Symulator tomografu

Informatyka w medycynie
Michał Miłek 151824
Sebastian Nowak 152065

13 marca 2024

1 Skład grupy

Michał Miłek (151824) & Sebastian Nowak (152065)

2 Model tomogramu

W naszym projekcie użyto stożkowego modelu tomogramu.

3 Język programowania i biblioteki

Nasz projekt został napisany w języku programowania Python, wizualizacja została wykonana za pomocą Jupyter Notebook'a. Użyte biblioteki to:

- Python Imaging Library (wczytywanie obrazu)
- NumPy (niezbędne obliczenia)
- skimage (operacje na obrazach)
- matplotlib (wizualizacja wyników)
- pydicom (zapis i odczyt plików DICOM)

4 Opis głównych funkcji programu

4.1 Pozyskiwanie wyników detektorów

Do pozyskania wyników detektorów zostały użyte dwie funkcje radon_transform oraz avg brightness (funkcja pomocniczna dla radon_transform) które zostaną pokazane w oddzielnych skrawkach poniżej.

Code Listing 1: Radon transform

```
def radon_transform(image, alpha=2, phi_range=90, num_detectors=180, num_iterations=90):
    height, width = image.shape
    radius = int(np.sqrt(height**2 + width**2) / 2)
    x_offset = width // 2
    y_offset = height // 2
    phi_rad = np.radians(phi_range)

    sinogram = np.asarray([[0 for i in range(num_detectors)] for j in range(num_iterations)])

    for i in range(num_iterations):
        curr_alpha = i * np.radians(alpha)
        xe = int(radius * np.cos(curr_alpha)) + x_offset
        ye = int(radius * np.sin(curr_alpha)) + y_offset

        for j in range(num_detectors):
            xd = int(radius * np.cos(curr_alpha + np.pi - (phi_rad / 2) + j * phi_rad/(num_detectors-1))) + x_offset
            yd = int(radius * np.sin(curr_alpha + np.pi - (phi_rad / 2) + j * phi_rad/(num_detectors-1))) + y_offset
            sinogram[i][j] = avg_brightness(line_nd([xe, ye], [xd, yd]), image)
    return sinogram
```

Code Listing 2: Average brightness

```
def avg_brightness(line, image):
    width, height = image.shape
    sum = 0
    line = list(zip(line[0], line[1]))
    for [x, y] in line:
        if (x < 0 or x >= width) or (y < 0 or y >= height):
            continue
        sum += image[x,y]
    return int(sum / len(line))
```

4.2 Uśrednianie wyniku

Uśrednianie wyniku jest w funkcji backprojection a funkcja pomocniczna w uśrednianiu (draw_line) będzie przedstawiona w następnym skrawku kodu.

Code Listing 3: Backprojection

```
def backprojection(image, sinogram, alpha=2, phi_range=90, num_detectors=180, num_iterations=90):
    height, width = image.shape
    radius = int(np.sqrt(height**2 + width**2) // 2)
    x_offset = width // 2
    y_offset = height // 2
    phi_rad = np.radians(phi_range)

    reconstruction = np.asarray([[0 for i in range(image.shape[0])] for j in range(image.shape[1])])
    pixel_counter = np.asarray([[1 for i in range(image.shape[0])] for j in range(image.shape[1])])

    for i in range(num_iterations):
        curr_alpha = i * np.radians(alpha)
        xe = int(radius * np.cos(curr_alpha)) + x_offset
        ye = int(radius * np.sin(curr_alpha)) + y_offset

        for j in range(num_detectors):
            xd = int(radius * np.cos(curr_alpha + np.pi - (phi_rad / 2) + j * phi_rad/(num_detectors-1))) + x_offset
            yd = int(radius * np.sin(curr_alpha + np.pi - (phi_rad / 2) + j * phi_rad/(num_detectors-1))) + y_offset
            reconstruction = draw_line(line_nd([xe, ye], [xd, yd]), reconstruction, i, j, pixel_counter, sinogram)

    for y, x in np.ndindex(reconstruction.shape):
        reconstruction[x][y] = reconstruction[x][y] / pixel_counter[x][y]
    return reconstruction
```

Code Listing 4: Draw line

```
def draw_line(line, image, i, j, pixel_counter, sinogram):
    height, width = image.shape
    line = list(zip(line[0], line[1]))
    for [y, x] in line:
        if (x < 0 or x >= height) or (y < 0 or y >= width):
            continue
        image[x][y] += sinogram[i][j]
        pixel_counter[x][y] += 1
    return image
```

4.3 Zapis i odczyt plików DICOM

W skrawkach poniżej będzie kolejno pokazany zapis a potem odczyt plików DICOM.

Code Listing 5: save DICOM file

```
def save_as_dicom(file_name, img, patient_data):
    img = convert_image_to_ubyte(img)
    # Populate required values for file meta information
    meta = Dataset()
    meta.MediaStorageSOPClassUID = pydicom._storage_sopclass_uids.CTImageStorage
    meta.MediaStorageSOPInstanceUID = pydicom.uid.generate_uid()
    meta.TransferSyntaxUID = pydicom.uid.ExplicitVRLittleEndian

    ds = FileDataset(None, {}, preamble=b"\0" * 128)
    ds.file_meta = meta

    ds.is_little_endian = True
    ds.is_implicit_VR = False

    ds.SOPClassUID = pydicom._storage_sopclass_uids.CTImageStorage
    ds.SOPInstanceUID = meta.MediaStorageSOPInstanceUID

    ds.PatientName = patient_data["PatientName"]
    ds.PatientID = patient_data["PatientID"]
    ds.ImageComments = patient_data["ImageComments"]

    ds.Modality = "CT"
    ds.SeriesInstanceUID = pydicom.uid.generate_uid()
    ds.StudyInstanceUID = pydicom.uid.generate_uid()
    ds.FrameOfReferenceUID = pydicom.uid.generate_uid()

    ds.BitsStored = 8
    ds.BitsAllocated = 8
    ds.SamplesPerPixel = 1
    ds.HighBit = 7

    ds.ImagesInAcquisition = 1
```

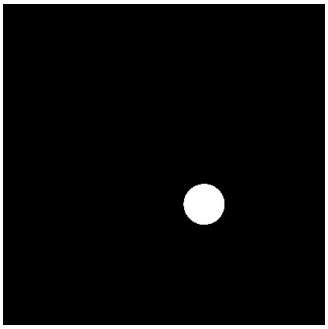
```
    ds.InstanceNumber = 1

    ds.Rows, ds.Columns = img.shape

    ds.ImageType = r"ORIGINAL\PRIMARY\AXIAL"

    ds.PhotometricInterpretation = "MONOCHROME2"
    ds.PixelRepresentation = 0

    pydicom.dataset.validate_file_meta(ds.file_meta, enforce_standard=True)

    ds.PixelData = img.tobytes()

    ds.save_as(file_name, write_like_original=False)
    print("Saved successfully")
```

Code Listing 6: read DICOM file

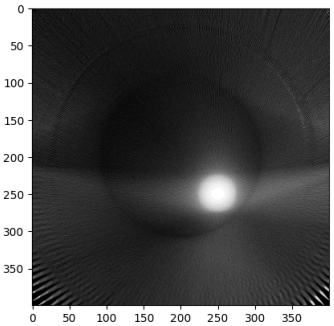
```
def read_from_dicom(path):
    ds = dcmread(path, force=True)

    # Normal mode:
    print()
    print(f"File path ........: {path}")
    print(f"SOP Class ........: {ds.SOPClassUID} ({ds.SOPClassUID.name})")
    print()

    pat_name = ds.PatientName
    print(f"Patient's Name...: {pat_name.family_comma_given()}")
    print(f"Patient ID .......: {ds.PatientID}")
    print(f"Modality .........: {ds.Modality}")
    print(f"ImageComments ....: {ds.ImageComments}")
    print(f"Image size .......: {ds.Rows} x {ds.Columns}")

    # plot the image using matplotlib
    plt.imshow(ds.pixel_array, cmap=plt.cm.gray)
    plt.show()
```

5 Przykłady działania

5.1 Przykład 1.

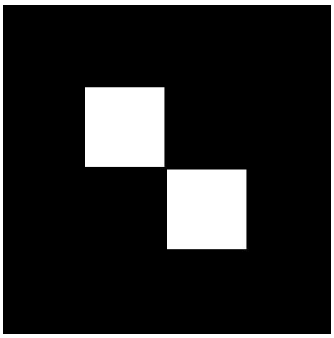


Rysunek 1: Przykładowy obraz wejściowy "Kropka.jpg"

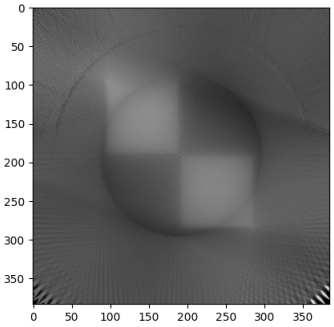


Rysunek 2: Zrekonstruowany obraz "Kropka.jpg"

5.2 Przykład 2.



Rysunek 3: Przykładowy obraz wejściowy "Kwadraty2.jpg"



Rysunek 4: Zrekonstruowany obraz "Kwadraty2.jpg"