Lab Report

ENSC 477 - Biomedical Image Acquisition

Lab 1

Group 9

Group Member	Student ID
Benjamin Martin	301347720
Daria Zhevachevska	301326410

School of Engineering

Faculty of Applied Sciences

Simon Fraser University

Fall 2020

Introduction:

This lab explores the method of backprojection in computed tomography using emulating data that would arise from a collimated x-ray apparatus.

When constructing an image in the form of a laminogram from a sinogram using backprojection, we observe the effects of varying step sizes for the angle theta of each individual projection, the range of the angle theta, and the amount of projections recorded. Artifacts that produce glow and shadow in the laminogram are observed and contrasted with a laminogram contrasted from the same sinogram which has been high pass filtered.

In actual x-ray imaging, the intensities recorded are governed by a Poisson process which creates noise in each bar phantom. To explore the effects of noise on both the sinogram and the laminogram, simulated noise is introduced to the data. We then record the effects of dosage strength, differing amounts of noise, and adjusting window sizes for the windowed high pass filtration function when applying backprojection to a noisy data set.

Real medical x-ray images from a volumetric CT data set are explored using FIJI software. A volumetric rending, as well axial, coronal, and sagittal slices of the rendering are produced and recorded.

Part 1: Simulation of X-ray CT Acquisition

Method

Data was obtained from an apparatus that parallels x-ray acquisition without the need of exposing students to any dose of x-rays. Using a HeNe laser, light is focused using a lens to simulate the effects of collimating x-rays. Then, the light passes through a stage which holds two pins orthogonal to the light source which totally absorb light rays. Behind the stage, the light that passes through is captured and detected by a line detector. To produce a sinogram, the stage is rotated in steps of equivalent degrees for a range of angles, and the intensity of light that reaches the detectors is recorded at each angle. The pixel spacing of each detector is 4µm.

To produce a laminogram, the method of backprojection is applied to the sinogram. For each angle θ where data is recorded, the 1-D data set $g(l,\theta)$ is stretched to a 2-D array, stretching the value of $g(l, \theta)$ along the axis perpendicular to θ. Then the full set of backprojections is summed to form a laminogram.

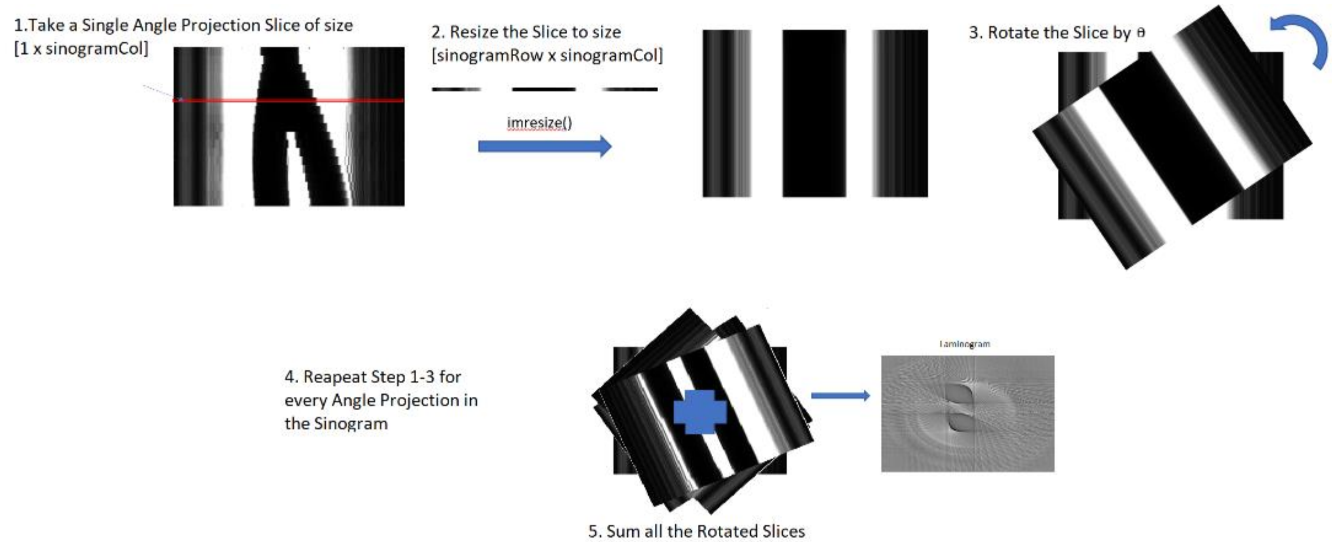

Figure 1. *Backprojection Flowchart*

Artifacts of glow and shadow are removed by applying a windowed high pass filter in the frequency domain to the sinogram before processing the data with our backprojection algorithm described above. Various cut-off frequencies of the window filter are tested.

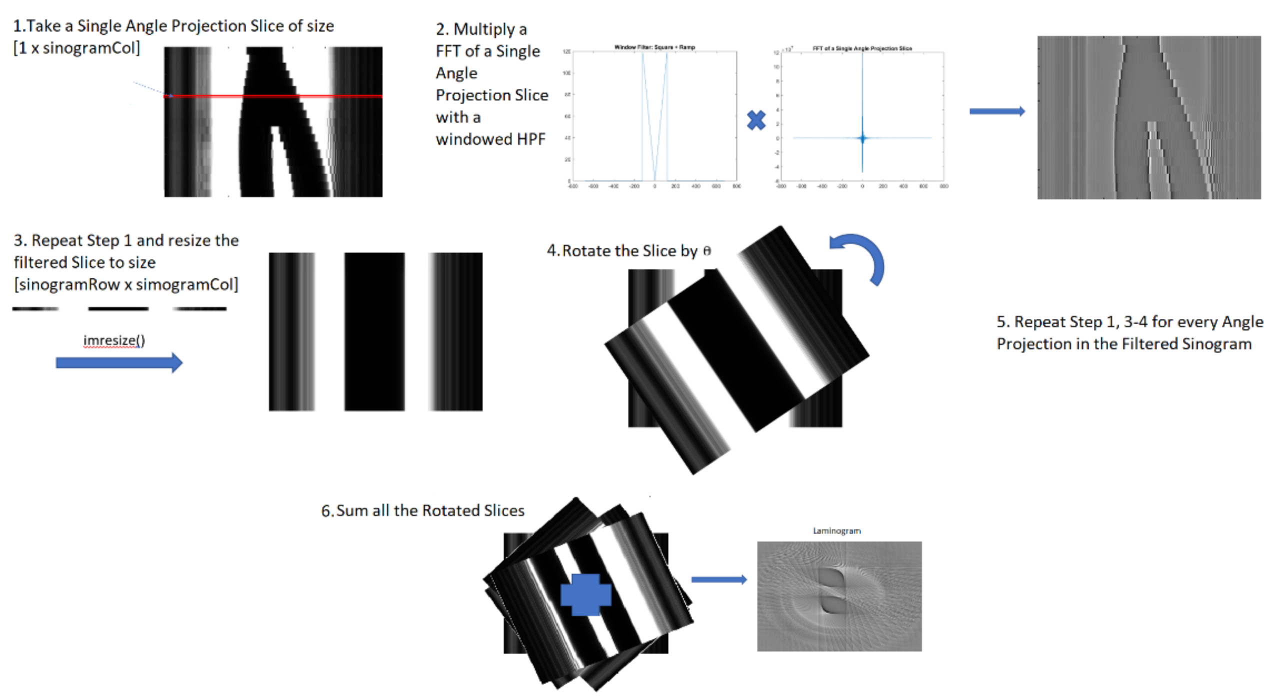

Figure 2. *Filtered Backprojection Flowchart*

Results

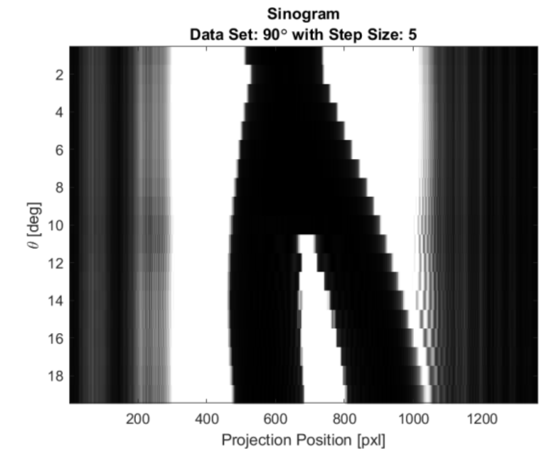

Figure 3. *The Left Trace is the Pin Closest to the Axis of Rotation*

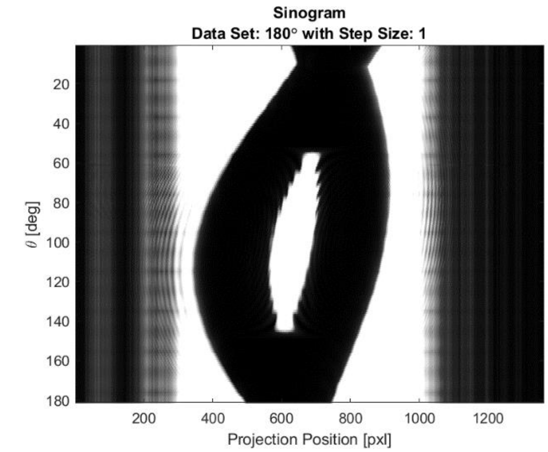

Figure 4. *The Right Trace is the Pin Closest to the Axis of Rotation*

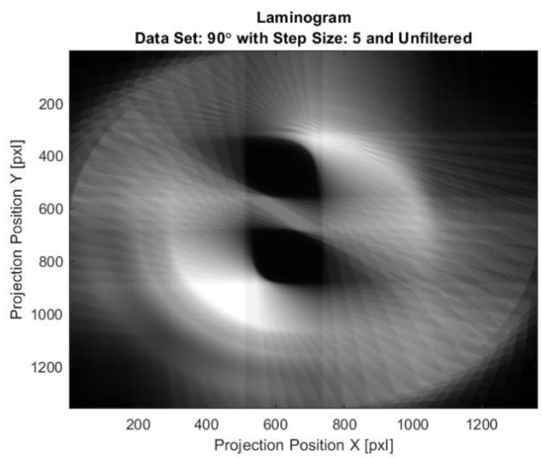

Figure 5. *Unfiltered Laminogram with Glow and Shadow Artifacts Present*

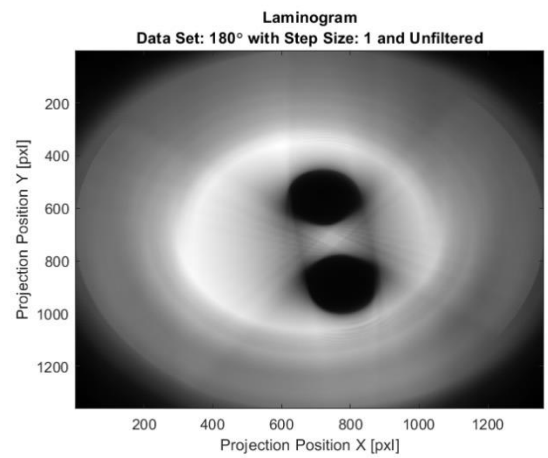

Figure 6. *Unfiltered Laminogram with Circular Glow and Shadow across the Wide Area*

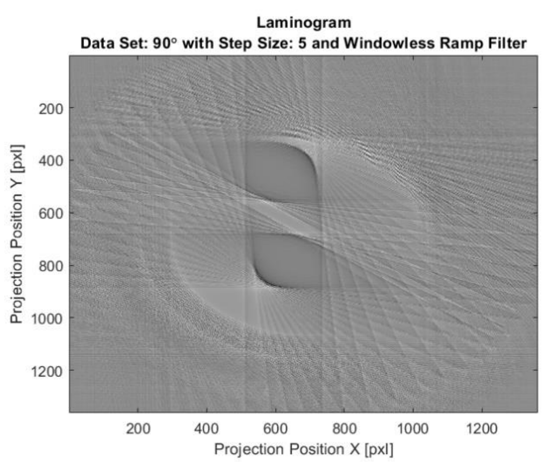

Figure 7. *Filtered Laminogram with a Ramp Filter, where Sharp Details are Preserved*

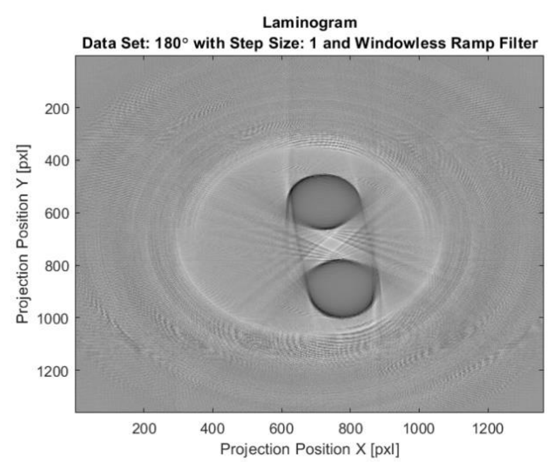

Figure 8. *Filtered Laminogram with a Ramp Filter, with Noise displayed as Rings around the Detected Pins*

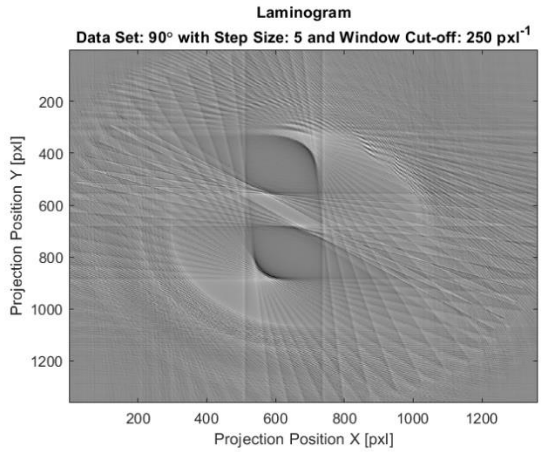

Figure 9. Filtered Laminogram with a Windowed Ramp with Removed High Frequency Blurring

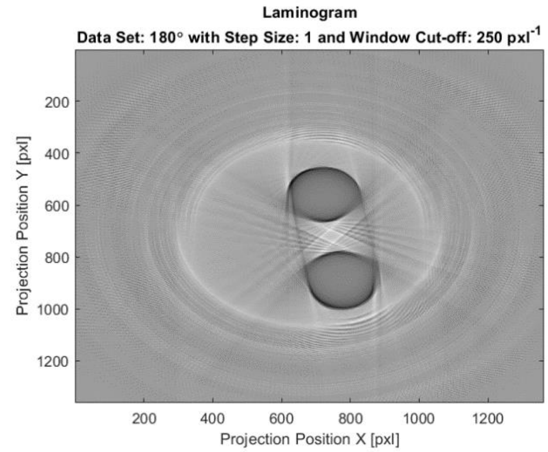

Figure 10. Filtered Laminogram with a Windowed Ramp with Removed Blurring in Large Circles at the Exteriors

Discussion

Analyzing the sinograms in Figure 3 and Figure 4, it can be seen from the overlapping absorbed section that the pins were aligned parallel to the beam when the mounting stage was at an angle ɵ of 0°. Then, as the mounted pins are rotated, the sections absorbed split as both pins are fully absorbing light. The data set sinogram with a 180° range of rotation then returns the pins to a position where they are overlapping as the light absorbed by the two pins overlaps again. The 90° data set sinogram shows the pins rotating only until the pins position are aligned orthogonal to the light source and the two separate areas where light is absorbed is distinctly displayed. It is also visible that the data set with step sizes of 1° rotation produces a smooth sinogram, where 5° steps are large enough to display significant changes in the light intensities detection which appears as blocks on the sinogram.

The initial position when data was captured, ɵ of 0°, is visible in Figures 4 through 10 with the x axis being closest to the light source. In both 90° and 180° data sets, the curvature of the pins' exteriors are well defined on the bottom left and top right corners of the laminogram. This is because those edges are first detailed by rotation giving differences of positions where light has been absorbed or not. In the 90° data set, as the mount rotates from 0° to 90°, the position of the detectors behind the pins in the laminograms' bottom right and top left corners are blocked almost entirely from light (although it is a different edge of the pins blocking it). This causes that exterior curvature to be poorly defined using a 90° rotation.

The poorly defined edge is amplified by the characteristic that any depth of pin will absolutely block light, so there is no recognition of depth. In an actual x-ray, more definition would be possible since the number of x-rays blocked by tissue is governed by the width of the tissue as well as its linear attenuation coefficient.

Contrasting the laminograms with 90° data sets, the 180° data sets show well defined curves across 180° of rotation, and in an actual x-ray there would be no need to rotate further than 180°. It is not only wasteful to rotate further than 180° but potentially harmful as x-ray dosage increases. A 180° x-ray data capture is a flipped data set to a 0° data set. Since our apparatus has complete absorption, the sides of the pins closest to each other are still not ideally defined at 180°.

During naive backprojection, artifacts arise that display as both glow and shadow. As the apparatus rotates, it may take many rotations before a specific detector switches from receiving light to receiving none (or the opposite). Measured light intensities across the line are stretched in back projection, so an object's absorption (or not) influences the entire space along the axis of the image when in reality it should only define the absorption where the object exists. As the rotated backprojections are summed, those constants form a circular glow (where the recorded light intensity is high) or a shadow effect (low intensity) across a wide area. Spatially, these are low frequency signals. So, applying a high pass filter in the frequency domain to a Fourier transformed sinogram filters out these signals, while retaining the high frequency signals responsible for sharp details.

Although resolution increases without filtering out any high frequency spatial signals, a window is included in filtration for practical purposes. The integral of an inverse Fourier transform of an ideal ramp function does not converge. High enough frequencies become undetectable, as the distance between two figures falls under the contrast definition of Full Width Half Maximum. Motion artifacts create high frequency blurring and can be removed without significant loss of definition. Noise in detecting pure light is visible as a ring around the windowless filtered laminograms where no light has been absorbed by the pins at any angle of rotation. Through trial, a windowed filter with a cut-off frequency of 250 pixels^{-1} was found to effectively retain resolution while removing some of the blurring in large circles at the exteriors of the laminograms.

Part 2: Effects of Noise and Filtering in X-ray CT

Method

The data with a low zero noise and high intensity is used to perform CT reconstruction of a human abdomen. Since the initial data is a very unrealistic case for real medical imaging, the effects of adding noise and lowering the dose are examined. Because photon noise follows a Poisson distribution, the noise is added to the initial sinogram with the help of a MATLAB inbuilt function `imnoise(…, 'poisson')`. The dose level is lowered by simply dividing the noisy data by a constant.

Reconstructed images are compared with and without filtering, in the absence and presence of noise and varying amounts of dose. Backprojection and filtered backprojection are performed as in Part 1.

Results

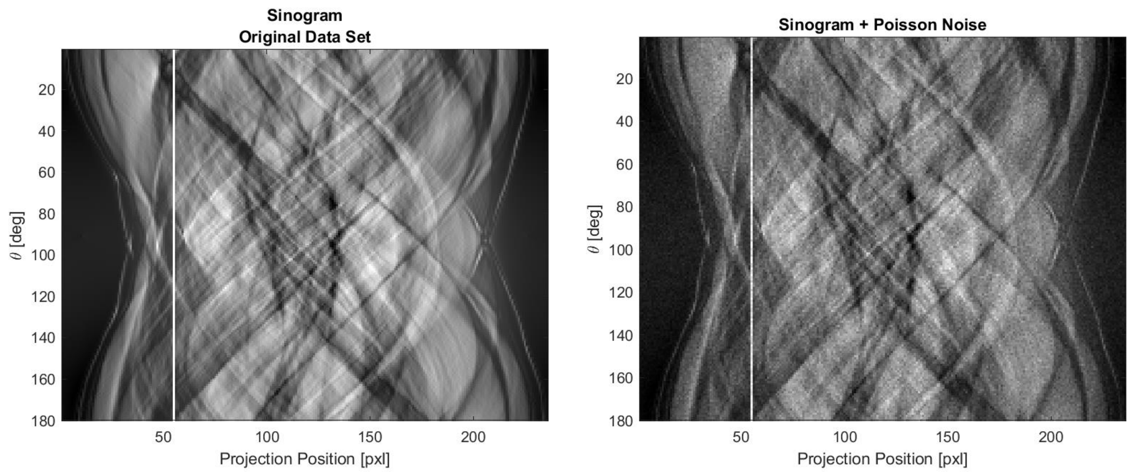

Figure 11. *Sinogram of the Original Data (left), Sinogram with Poisson Noise (right)*

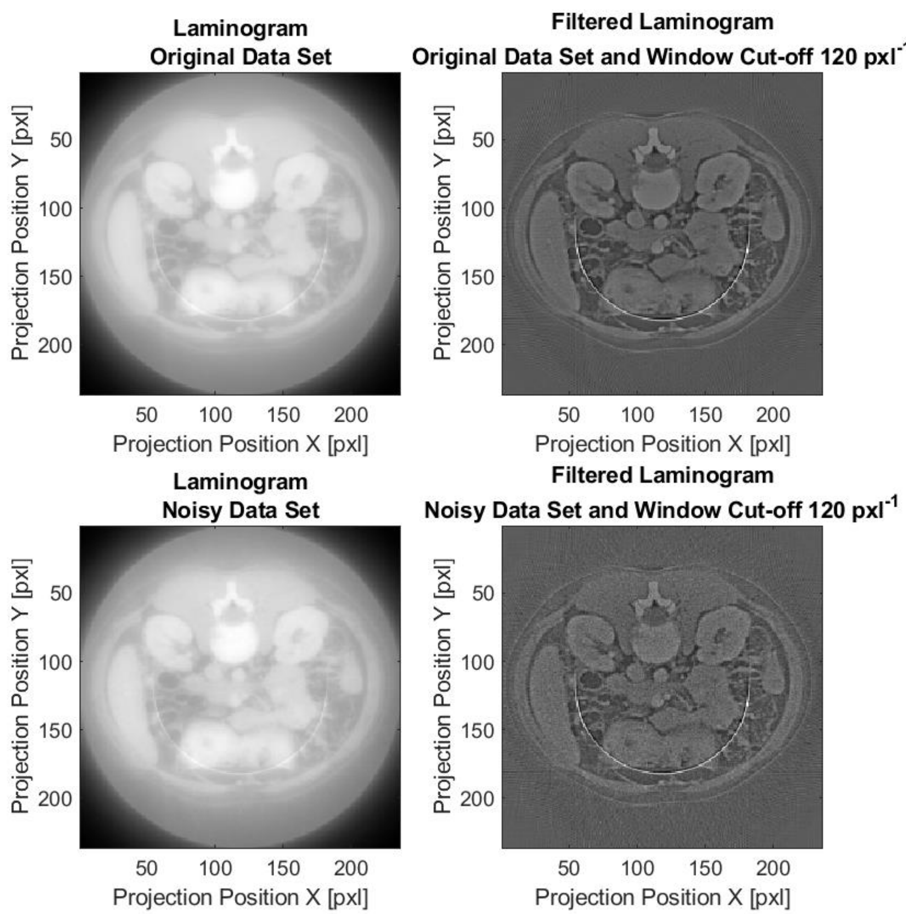

Figure 12. *Laminogram and Filtered Laminogram of the Original Data (top row) and the Noisy Data (bottom row)*

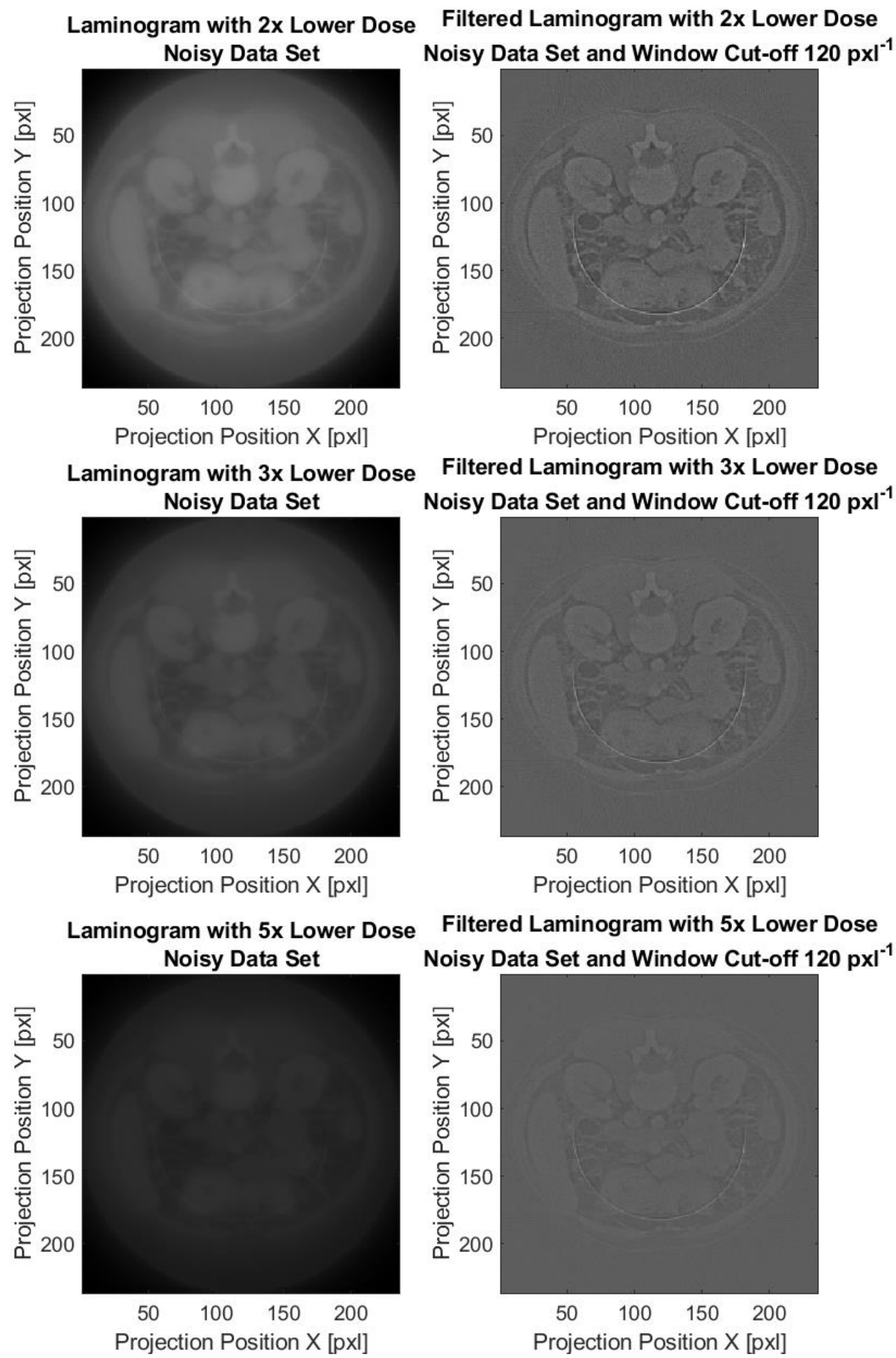

Figure 13. *Laminogram and Filtered Laminogram of the Noisy Data with 2x Lower Dose (1st row), 3x Lower Dose (2nd row) and 5x Lower Dose (3rd row)*

Discussion

The original data is given as a high intensity and low noise signal. This is an unrealistic case in CT acquisition, since there is always noise present in the signal. Firstly, the main source of noise comes from the quantum noise properties of x-ray photons, in addition, CT machine introduces the unwanted electronic noise. Secondly, as radiation dose is decreased, image noise increases. The reason for doing so is because during the CT exams patients are exposed to a significant amount of radiation doses, which in return creates a risk of having cancer.

Therefore, more realistic data sets are created by adding Poisson noise and using a lower radiation dose. In response to such changes the Signal-to-Noise Ratio (SNR) of the image is affected. In general, SNR describes the relative strength of signal to noise. In imaging the signal is represented by the contrast and noise is the random fluctuations. SNR is used to characterize the medical imaging system. Higher SNR values correspond to better image quality, whereas lower values to worse images.

Figure 12 and Figure 13 depict the effects of both noise and blur respectively. When comparing the results of backprojection and filtered backprojection of the original data and data with noise added, the quality of the reconstructed image decreases rapidly with increasing noise contamination. Sinograms with lower dose return blurred CT images, this blur causes lower contrast of the image consequently, lower SNR. That being so, noise and low dose reduce the overall quality of the reconstructed image and thus, reduces the SNR.

One consideration for increasing SNR while maintaining dosage is the bandwidth of the window used in the filter. The data measured in the sinogram is discrete since the signal is measured once by each detector, and when displayed each pixel width averages one or more samples. The pixel width acts as a low pass filter, with high spatial frequencies appearing under sampled. The pixel's low pass effect limits the usefulness of a wide window for filtration. Since the bandwidth of the window is proportional to the standard deviation of the noise in CT, lowering the bandwidth increases SNR. Once the window bandwidth is smaller than the low pass effect of the pixels, the SNR will still increase with lowered bandwidth, but the image will also lose resolution.

Part 3: CT Visualization

Method

A volumetric CT data of head-to-waist is download from National Institutes of Health database. An axial, coronal, sagittal slices and a volumetric body mock-up are generated and displayed using FIJI, a 3D visualization software.

Results

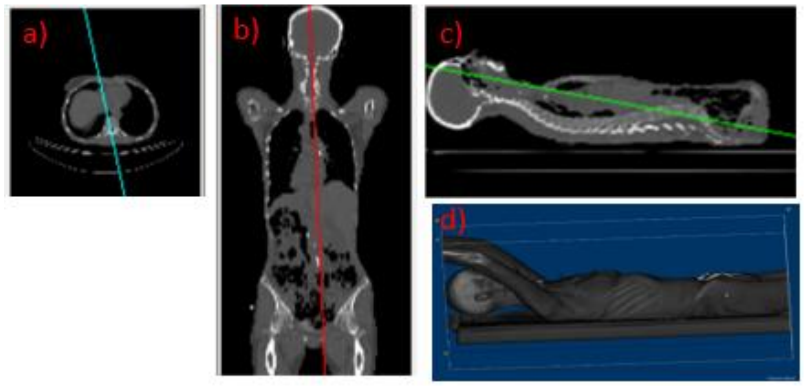

Figure 14. a) Axial View b) Coronal View c) Sagittal View d) Volumetric Body Mock-Up

Discussion

CT volumetric rendering techniques allow us to visualize the human body in finer details, which permits better medical treatment planning, such as a surgery. The high-volume coverage and continuous rotation of the detector gives the opportunity to see the important details in between the slices.

Figure 14 included in the Results displays the three standard 2D projections used in medical imaging and a 3D body mock-up. Figure 14 a) is an axial slice, which is parallel to the ground and divides the body into top and bottom parts, b) is the coronal slice and it divides the body into front and back, c) is the sagittal slice, a plane that divides the body into a left and a right halves, d) is a reconstructed 3D head-to-waist body model, that can be rotated by 360˚ to provide a better spatial view of the patient's anatomy.

Appendix:

```
%%%%%%%%%%%%%% Q1 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Load Data
clc; clear; close all;
addpath ('C:\Users\Dasha\Desktop\Fall 2020\ENSC
477\Labs\Lab1-XrayCT\');
sinogram90_5 = load('Data\90deg_5DegStep.csv');
sinogram = sinogram90_5';
step = 5;
deg = 90;

%% Display the Sinogram
figure;
imagesc(sinogram); colormap('gray');
title({['Sinogram'] ['Data Set: ', num2str(deg), '\circ
with Step Size: ', num2str(step)]});
xlabel('Projection Position [pxl]'); ylabel('\theta
[deg]');
% print(gcf,'-dpng',['Figure1c_Question1' '.png']); %save
fig

%% Backprojection
laminogram = backProjection(sinogram, step);
figure;
imagesc(laminogram); colormap('gray'); axis off;
title({['Laminogram'] ['Data Set: ', num2str(deg), '\circ
with Step Size: ', num2str(step)]});
% print(gcf,'-dpng',['Figure2c_Question1' '.png']); %save
fig

%% Filtered Backprojection
%Select the type of the window: 'square', 'hamming', 'cos'
as input value
%for the filteredBackProjection function
fltrLaminogram = filteredBackProjection(sinogram, step,
'square');
figure;
imagesc(fltrLaminogram); colormap('gray'); axis off;
title({['Filtered Laminogram'] ['Data Set: ', num2str(deg),
'\circ with Step Size: ', num2str(step)]});
% print(gcf,'-dpng',['Figure3c_Question1' '.png']); %save
fig
```

```
%%%%%%%%%%%%%% Q2 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Load Data
clc; clear; close all;
addpath ('C:\Users\Dasha\Desktop\Fall 2020\ENSC
477\Labs\Lab1-XrayCT\');
data = load('sino1.csv')';
sinogram = data';

%% Display the Sinogram
figure;
imagesc(sinogram); colormap('gray');
title({['Sinogram'] ['Original Data Set']});
xlabel('Projection Position [pxl]'); ylabel('\theta
[deg]');
print(gcf,'-dpng',['Figure1_Question2' '.png']); %save fig

%% Backprojection & Filtered Backprojection
laminogram = backProjection(sinogram, 1);

%Select the type of the window: 'square', 'hamming', 'cos'
as input value
%for the filteredBackProjection function
fltrLaminogram = filteredBackProjection(sinogram, 1,
'square');

figure;
colormap('gray');
subplot(1, 2, 1)
imagesc(laminogram); axis square;
title({['Laminogram'] ['Original Data Set']});
xlabel('Projection Position X [pxl]');
ylabel('Projection Position Y [pxl]');
subplot(1, 2, 2)
imagesc(fltrLaminogram); axis square;
title({['Filtered Laminogram'] ['Original Data Set and
Window Cut-off 120 pxl^{-1}']});
xlabel('Projection Position X [pxl]');
ylabel('Projection Position Y [pxl]');
% print(gcf,'-dpng',['Figure2_Question2' '.png']); %save
fig

%% Add Noise
%Poisson distribution depends on the data type.
```

```
%Sinogram is double precision, then input pixel values are
interpreted
%as means of Poisson distributions scaled up by 1e12.
noiseSinogram = imnoise(sinogram/1e12, 'poisson')*1e12;

figure;
imagesc(noiseSinogram); colormap('gray');
title({['Sinogram + Poisson Noise']});
xlabel('Projection Position [pxl]'); ylabel('\theta
[deg]');
% print(gcf,'-dpng',['Figure3_Question2' '.png']); %save
fig

%Comapare Backprojection and Filtered Backprojection of the
noisy sinogram
noiseLaminogram = backProjection(noiseSinogram, 1);

%Select the type of the window: 'square', 'hamming', 'cos'
as input value
%for the filteredBackProjection function
fltrNoiseLaminogram = filteredBackProjection(noiseSinogram,
1, 'square');

figure;
colormap('gray');
subplot(1, 2, 1)
imagesc(noiseLaminogram); axis square;
title({['Laminogram'] ['Noisy Data Set']});
xlabel('Projection Position X [pxl]');
ylabel('Projection Position Y [pxl]');
subplot(1, 2, 2)
imagesc(fltrNoiseLaminogram); axis square;
title({['Filtered Laminogram'] ['Noisy Data Set and Window
Cut-off 120 pxl^{-1}']});
xlabel('Projection Position X [pxl]');
ylabel('Projection Position Y [pxl]');
% print(gcf,'-dpng',['Figure4_Question2' '.png']); %save
fig


%% Change the Dose
%For obtaining a lower dose sinogram, select a constant
(fctr)
%to divide the original data in the presence of noise
```

```
fctr = 5;
doseSinogram = noiseSinogram/fctr;

%Comapare Backprojection and Filtered Backprojection of the
noisy sinogram
%with lower dose
noiseLaminogramLD = backProjection(doseSinogram, 1);
%Select the type of the window: 'square', 'hamming', 'cos'
as input value
%for the filteredBackProjection function
fltrNoiseLaminogramLD =
filteredBackProjection(doseSinogram, 1, 'square');

%Find the min and max intensity values where values less
than or equal to
%cmin map to the first color in the colormap and values
greater than or
%equal to cmax map to the last color in the colormap.
cmax1 = max(noiseLaminogram(:));
cmin1 = min(noiseLaminogram(:));
cmax2 = max(fltrNoiseLaminogram(:));
cmin2 = min(fltrNoiseLaminogram(:));

figure;
colormap('gray');
subplot(1, 2, 1)
imagesc(noiseLaminogramLD, [cmin1 cmax1]); axis square;
title({['Laminogram with ', num2str(fctr), 'x Lower
Dose']...
    ['Noisy Data Set']});
xlabel('Projection Position X [pxl]');
ylabel('Projection Position Y [pxl]');
subplot(1, 2, 2)
imagesc(fltrNoiseLaminogramLD, [cmin2 cmax2]); axis square;
title({['Filtered Laminogram with ', num2str(fctr), 'x
Lower Dose']...
    ['Noisy Data Set and Window Cut-off 120 pxl^{-1}']});
xlabel('Projection Position X [pxl]');
ylabel('Projection Position Y [pxl]');
print(gcf,'-dpng',['Figure5c_Question2' '.png']); %save fig
```

```
%%%%%%%%%%%%%% backProjection()%%%%%%%%%%%%%%%%%%%%%
function laminogram = backProjection(sinogram, step)
[row, col] = size(sinogram);
laminogram = zeros(col, col);

%Resize, rotate and sum all the projections
for ii = 1:row
    resizeProj = imresize(sinogram(ii,:), [col, col],
'nearest');
    rotProj = imrotate(resizeProj, (ii-1)*step, 'bilinear',
'crop');
    laminogram =  laminogram + rotProj;
end
end

%%%%%%%%%%%%%% filteredBackProjection()%%%%%%%%%%%%%%%%%
function laminogram = filteredBackProjection(sinogram,
step, w)
[row, col] = size(sinogram);
laminogram = zeros(col, col);

%Define ramp function
midPoint = floor(col/2) + 1;
ramp = abs(linspace(1-midPoint, midPoint, col)); %This
elegant definition was Ben's idea :)

%Define the window according to the type
filt = zeros(1, col);

if w == "square"  %Square Window
    rho = 120; %cut-off freq
    filt = ramp;
    filt(1:midPoint - rho) = 0;
    filt(midPoint + rho:end) = 0;
%     figure(2);
%     plot(x, filt);  title('Window Filter: Square +
Ramp');
elseif w == "hamming"  %Hamming Window
    w = hamming(col)';
    % figure(3);
    % plot(x, w); title('Hamming');
    filt = ramp .* w;
    % figure(4);
```

```
    % plot(x, filt);  title('Hamming + ramp');
elseif w == "cos"  %Cos Window
    cosW = cos(linspace(-pi/2, pi/2, col));
    % figure(5);
    % plot(x, cosW); title('Cos');
    filt = ramp .* cosW;
    % figure(6);
    % plot(x, filt); title('cos + ramp');
end

%Perform Filtered Backprojection
for ii = 1:row
    %Multiply filter with fft of the single projection
    FFTSngm = fftshift(fft2(sinogram(ii,:)));
    filterSngm = FFTSngm .* filt;

    %Inverse the multiplication above and take only real
part
    filterSngm = real(ifft2(ifftshift(filterSngm)));

    %Resize, rotate and sum all the projections
    resizeProj = imresize(filterSngm, [col, col],
'nearest');
    rotProj = imrotate(resizeProj, (ii-1)*step, 'bilinear',
'crop');
    laminogram =  laminogram + rotProj;
end
end
```