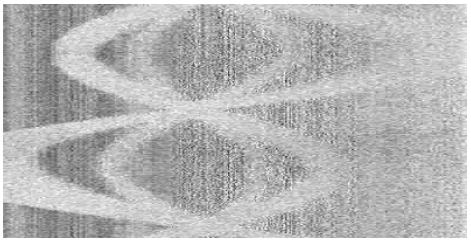

Figura 17: Sinograma 8

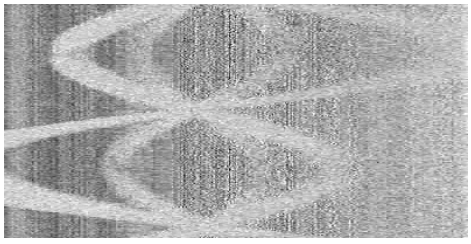

Figura 18: Sinograma 9

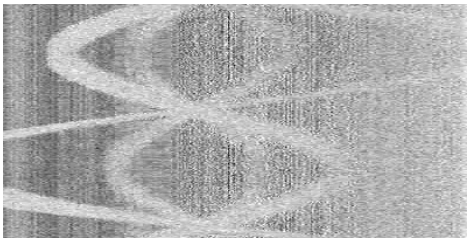

Figura 19: Sinograma 10

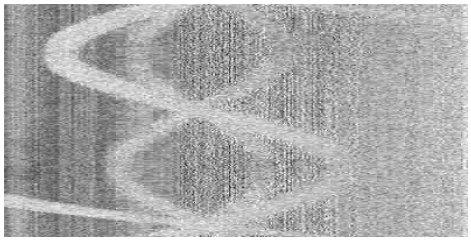

Figura 20: Sinograma 11

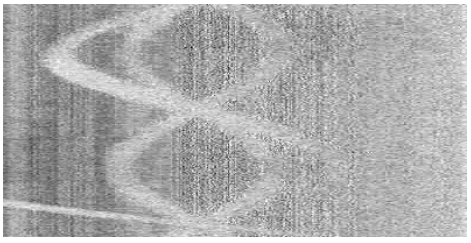

Figura 21: Sinograma 12

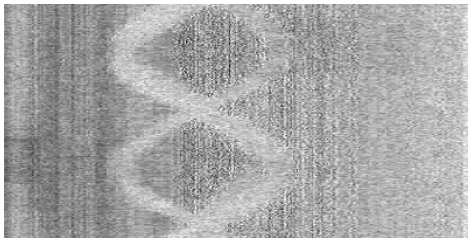

Figura 22: Sinograma 13

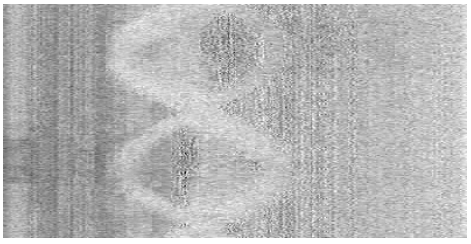

Figura 23: Sinograma 14

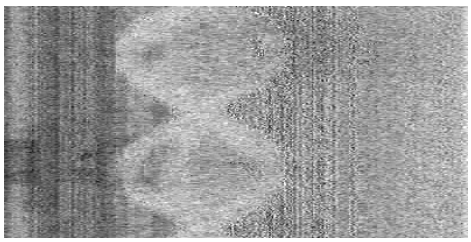

Figura 24: Sinograma 15

5.6 Inserção de randon e reconstrução de 15 planos axiais

```
import numpy as np
import matplotlib.pyplot as plt
from skimage.transform import radon, rescale
from skimage import io
from skimage.transform import iradon

file_path='/content/drive/MyDrive/conjunto3/sinogramV5/sinograma3
   .png'

image = bio.imread(file_path, as_gray=True)

# Redimensionar a imagem
image = rescale(image, scale=0.4, mode='reflect', channel_axis=
   None)


fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(8, 4.5))
ax1.set_title("Original")
ax1.imshow(image, cmap=plt.cm.Greys_r)

theta = np.linspace(0., 180., max(image.shape), endpoint=False)
sinogram = radon(image, theta=theta)
dx, dy = 0.5 * 180.0 / max(image.shape), 0.5 / sinogram.shape[0]
ax2.set_title("Radon transform\n(Sinogram)")
ax2.set_xlabel("Projection angle (deg)")
ax2.set_ylabel("Projection position (pixels)")
ax2.imshow(sinogram, cmap=plt.cm.Greys_r,
    extent=(-dx, 180.0 + dx, -dy, sinogram.shape[0] + dy),
    aspect='auto')
ax2.axis('off')

reconstructed_image = iradon(sinogram, theta=theta, circle=True)


plt.imsave('/content/drive/MyDrive/conjunto3/radonv2/radon3.png',
    reconstructed_image, cmap=plt.cm.Greys_r)

plt.figure()
plt.title("Reconstructed Image")
plt.imshow(reconstructed_image, cmap=plt.cm.Greys_r)
plt.show()
```

Este código em Python destaca uma implementação avançada da transformada de Radon em imagens, uma técnica fundamental em tomografia computadorizada. A transformada de Radon realiza projeções da imagem original em vários ângulos, gerando um sinograma que registra as intensidades em cada projeção angular. A etapa onde é implementada ao código a transformada de Radon, é essencial para capturar informações detalhadas sobre a distribuição da densidade do objeto. O sinograma resultante oferece uma representação única da estrutura do objeto. A implementação prática dessa transformada destaca sua importância na análise tridimensional de objetos por meio da obtenção de projeções em diferentes orientações angulares, proporcionando uma base sólida para estudos em tomografia e processamento de imagens.

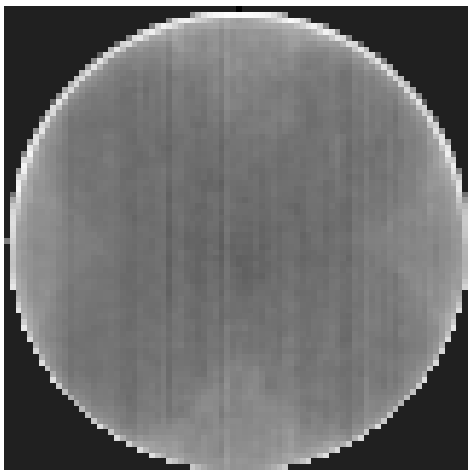

Figura 25: Radon 1

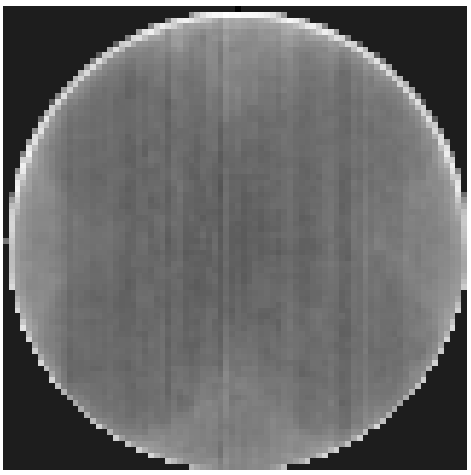

Figura 26: Radon 2

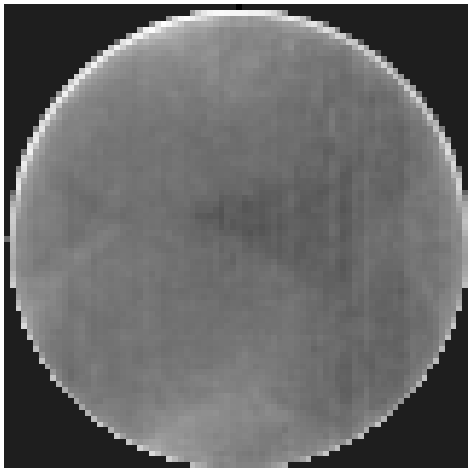

Figura 27: Radon 3

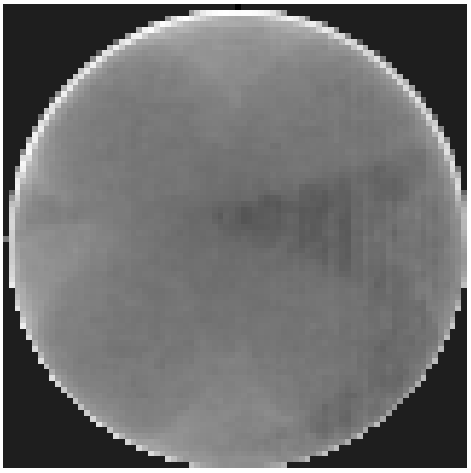

Figura 28: Radon 4

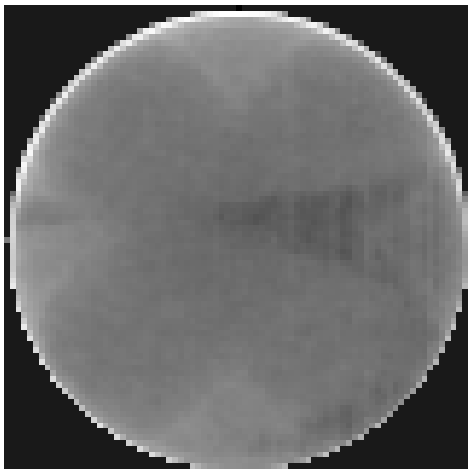

Figura 29: Radon 5

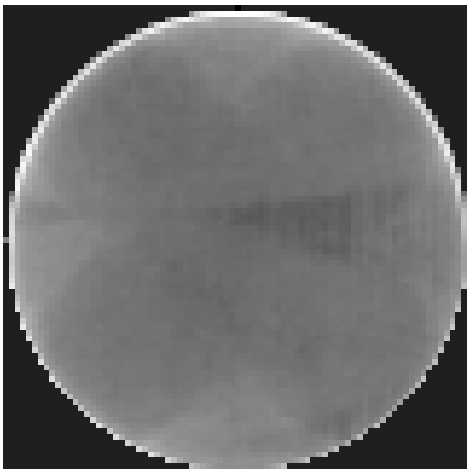

Figura 30: Radon 6

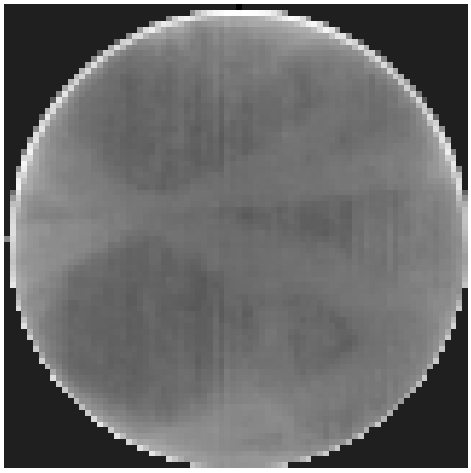

Figura 31: Radon 7

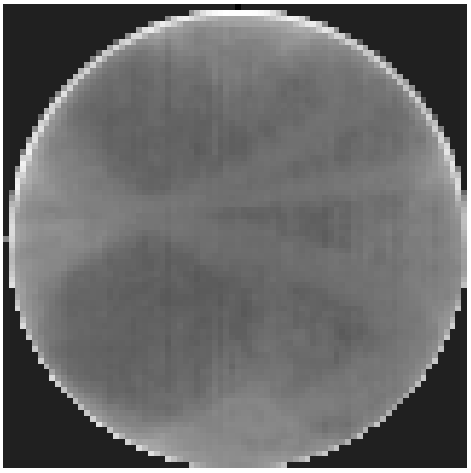

Figura 32: Radon 8

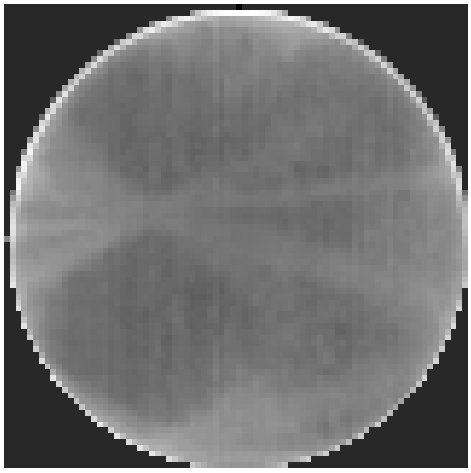

Figura 33: Radon 9

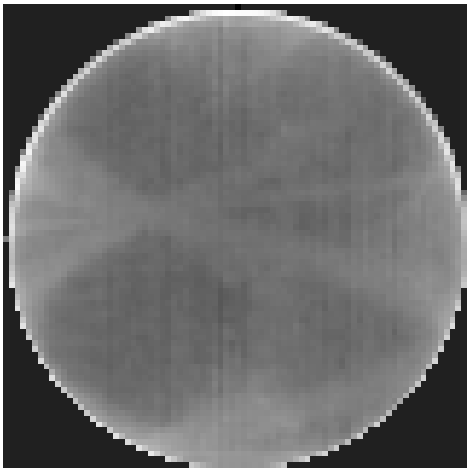

Figura 34: Radon 10

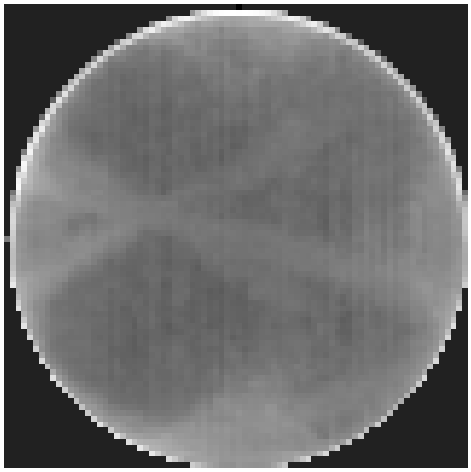

Figura 35: Radon 11

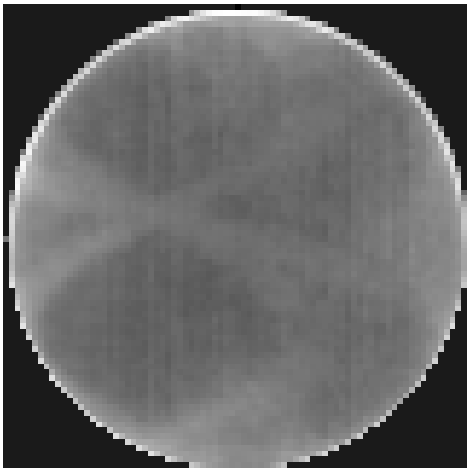

Figura 36: Radon 12

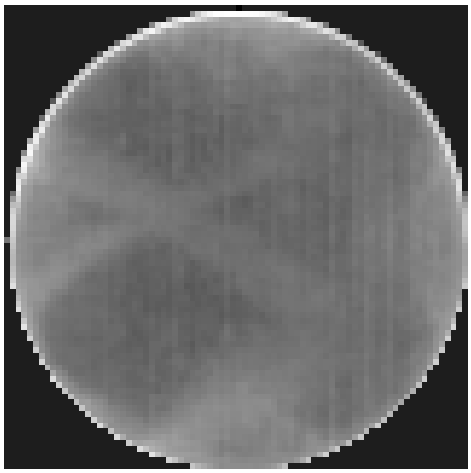

Figura 37: Radon 13

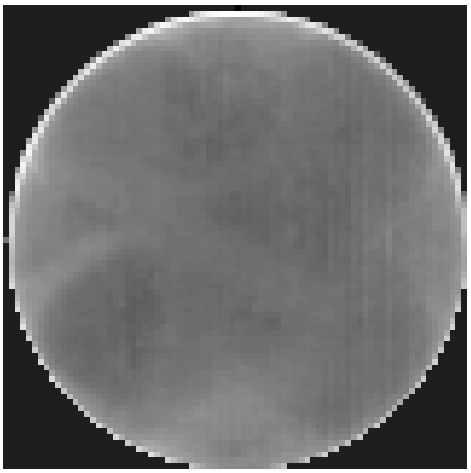

Figura 38: Radon 14

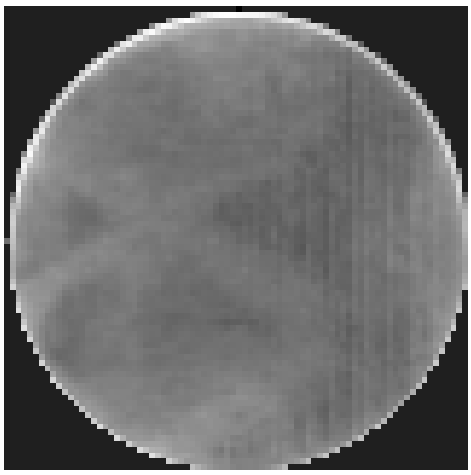

Figura 39: Radon 15

5.7 Reconstrução em 3D

A geração de objetos 3D por combinação de eixos obtidos a partir de sinogramas gerados por imagens mostrou-se um avanço significativo, embora algumas desvantagens mereçam atenção especial. Observa-se a presença de áreas pretas nas bordas das imagens, que idealmente deveriam ser eliminadas, indicando a necessidade de melhorias durante a fase de reconstrução. Além disso, a fusão dos planos axiais não parece ser completamente consistente, indicando áreas que necessitam de melhorias durante o processo de reconstrução tridimensional.

Esses desafios identificados, como áreas indesejadas e diferenças na integração das imagens planas, destacam as complexidades envolvidas na obtenção de modelos 3D precisos a partir de informações 2D. Considerando as etapas mencionadas anteriormente, desde a transformação do radon até a reconstrução, cada etapa desempenha um papel crucial na fidelidade do objeto final e qualquer impressão mesmo que na foto inicial poderiam gerar acúmulo de imperfeições que fizeram chegar nesse resultado.

Para superar essas limitações, estratégias de pós-processamento podem ser exploradas, parâmetros ajustados e algoritmos de reconstrução refinados, além de uma modificação na captura de fotos para gerar e diminuir a carga no pós-processamento. Além disso, técnicas avançadas de filtragem e suavização podem ser aplicadas para melhorar a qualidade visual dos modelos, minimizando áreas de inconsistência e maximizando a precisão nas juntas dos eixos.

Esta análise crítica de objetos não apenas destaca os desafios inerentes à reconstrução tridimensional, mas também aponta oportunidades para otimização contínua, facilitando assim o progresso em direção a modelos mais precisos e confiáveis. O processo de melhoria destas técnicas não só ajuda a melhorar a qualidade das representações, mas também expande as aplicações potenciais destes métodos em áreas como diagnóstico médico, desenho industrial e investigação científica.

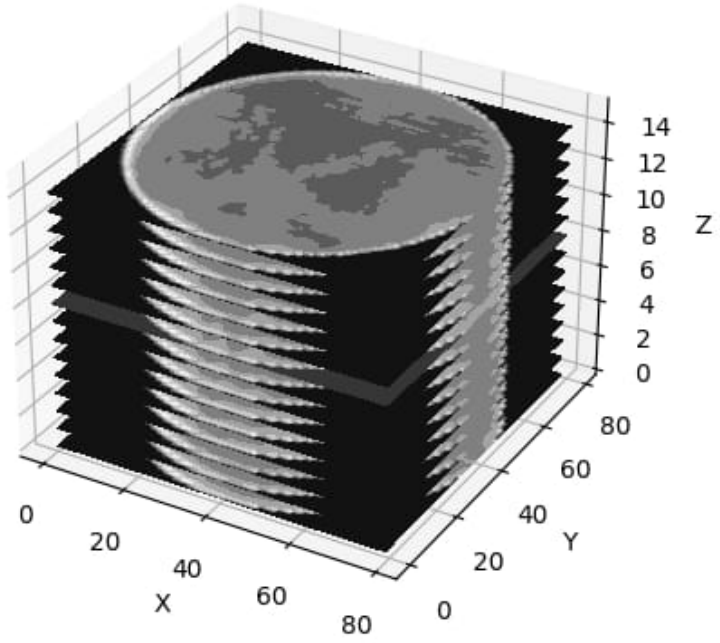

Figura 40: Tentativa de reconstrução em 3D

6 Conclusão:

Com este trabalho conseguimos ter em mente que um tomógrafo de luz representa uma ferramenta valiosa e versátil, com potencial para contribuir significativamente em setores variados, proporcionando informações detalhadas e precisas sobre estruturas internas de objetos, o que se reflete em avanços tanto científicos quanto tecnológicos. É um campo promissor que continua a evoluir, oferecendo possibilidades crescentes de aplicação e aprimoramento em diversos contextos.

É importante ressaltar que este é um projeto que engloba várias disciplinas, desde a Engenharia de Produção até a Engenharia da Computação, representando uma intersecção entre tecnologia, inovação e eficiência na produção industrial e no desenvolvimento tecnológico. É uma ferramenta valiosa que contribui para aprimorar processos, melhorar a qualidade e impulsionar pesquisas em diversas áreas. Isso bem é explorado quando falamos que na Indústria de Produção, o tomógrafo de luz pode ser aplicado para inspeção não destrutiva de peças, identificando defeitos, inconsistências ou irregularidades em materiais. Isso contribui diretamente para o controle de qualidade e, consequentemente, para a eficiência e segurança dos processos produtivos, em contraponto temos o lado da Área de Computação que por sua vez, envolve a integração de tecnologias avançadas, incluindo sensores, câmeras, processamento de imagem e algoritmos. Engenheiros da Computação têm um papel crucial na criação e otimização desses sistemas eletrônicos e na implementação de algoritmos de reconstrução tridimensional.

Durante o processo conseguimos também observar quesitos importantes para o amadurecimento dos integrantes de nosso Time em todos os quesitos, comunicação, desenvolvimento do projeto, tarefas com prazos curtos a serem cumpridos e solicitados de último momento, uma de nossas maiores dificuldades foi a produção do código em Python pois a Implementação de algoritmos eficientes para reconstrução tridimensional era complexa (tendo em mente que aprender algo deste nível em torno de uma semana era algo desafiador). Além disso, a escolha

do algoritmo correto para gerar um modelo preciso a partir das múltiplas imagens capturadas é crucial.

Para podermos avançar com nossas tarefas era importante ter ciência dos erros que acontecem ao longo do trajeto para poder ter sucesso, resultando em um melhor desempenho e alcance de objetivos comuns mesmo com nossas desavenças.

Referências

Tomografia por Emissão de Luz (LET): Zhang, H., Wu, J., Zhang, S. (2017). Light-Emitting Tomography: A Review. Sensors, 17(3), 560. Scanners 3D e Varredura por Projeção de Luz

Estruturada: Zhang, S., Zhang, H. (2012). A review on structured light pattern projection for three-dimensional shape measurement. Optics and Lasers in Engineering, 50(6), 883- 901.

Salvi, J., Fernandez, S., Pribanic, T., Llado, X. (2010). A state of the art in structured light patterns for surface profilometry. Pattern Recognition, 43(8), 2666-2680.

GRUPPETTA, Stephen. Image Processing With the Python Pillow Library. Real Python, 2023. Disponível em: https://realpython.com/image-processing-with-the-python-pillow-library/. Acesso em: 7 dez. 2023. citação: Gruppetta (2023)

IDENTIFY, No. Raspberry Pi Camera Module 2: The Raspberry Pi Camera Module 2 replaced the original Camera Module in April 2016. Raspberry Pi, 2017. Disponível em: https://www.raspberrypi.com/products/camera-module-v2/. Acesso em: 7 dez. 2023. citação: Identify (2017)

CIN (Recife). Centro de Informática da UFPE: Há quase 50 anos inovando e empreendendo: O Centro de Informática da UFPE é um dos mais renomados centros de ensino e pesquisa em computação do Brasil e da América Latina, formador de profissionais qualificados e de excelência em Tecnologia da Informação e Comunicação (TIC).. Brazil, 2023. Disponível em: https://portal.cin.ufpe.br. Acesso em: 7 dez. 2023. citação: (CIN, 2023)

ARAUJO, BR. Reconstrução Tomográfica de imagens SPECT a partir de poucos dados utilizando variação total. Orientador: Prof. Dr. Elias Salomão Helou Neto. 2017. 86 f. Dissertação (Ciências de Computação e Matemática Computacional) - ICMC - USP, São Carlos, 2017. citação: (Araujo, 2017)

COMPUTED Tomography — Reconstruction — Back Projection — Filtered Back Projection — FFT — python. Produção: Image Processing AI Projects. Kolkata: Youtube, 2022. Disponível em: https://www.youtube.com/watch?v=eW6LWmkigrc. Acesso em: 7 dez. 2023. citação: (Computed [...], 2022)

CT Scans and Tomographic Recon in PYTHON. Produção: Mr. P Solver. Canadá: Youtube, 2021. Disponível em: https://www.youtube.com/watch?v=qfAS8seVYvU. Acesso em: 7 dez. 2023. citação: (CT [...], 2021)

LOOKING through Objects - How Tomography Works!. Produção: Rafael C. Gonzalez. Alemanha: Youtube, 2021. Disponível em: https://www.youtube.com/watch?v=f0sxjhGHRPo. Acesso em: 7 dez. 2023. citação: (Looking [...], 2021)

SINOGRAM. Produção: Peter Nagy. [S. l.]: Youtube, 2020. Disponível em: https://www.youtube.com/ Acesso em: 7 dez. 2023. citação: (Sinogram [...], 2020)