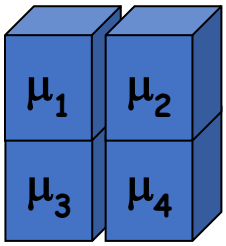
BME2104 -《生物医学影像技术》Home Work #3

Due Date: May 3, 2024

Problem 1. A CT slice image is shown below. It is known that the four voxels (μ1, μ2, μ3, μ4) correspond to water, air, tissue and bone, respectively. And μ1=664, μ2=10, μ3=750, μ4=1440. Please find the corresponding CT number of the four voxels.



$$CT\ number\ for\ \mu_1 = \frac{\mu_1 - \mu_1}{\mu_1} * 1000 = 0$$

$$CT\ number\ for\ \mu_2 = \frac{\mu_2 - \mu_1}{\mu_1} * 1000 = -984.93976$$

$$CT\ number\ for\ \mu_3 = \frac{\mu_3 - \mu_1}{\mu_1} * 1000 = 129.51807$$

$$CT\ number\ for\ \mu_4 = \frac{\mu_4 - \mu_1}{\mu_1} * 1000 = 1168.6747$$

Problem 2. An x-ray image of a mouse was taken in a 2D radiography experiment. The mouse x-ray image was saved as an image file ("1.tif"). At the exact same experimental conditions (same operating parameters for the x-ray source and detector, and acquisition geometry), a blank image ("blank.tif") and a dark image ("dark.tif") image were also taken. The blank image was taken with the x-ray source ON and without object between the source and the detector. The dark image was taken with the x-ray source OFF, which can be considered as the detector 'dark' offset when there was no x-ray. That is, the 'real' x-ray signal when the source was ON should take account of this dark offset by subtracting the x-ray images by this dark offset. All the three x-ray images ("1.tif", "blank.tif", and "dark.tif") can be found within the ZIP file "ExampleImages-ImagePreprocessing" at the Backboard site in the "Teaching Materials" folder.

Please answer the following problems, and submit your results along with your matlab/python codes.

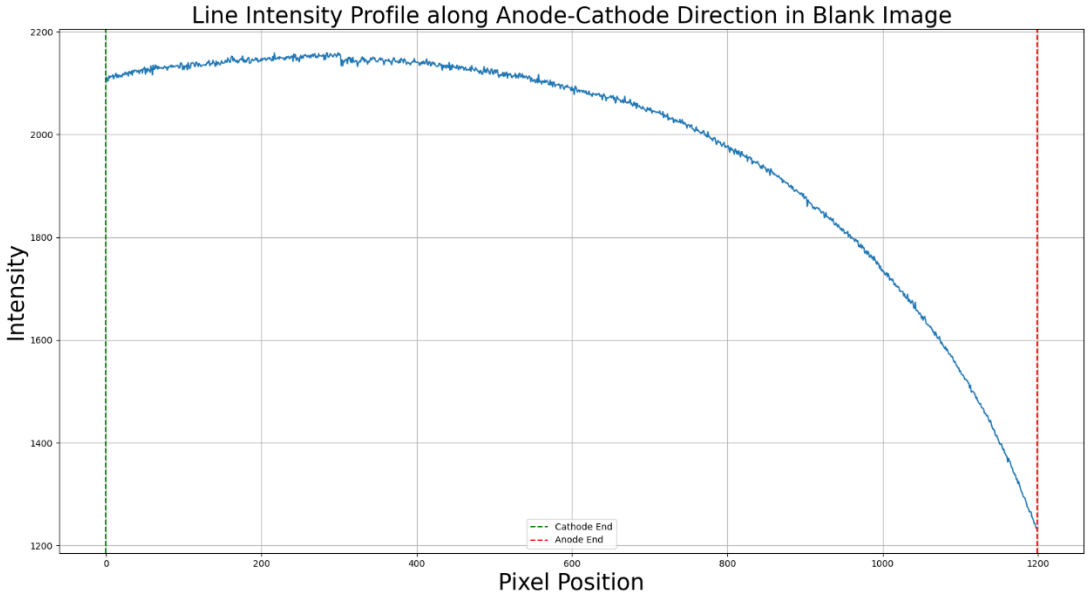
1) Pre-process the raw x-ray image (1.tif) into an acceptable radiography image of the mouse. The radiography image should be based on the amount of attenuation to x-rays by the mouse

tissues. Please show your mouse radiography image, for example using imagesc() matlab function and using proper window/level to show good contrast in the image.

```
import numpy as np
import matplotlib.pyplot as plt
from skimage import io

X = io.imread('./Data4HW3/1.tif')

Offset = io.imread('./Data4HW3/dark.tif')

X = X - Offset

Y = io.imread('./Data4HW3/blank.tif')

X = X.astype(float)
Y = Y.astype(float)

S = np.log(Y / X)

plt.figure()
plt.imshow(S, cmap='gray')
plt.title(' mouse radiography image ')
plt.colorbar()

plt.show()
```

2) Examine the amount of anode heel effect in this x-ray imaging setup, by plotting a line intensity profile along the anode-cathode direction **in the blank image.** Label in the plot the cathode end and the anode end.

```
blank_image = io.imread('./Data4HW3/blank.tif').astype(float)
line_profile = np.mean(blank_image, axis=0)

plt.figure()
plt.plot(line_profile)
plt.title('Line Intensity Profile along Anode-Cathode Direction in Blank Image')
plt.xlabel('Pixel Position')
plt.ylabel('Intensity')
plt.grid(True)


cathode_position = 0
anode_position = blank_image.shape[1] - 1
plt.axvline(x=cathode_position, color='g', linestyle='--', label='Cathode End')
plt.axvline(x=anode_position, color='r', linestyle='--', label='Anode End')
plt.legend()

plt.show()
```



The green line is Cathode End and the red line is the Anode End.

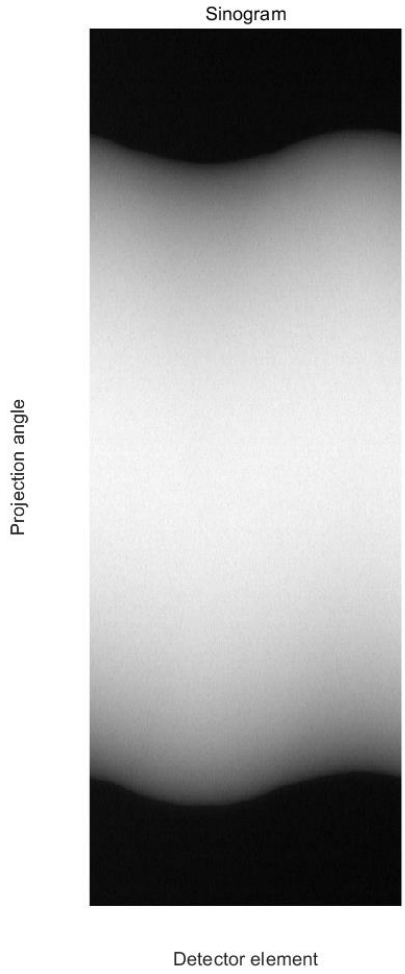
Problem 3. This CT problem focuses on image acquisition, reconstruction, and analysis. In image acquisition, you will assemble the sinogram based on the attenuation signals. Image reconstruction, the process of converting projection images (and hence sinogram) into slice images, can be carried out using the ifanbeam() function in Matlab. In image analysis, you will investigate the noise, CNR, and beam hardening artifact in the CT image.

Problem Description:

In a micro-CT scan, 360 projection images were acquired at 1 degree angular interval for a water phantom (i.e. a thin-walled cylinder filled with water). Provided are all those 360 projection images, with their names as 0001.tif …… 0360.tif (i.e. file names correspond to their respective projection angles). Also provided is the blank image, that is, the image when x-rays were on and no object was present. All images acquired by a LINE detector with 1012 pixels. All images have been corrected for the bad pixels, bad lines, and dark offset. Please complete the following tasks:

Task 1. Based on the 360 single-row projection images, and the corresponding blank image, please construct the corresponding sinogram. You can visualize the sinogram using the imshow() function. Please attach the sinogram to your report.

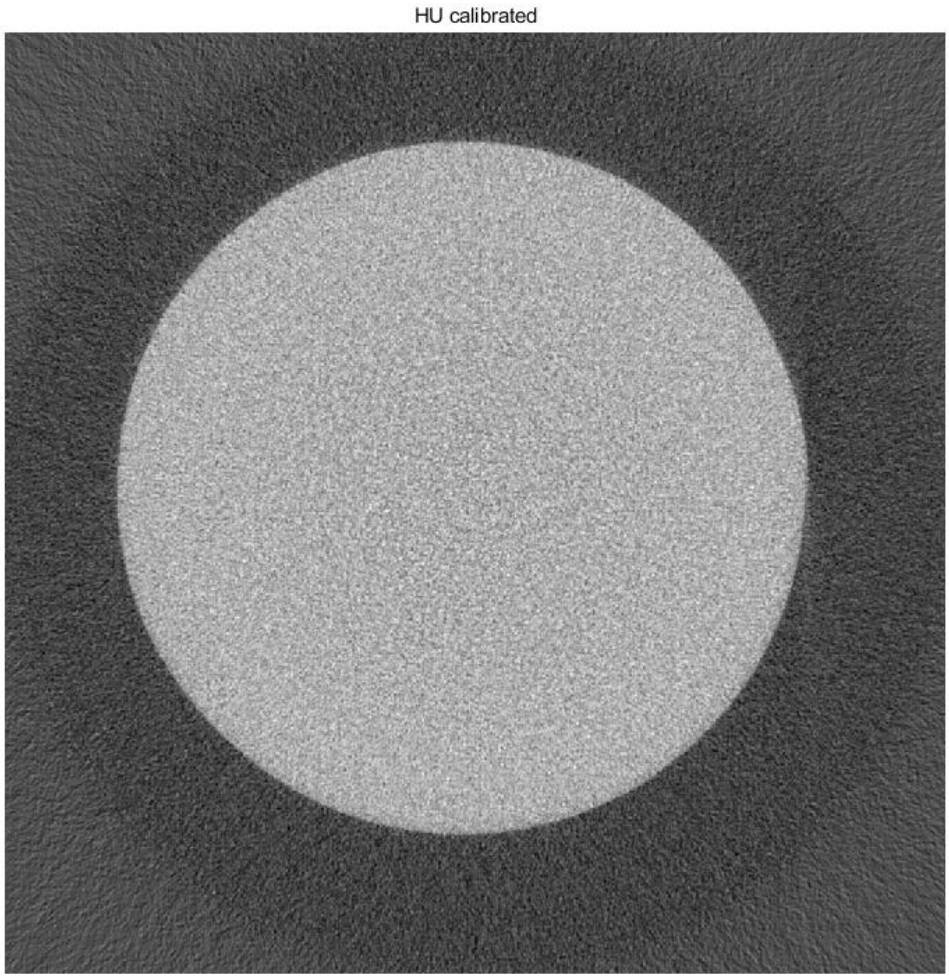
```
image_dir = './Data4CT';

blank_image = imread(fullfile(image_dir, 'blank.tif'));

num_projections = 360;
projection_images = cell(num_projections, 1);
for i = 1:num_projections
    filename = sprintf('%04d.tif', i);
    projection_images{i} = imread(fullfile(image_dir, filename));
end

sinogram = zeros(num_projections, size(projection_images{1}, 2));

for i = 1:num_projections
    sinogram(i, :) = log(double(blank_image)./double(projection_images{i}));
end
figure;
imshow(sinogram', []);
title('Sinogram');
xlabel('Detector element');
ylabel('Projection angle');
```

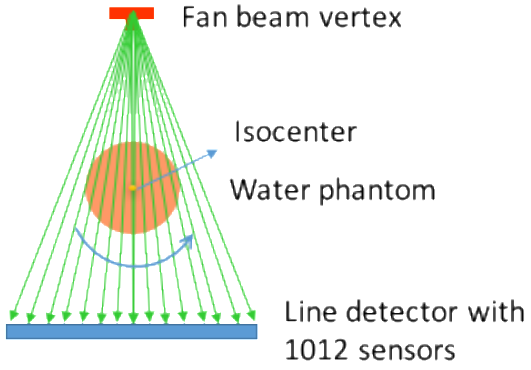


Task 2. Using the sinogram from Task 1, please reconstruct the corresponding CT slice image using the matlab function ifanbeam(), and the following geometry parameters:

```
SID = 461.43;
SDD = 625.5;
detector_pixel_size = 0.05;
number_of_projections = 360;
angular_interval_per_projection = 1;
scan_angular_coverage = 360;

D = SID / detector_pixel_size;
figure;

reconstructed_image = ifanbeam(sinogram', D, ...
    'FanSensorGeometry','line',...
    'OutputSize',1012,...
    'Filter','Hamming');


imshow(reconstructed_image, []);

x_left_top = 400;
y_left_top = 400;

```

```
x_right_top = x_left_top + 300;
y_right_top = y_left_top;

x_right_bottom = x_right_top;
y_right_bottom = y_left_top + 200;

x_left_bottom = x_left_top;
y_left_bottom = y_right_bottom;

hold on;
plot([x_left_top, x_right_top, x_right_bottom, x_left_bottom, x_left_top], ...
     [y_left_top, y_right_top, y_right_bottom, y_left_bottom, y_left_top], ...
     'r', 'LineWidth', 3);
title('Reconstructed CT Slice');
```



Source-To-Isocenter distance 461.43 mm. *Source-To-Detector distance 625.5 mm.* *Detector pixel size 0.05 mm.* *Number of projections 360.* *Angular interval per projection 1 deg.* *Scan angular coverage 360 deg.* *Center of rotation is the center point of the projections.*	

Please include your reconstructed slice image in your report.

*Hint for using ifanbeam(): check its help file at https://www.mathworks.com/help/images/ref/ifanbeam.html?requestedDomain=www.mathworks.com., and at https://www.mathworks.com/help/images/ref/fanbeam.html. The format should be

ifanbeam(F,D,param1,val1,param2,val2,...)

where each column of F contains fan-beam projection data at one rotation angle, D is the distance from the fan-beam vertex to the center of rotation. D should be expressed in number of pixels, with the pixel size being the *size of pixel in the reconstructed image at **isocenter*** (Note: not the size of the detector pixel). When specifying the 'Filter", use 'Hamming' filter.

Task 3. Based on the CT slice from Task 3, after HU calibration, please do the following:

a. Find out the noise level in the image.
b. Calculate the CNR between water and air region.
c. Draw a line intensity profile across the center of the water phantom. Can you see the beam hardening artifact? Explain why.

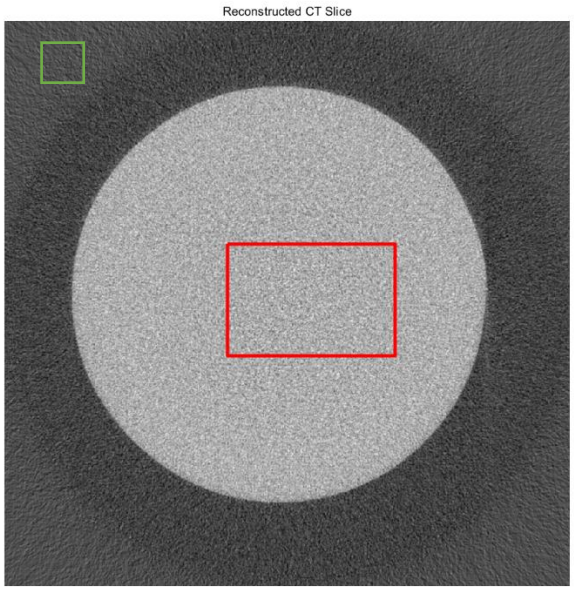
For HU calibration, the green area is the air while the red area is the water. The mean values of the ROIs can be calculated respectively. The coefficients a and b can be calculated by

$$a = \frac{1}{meanWater - meanAir} * 1000$$

$$b = \frac{-\frac{meanWater}{meanAir}}{\frac{meanWater}{meanAir} - 1} * 1000$$

The HU calibrated image can be calculated by

$$HU_{image} = a * reconstructed_{image} + b$$



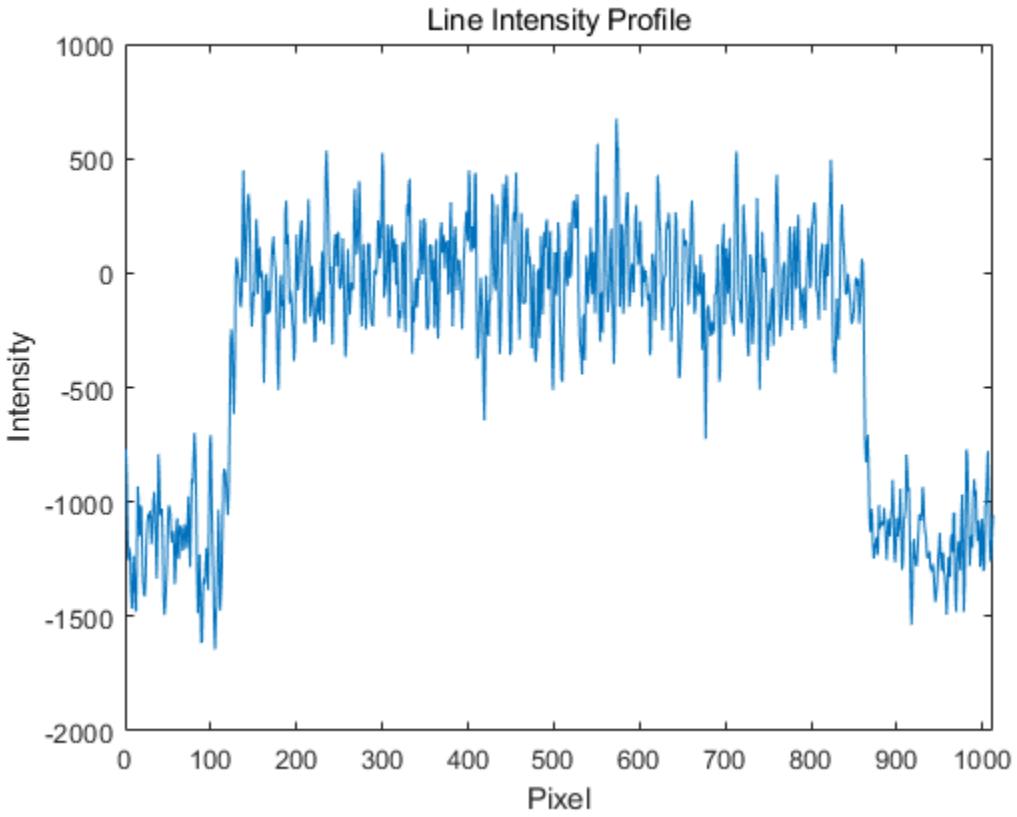
a. Noise level can be calculated by the standard deviation of the background which is 127.583659.
b. CNR can be calculated by

$$\mathrm{CNR} = \frac{|meanWater - meanAir|}{\sqrt{stdWater^2 + stdAir^2}}$$

which is 2.918166.

```
Noise Level: 127.583659
CNR between water and air: 2.918166
```

c. The line intensity profile is as follows, the beam hardening artifact cannot be seen, the reason may include noise interference or machine calibration of the original data, it is normal not to see the curve.




```
image_dir = './Data4CT';

blank_image = imread(fullfile(image_dir, 'blank.tif'));

num_projections = 360;
projection_images = cell(num_projections, 1);
for i = 1:num_projections
    filename = sprintf('%04d.tif', i);
    projection_images{i} = imread(fullfile(image_dir, filename));
end

sinogram = zeros(num_projections, size(projection_images{1}, 2));

for i = 1:num_projections
    sinogram(i, :) = log(double(blank_image)./double(projection_images{i}));
end
figure;
imshow(sinogram', []);
title('Sinogram');
xlabel('Detector element');
ylabel('Projection angle');

SID = 461.43;
SDD = 625.5;
detector_pixel_size = 0.05;
number_of_projections = 360;
angular_interval_per_projection = 1;
scan_angular_coverage = 360;

D = SID / detector_pixel_size;
```

```
figure;

reconstructed_image = ifanbeam(sinogram', D, ...
    'FanSensorGeometry','line',...
    'OutputSize',1012,...
    'Filter','Hamming');


imshow(reconstructed_image, []);


x_left_top = 400;
y_left_top = 400;

x_right_top = x_left_top + 300;
y_right_top = y_left_top;

x_right_bottom = x_right_top;
y_right_bottom = y_left_top + 200;

x_left_bottom = x_left_top;
y_left_bottom = y_right_bottom;

hold on;
plot([x_left_top, x_right_top, x_right_bottom, x_left_bottom, x_left_top], ...
     [y_left_top, y_right_top, y_right_bottom, y_left_bottom, y_left_top], ...
     'r', 'LineWidth', 3);
title('Reconstructed CT Slice');

reconstructedSlice = double(reconstructed_image);

waterROI_forCalibration = reconstructedSlice(400:700, 400:600);
airROI_forCalibration = reconstructedSlice(3:10, 3:10);

meanWater = mean(waterROI_forCalibration(:));
meanAir = mean(airROI_forCalibration(:));

airHU = -1000;
waterHU = 0;

a = (airHU - waterHU) / (meanAir - meanWater);
b = (airHU * meanWater / meanAir) / (meanWater / meanAir - 1) ;

HU_calibrated_image = a * reconstructed_image + b;

waterROI = HU_calibrated_image(200:400, 200:400);
airROI = HU_calibrated_image(3:10, 3:10);


      figure;
      imshow(HU_calibrated_image,[]);
      title('HU calibrated')

      noiseLevel = std(airROI(:));
      meanSignalWater = mean(waterROI(:));
      meanSignalAir = mean(airROI(:));
      CNR = abs(meanSignalWater - meanSignalAir) / sqrt((std(airROI(:))^2)+std(waterRO
  I(:))^2);


```

```
112.        lineProfile = reconstructedSliceHU(size(reconstructedSliceHU,1)/2, :);
113.        figure;
114.        plot(lineProfile);
115.        title('Line Intensity Profile ');
116.        xlabel('Pixel');
117.        ylabel('Intensity');
118.
119.        fprintf('Noise Level: %f\n', noiseLevel);
120.        fprintf('CNR between water and air: %f\n', CNR);
121.
122.        xlim([0,1012])
```