# Lab 3: Grey Level Operations

Daniel Sherman (0954083), *Student, University of Guelph*

***Abstract*—Grey Level operations were explored to increase contrast in images, and to segment image features by inspection. Given point functions were utilized on test images to discover the effects. Histogram equalization was performed for a given histogram, and the effects of the found equalization function was observed on the other images. One test image's histogram was found using MATLAB commands and was segmented into four classes by inspection to figure out the area of features. Finally, a custom histogram routine without using any commands in MATLAB is proposed.**

***Index Terms*—Grey Level Transformation, Point Operations, Histogram, Histogram Equalization**

## I. Introduction

WHILE applying filters like average, Gaussian, or median can highlight and display image features, they change, or distort the geometric relationships within the image. Sometimes, it is desirable to preserve geometric relationships. It may be desirable to modify the contrast of certain Grey Level ranges to enhance a feature for inspection, or to saturate some Grey Level ranges as they may not be necessary to differentiate in an image.

Especially in medical imaging, modalities such as x-ray or CT require careful studying by Radiologists to determine if there is a break in a bone or a tumor in a mammogram, as examples. Radiologists often choose to distort Grey Levels by applying a Sigmoidal Grey Level transformation to saturate high and low Grey Levels, while expanding the middle Grey Levels. This essentially increases the contrast of that middle region, and does so approximately linearly, so the gradient is easier for Radiologists to read.

Grey Level operations differ from filtering, as a Grey Level operation focuses on single pixel values, and the transformed pixel value solely depends on the original pixel Grey Level, whereas traditional filters take a neighbourhood of pixel values to determine the transformed pixel value.

In this lab, Grey Level transformations were explored on three test images. They were passed through custom Grey Level transformations in order to compare them to simple linear scaling of Grey Levels.

The MRI test image's histogram was determined in order to segment features based on Grey Level thresholds, and to determine the area each feature took in the image.

Histogram equalization was also explored using an arbitrary histogram.

Finally, a custom routine for determining an image's histogram and equalizing it is proposed.

**Lab Report received May 26, 2020**

## II. Methods

### A. Grey Level Operations

Test images were loaded into MATLAB, and can be seen in Figure 1, Figure 2, and Figure 3.

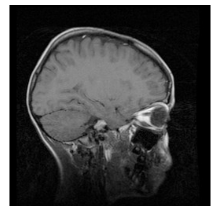

Fig. 1: Test image ”mri.jpg” (256x256)

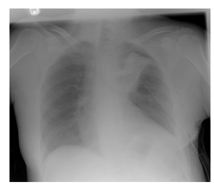

Fig. 2: Test image ”chest1.jpg” (697x806)

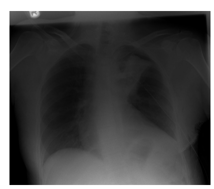

Fig. 3: Test image ”badchest.jpg” (697x806)

Grey Level Transformations seen in Equation 1 (referred to as the $\gamma$-Transform) and Equation 2 (referred to as the $\alpha$-Transform) were applied to each of the above test images, where $I_{max}$ is the maximum intensity value and $I$ is the pixel intensity value in the image. They were applied for $\gamma = 0.5, 2$, and $\alpha = 4, 8, 16$.

$$f_1(I) = I_{max}(\frac{I}{I_{max}})^\gamma \tag{1}$$

$$f_2(I) = \frac{I_{max}}{1 + e^{-\alpha(\frac{I}{I_{max}} - \frac{1}{2})}} \tag{2}$$

The results of the $\gamma$-Transform and the $\alpha$-Transform were compared to just saturating the top and bottom 1% using the MATLAB command `imadjust()`. Results were compared using image subtraction.

For Figure 1, the histogram was determined using the MATLAB command `imhist()`. Thresholds for the background, eyeball tissue, brain tissue, and the fatty tissue were estimated by inspecting the histogram. The area of each tissue was then determined.

### *B. Arbitrary Histogram Equalization*

A histogram was given, seen in Equation 3.

$$H(I) = 12A_0[(\frac{I}{I_{max}})^2 - (\frac{I}{I_{max}})^3] \tag{3}$$

The Equalization function of that histogram, $f_{eq}(I)$, was derived by hand, seen below in Equation 4. A full derivation can be seen in section B.

$$f_{eq}(I) = \frac{4I^3}{I_{max}^2} - \frac{3I^4}{I_{max}^3} \tag{4}$$

### *C. Custom Histogram Calculation Routine*

A method of calculating an image's histogram without using MATLAB commands was coded and implemented with each of the test images. The results were compared with the images with the 1% saturation from `imadjust()`.

To get the histogram, the image was first reshaped to be a *1 x length*$^2$ vector for computational ease. Iterating through all possible Grey Levels, the count for each Grey Level was found and stored in a vector containing the Histogram counts.

The Equalization function, $f_{eq}(I)$, was numerically calculated by applying the summation seen below in Equation 5, for every possible Grey Level.

$$f_{eq}(I) = I_{max} \sum_{k=1}^{k-1} \frac{H(I)}{lw} \tag{5}$$

Where $k$ is the MATLAB index element of the Histogram vector $H(I)$. Because MATLAB counts from 1, the Grey Level is $k-1$. $l$ and $w$ are the length and width of the image in pixels, respectively.

The Probability and Cumulative Density functions were found, as well as each transformed pixel value.

The reshaped image was passed through the equalization function, and finally reshaped to its original dimensions.

## III. RESULTS AND DISCUSSION

For $\gamma = 0.5$, each of the test images became brighter and for $\gamma = 2$, the resulting images became darker. This is seen in Figure 4, Figure 6, and Figure 8. This is to be expected, as $f_1(1)$ is above a line with unity slope on Figure 16 for $\gamma = 0.5$, and below the line for $\gamma = 2$.

Subtracting the $\gamma$-Transformed image from an image saturated by 1% on both sides using `imadjust()`, reveals some similarities and differences in the performance of the $\gamma$ values. When $\gamma = 0.5$, the result of the subtraction leaves only a bit of background noise and edges, indicating that the performance of the two methods is very similar, while for $\gamma = 2$, the resulting image after subtraction appears to be a darker version of the original image. This indicates that the two Grey Level operations are quite different. This is seen in Figure 5, Figure 7, and Figure 9.

For the $\alpha$-Transformations, as $\alpha$ increased, more of the high and low Grey Levels became brighter and darker, respectively. This is seen in Figure 10, Figure 12, and Figure 14, though is easily seen and described best in Figure 10. As $\alpha$ increases, more and more of the middle Grey Levels in the tissue just under the brain become darker and darker. This can be seen graphically in Figure 17, as the Sigmoidal shape of the curve becomes steeper for an increasing $\alpha$. Furthermore, the steep curve starts at higher Grey Levels, and ends at lower Grey Levels when $\alpha$ increases, which corresponds to more Grey Levels becoming brighter and darker.

Subtracting the $\alpha$-Transformed image from an image saturated by 1% on the high and low ends by using `imadjust()` demonstrates that the Grey Level transformation works very similar for low levels of $\alpha$, but not so much for high levels. This can be attributed to the fact that for $\alpha = 4$, the $f_2(I)$ curve is almost linear, which is similar to the linear scaling that is done using `imadjust()`. After subtraction, there is not much seen in the resulting image, just a dim version of the original image, if anything at all. As $\alpha$ increases however, the structures in the image that remain are more distinct, as the Sigmoidal curve increases the contrast for the remaining Grey levels. This is seen in Figure 11, Figure 13, and Figure 15.

The Grey Level threshold to segregate the background from the rest of the image was the Grey Level 2. Likewise, for the Eye 29, for the Brain 89, leaving the remaining Grey Levels up to 255 to be categorized as Fat tissue. Each threshold was determined by inspecting Figure 18. The areas of each feature can be seen in Table I. The estimation of thresholds was not accurate, as the area estimated of each area seems either grossly over or under represented. By inspection, it does not make sense that the area of the background was 14.84%, and the eye area should not be as high as 41.53%. Better segmentation methods should be implemented to determine the area of features.

TABLE I: Features defined in mri.jpg, and areas calculated

| Feature | Proportion of Area |
|---|---|
| Background | 0.1484 |
| Eye | 0.4153 |
| Brain | 0.1402 |
| Fat | 0.2961 |

The equalization function for Equation 3 was derived in section B, and is plotted in Equation 4. After applying Equation 4 to the test images, a lot of Grey Levels became darkened. This makes sense, as the equalization function was derived to increase the contrast of a specific histogram (Equation 3), and as a result, will not be effective for images without that particular histogram. The darkening of the images is seen in Figure 20, Figure 21, and Figure 22.

The MATLAB code for the custom histogram routine was effective. Utilizing the code provided in section C, it was found that the custom histogram routine resulted in an output that was the same histogram as the `imhist()` command. After running all the code in section C, if the code snippet `find(my_mri_hist - mri_hist')` is run in the command window, the answer is an empty vector, indicating that the vector `my_mri_hist` (the histogram found in the custom routine) was identical to the vector `mri_hist` (the histogram of mri.jpg found using the `imhist()` command). Each of the custom histograms can be seen in Figure 23, Figure 25, and Figure 27.

Furthermore, the custom histogram routine increased the contrast of each of the test images, sometimes resulting in an image where the body organs can be seen more clearly due to more contrast (Figure 21, Figure 22), sometimes amplifying noise (Figure 20).

Subtracting the image after equalization from the image saturated 1% on the high and low end revealed the degree of contrast increase in the images. For the chest images, the subtraction leaves only the organ structures, indicating that the equalization does a better job at increasing the contrast than the 1% saturation (Figure 26, Figure 28). The MRI image however, amplified noise contrast, making the overall image tougher to distinguish features (Figure 24).

## IV. Conclusion

Grey Level operations change properties like the contrast of the image while maintaining the geometry of features.

Based on the transformation, pixel intensities can be changed such that the contrast is better or worse by spacing out intensities for Grey Level ranges. However, increasing the contrast over the image can also increase the contrast of the noise, negatively affecting the visual properties of the image. If the transformation was approximately close to linear, the closer the transformation would act like the MATLAB function `imadjust()`. This was seen experimentally by image subtraction.

A histogram was found for mri.jpg, and was segmented into four classes by inspection to assign tissue types to areas. This was found to not be very effective, as it was not a precise method.

Histogram equalization for a given histogram was performed by hand. The equalization transformation was performed on the test images. The equalization transformation was found to not increase contrast, as the equalization function was derived for one particular image.

The custom histogram finding routine was proven to be successful, as the results were equivalent to the MATLAB output for `imhist()`. While equalization technically increases the contrast, the resulting images are not the best visually, as the contrast of the noise was also increased. The equalization of the images was quite different than the 1% contrast reduction, as the subtracted images displayed an image resembling the original image.

## Appendix A
## Image Results

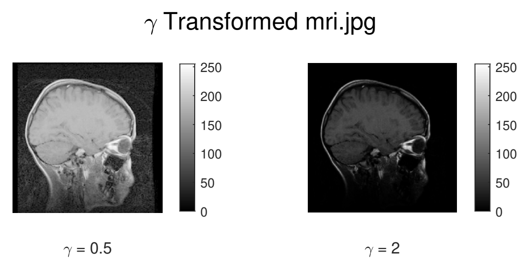


Fig. 4: $\gamma$-Transformed mri.jpg for $\gamma = 0.5, 2$

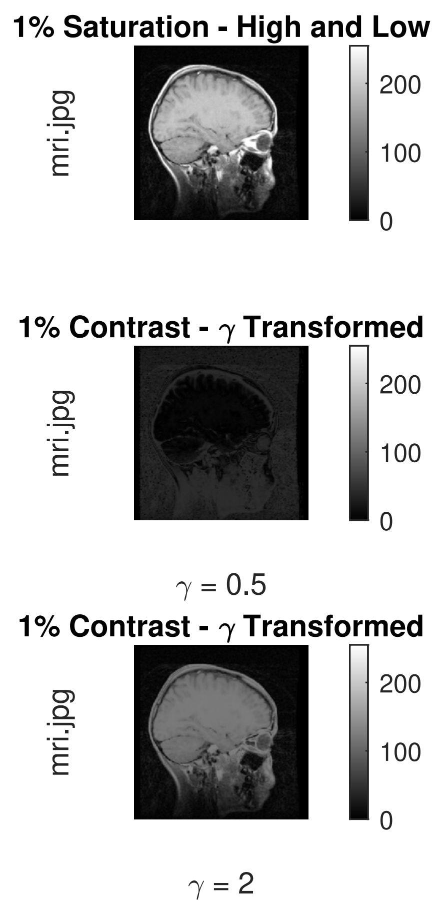


Fig. 5: 1% Saturation - High and Low, and $\gamma$-Transformed mri.jpg subtracted from the 1% Saturation Image

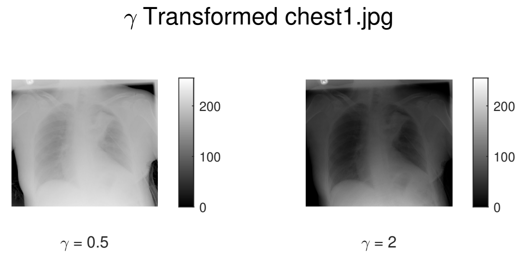


Fig. 6: $\gamma$-Transformed chest1.jpg for $\gamma = 0.5, 2$

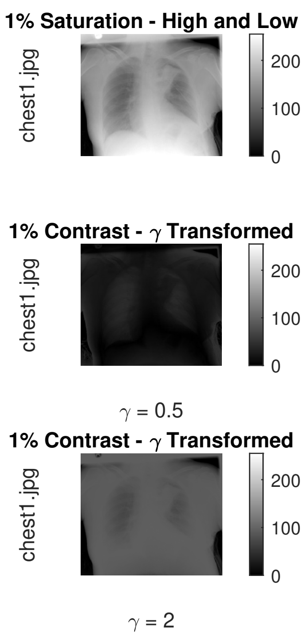


Fig. 7: 1% Saturation - High and Low, and $\gamma$-Transformed mri.jpg subtracted from the 1% Saturation Image

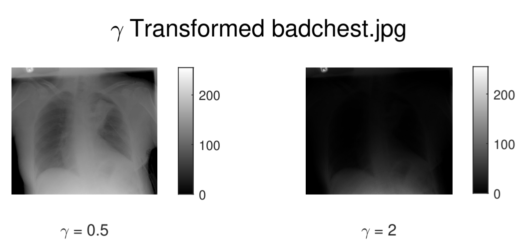


Fig. 8: $\gamma$-Transformed badchest.jpg for $\gamma = 0.5, 2$

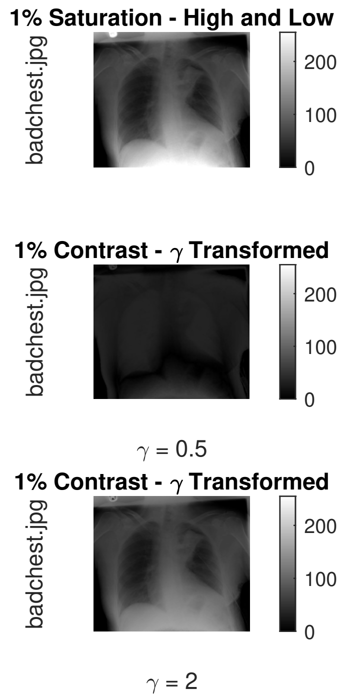


Fig. 9: 1% Saturation - High and Low, and $\gamma$-Transformed mri.jpg subtracted from the 1% Saturation Image

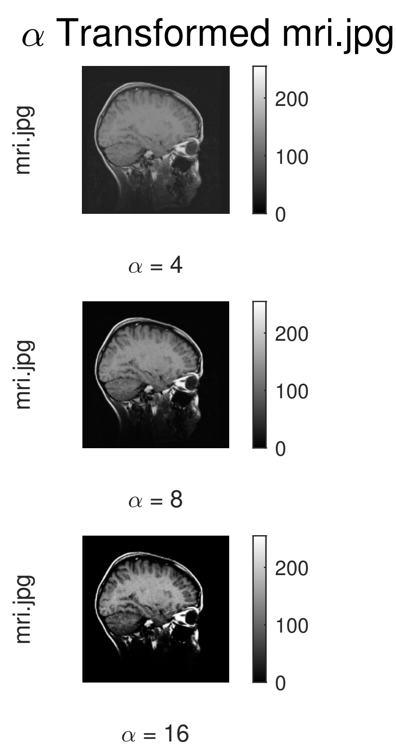


Fig. 10: $\alpha$-Transformed mri.jpg for $\alpha = 4, 8, 16$

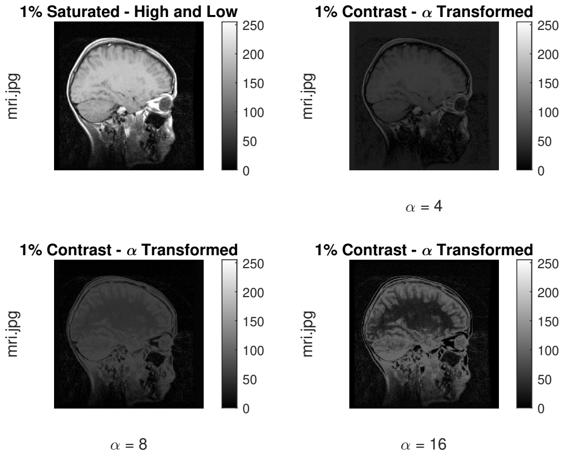


Fig. 11: 1% Saturation - High and Low, and $\alpha$-Transformed mri.jpg subtracted from the 1% Saturation Image

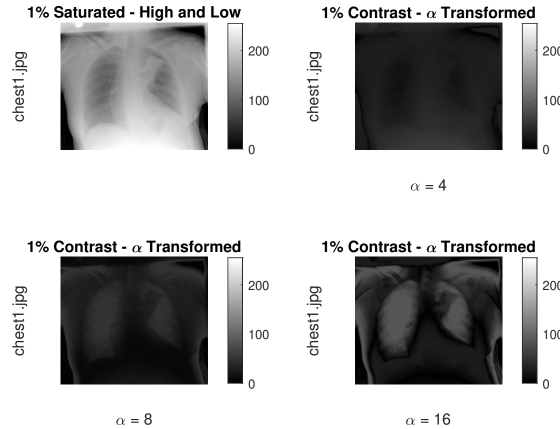


Fig. 13: 1% Saturation - High and Low, and $\alpha$-Transformed chest1.jpg subtracted from the 1% Saturation Image

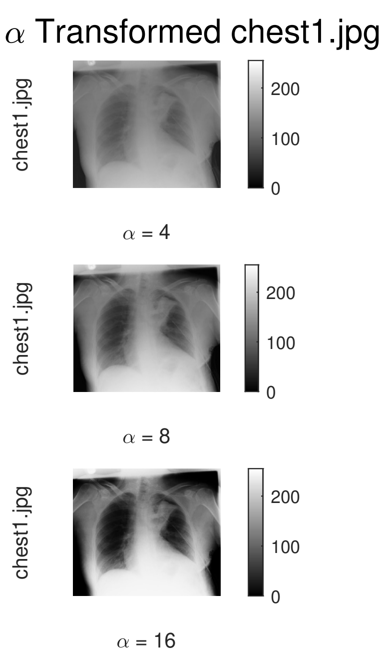


Fig. 12: $\alpha$-Transformed chest1.jpg for $\alpha = 4, 8, 16$

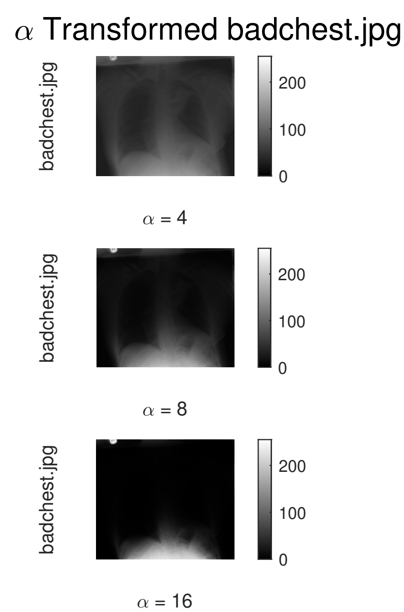


Fig. 14: $\alpha$-Transformed badchest.jpg for $\alpha = 4, 8, 16$

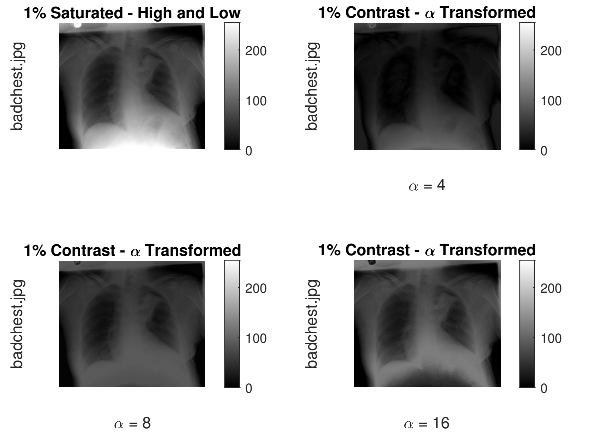


Fig. 15: 1% Saturation - High and Low, and $\alpha$-Transformed badchest.jpg subtracted from the 1% Saturation Image

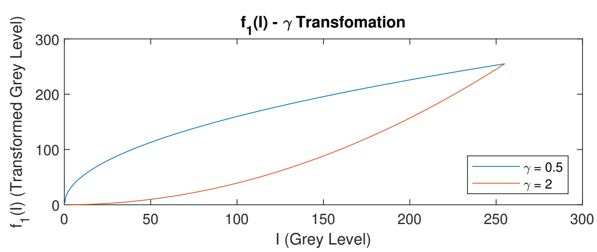


Fig. 16: Graph of Equation 1

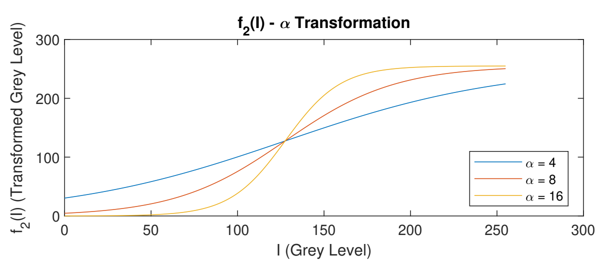


Fig. 17: Graph of Equation 2

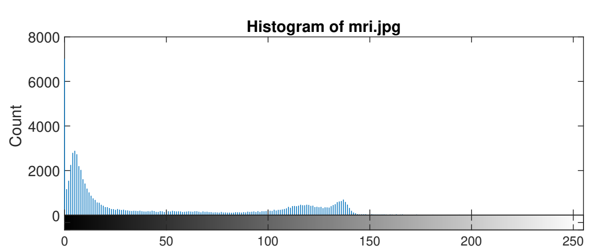


Fig. 18: Histogram of mri.jpg

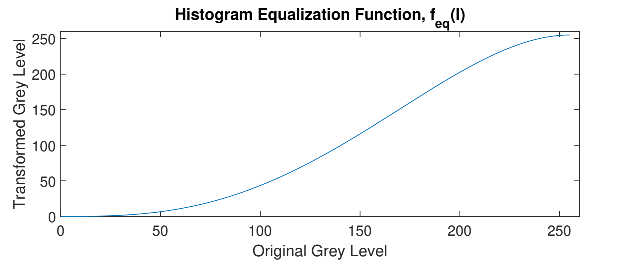


Fig. 19: Histogram Equalization function, $f_{eq}(I)$, for the given Histogram in Equation 3

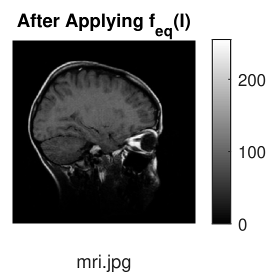


Fig. 20: mri.jpg after being passed through Equation 4

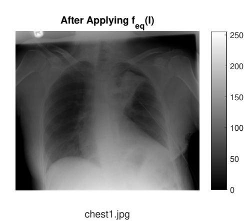


Fig. 21: chest1.jpg after being passed through Equation 4

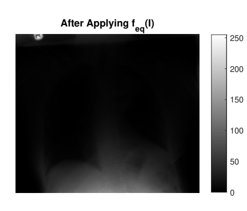


Fig. 22: badchest.jpg after being passed through Equation 4

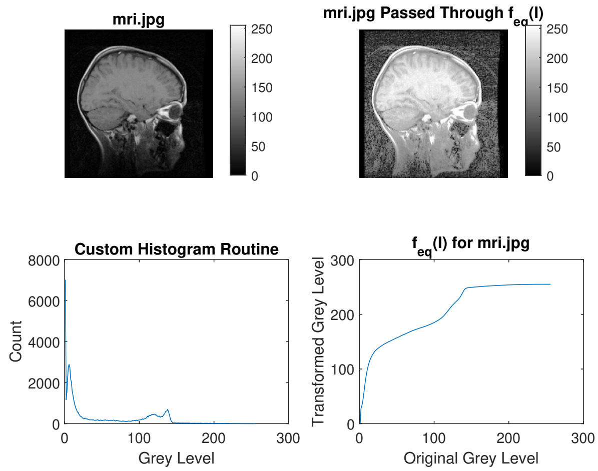


Fig. 23: mri.jpg (top left), mri.jpg passed through equalization function (top right), histogram found using custom routine (bottom left), and histogram equalization function found using custom routine (bottom right)

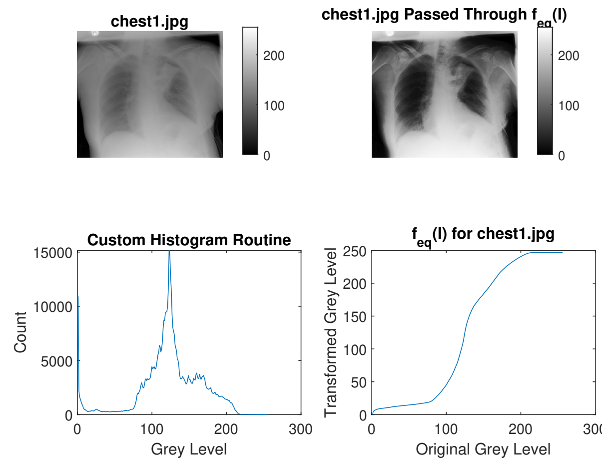


Fig. 25: chest1.jpg (top left), chest1.jpg passed through equalization function (top right), histogram found using custom routine (bottom left), and histogram equalization function found using custom routine (bottom right)

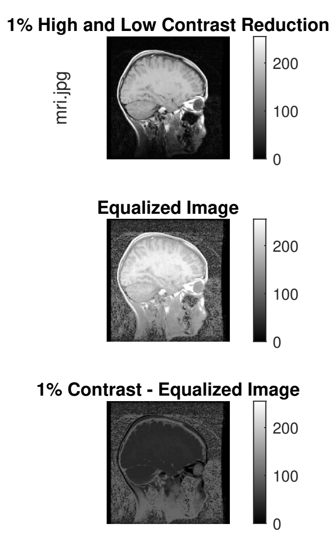


Fig. 24: Comparison of mri.jpg after being equalized through the custom histogram equalization routine

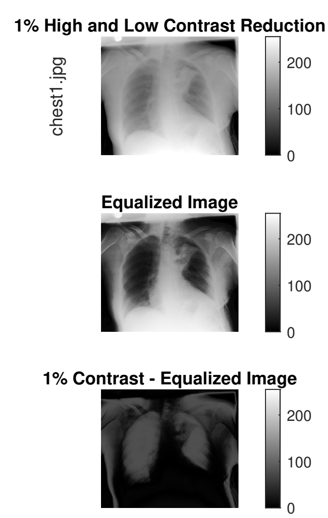


Fig. 26: Comparison of chest1.jpg after being equalized through the custom histogram equalization routine

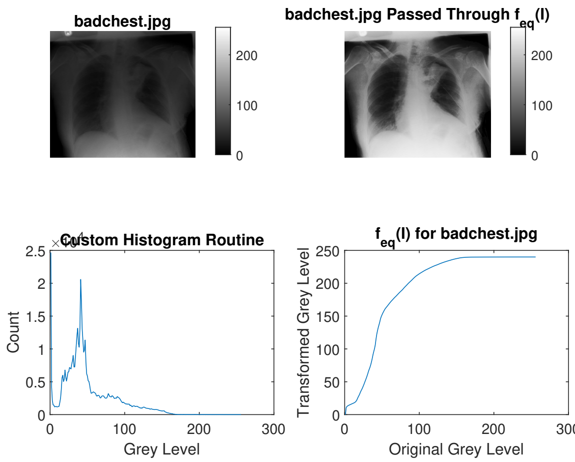


Fig. 27: badchest.jpg (top left), chest1.jpg passed through equalization function (top right), histogram found using custom routine (bottom left), and histogram equalization function found using custom routine (bottom right)

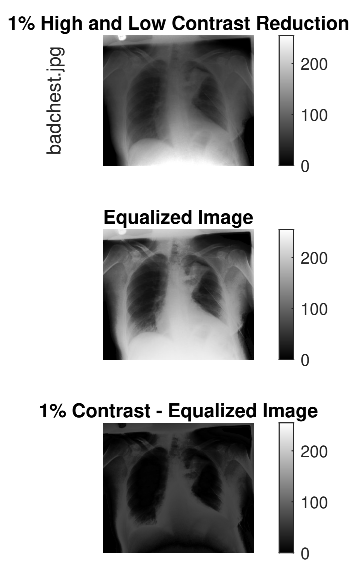


Fig. 28: Comparison of badchest.jpg after being equalized through the custom histogram equalization routine

## Appendix B
## Equalization Function Derivation

$$f_{eq}(I) = \frac{1}{A_0}\int_0^I H(\lambda)d\lambda$$

$$f_{eq}(I) = \frac{1}{A_0}\int_0^I 12A_0[(\frac{\lambda}{I_{max}})^2 - (\frac{\lambda}{I_{max}})^3]d\lambda$$

$$f_{eq}(I) = 12[\frac{\lambda^3}{3I_{max}} - \frac{\lambda^4}{4I_{max}}\Big|_0^I$$

$$f_{eq}(I) = \frac{4I^3}{I_{max}^2} - \frac{3I^4}{I_{max}^3}$$

# APPENDIX C
# MATLAB CODE

Listing 1: Main Code used in lab

```
%% ENGG 4660: MEDICAL IMAGE PROCESSING
% LAB 3: GREY LEVEL OPERATIONS
% DANIEL SHERMAN
% 0954083
% FEBRUARY 28, 2020

%% CLEAN UP

close all
clear all
clc

%%  LOAD IN IMAGES

mri = imread('mri.jpg');
chest = imread('chest1.jpg');
badchest = imread('badchest.jpg');

%% IMPLEMENT GAMMA TRANSFORM

[mri_g05, mri_g2] = gamma_transform(mri, 'mri.jpg');
[chest_g05, chest_g2] = gamma_transform(chest, 'chest1.jpg');
[bchest_g05, bchest_g2] = gamma_transform(badchest, 'badchest.jpg');

%% IMPLEMENT ALPHA TRANSFORM

[mri_a4, mri_a8, mri_a16] = alpha_transform(mri, 'mri.jpg');
[chest_a4, chest_a8, chest_a16] = alpha_transform(chest, 'chest1.jpg');
[bchest_a4, bchest_a8, bchest_a16] = alpha_transform(badchest, 'badchest.jpg');

%% DISPLAY GREY LEVEL TRANSFORMS

%gamma transform
bits = [0:255];

figure()
plot(bits, 255*(bits./255).^0.5);
hold on
plot(bits, 255*(bits./255).^2);
title('f_1(I) - \gamma Transfomation')
xlabel('I (Grey Level)')
ylabel('f_1(I) (Transformed Grey Level)')
legend('\gamma = 0.5', '\gamma = 2')

%alpha transform
figure()
plot(bits, 255./(1 + exp(-4*((bits./255) - 0.5))))
hold on
plot(bits, 255./(1 + exp(-8*((bits./255) - 0.5))))
hold on
plot(bits, 255./(1 + exp(-16*((bits./255) - 0.5))))
title('f_2(I) - \alpha Transformation')
xlabel('I (Grey Level)')
ylabel('f_2(I) (Transformed Grey Level)')
legend('\alpha = 4', '\alpha = 8', '\alpha = 16')

%% MRI HISTOGRAM

mri_hist = imhist(mri);

figure()
imhist(mri)
title('Histogram of mri.jpg')
ylabel('Count')
```

```
ylim([0 8000])

background_area = sum(mri_hist(1:3))/(256*256);
eye_area = sum(mri_hist(4:30))/(256*256);
brain_area = sum(mri_hist(31:90))/(256*256);
fat_area = sum(mri_hist(91:256))/(256*256);

%% EQUALIZE HISTOGRAM

histo = 12.*((bits/255).^2 - (bits/255).^3);

f_eq = 255.*(4.*(bits./255).^3 - 3.*(bits./255).^4);

figure()
plot(bits, f_eq)
title('Histogram Equalization Function, f_e_q(I)')
xlabel('Original Grey Level')
ylabel('Transformed Grey Level')
xlim([0 260])
ylim([0 260])

%% APPLY EQUALIZATION TO THE TEST IMAGES

eq_mri = histo_eq(mri, 'mri.jpg');
eq_chest = histo_eq(chest, 'chest1.jpg');
eq_bchest = histo_eq(badchest, 'badchest.jpg');

%% MY OWN ROUTINE

my_mri_hist = my_own_histogram(mri, 8, 'mri.jpg');
my_own_histogram(chest, 8, 'chest1.jpg')
my_own_histogram(badchest,8, 'badchest.jpg')
```

Listing 2: Code used to apply the $\gamma$-Transformation

```
function [gamma_img_1, gamma_img_2] = gamma_transform(image, name)
%% DOCUMENTATION

% FUNCTION TAKES AN IMAGE AND A FILE NAME.
% PERFORMS GREY LEVEL TRANSFORMATION f1(image) = I_max(image/I_max).^gamma
% FOR VARYING GAMMA VALUES AND RETURNS THE TRANSFORMED IMAGE

% MADE BY: DANIEL SHERMAN
% FEBRUARY 20, 2020

%% GAMMA TRANSFORM

gam = [0.5, 2];

I_max = double(max(image, [], 'all')); %get maximum grey level in the image

gamma_img_1 = I_max.*(double(image)./I_max).^gam(1);
gamma_img_2 = I_max.*(double(image)./I_max).^gam(2);

%% IMAGE ADJUST AND COMPARE

img_a = imadjust(image);

img_delta_1 = image_subtract(img_a, gamma_img_1, name, '\gamma = 0.5');
img_delta_2 = image_subtract(img_a, gamma_img_2, name, '\gamma = 2');

figure()
subplot(1,2,1)
imshow(uint8(gamma_img_1))
colorbar
xlabel('\gamma = 0.5')

subplot(1,2,2)
imshow(uint8(gamma_img_2))
```

```
xlabel('\gamma = 2')
sgtitle(strcat(['\gamma Transformed ' name]))
colorbar

figure()
subplot(3,1,1)
imshow(uint8(img_a))
title('1% Saturation - High and Low')
colorbar
ylabel(name)

subplot(3,1,2)
imshow(uint8(img_delta_1))
ylabel(name)
xlabel('\gamma = 0.5')
colorbar
title('1% Contrast - \gamma Transformed')

subplot(3,1,3)
imshow(uint8(img_delta_2))
ylabel(name)
colorbar
title('1% Contrast - \gamma Transformed')
xlabel('\gamma = 2')
```

Listing 3: Code used to apply the $\alpha$-Transformation

```
function [alpha_img_1, alpha_img_2, alpha_img_3] = alpha_transform(image, name)
%% DOCUMENTATION

% FUNCTION TAKES AN IMAGE AND A FILE NAME.
% PERFORMS GREY LEVEL TRANSFORMATION f2(image) = I_max/(1 + (image/(I_max - 1/2)).*e^(-alph))
% FOR VARYING ALPHA VALUES AND RETURNS THE TRANSFORMED IMAGE

% MADE BY: DANIEL SHERMAN
% MARCH 2, 2020

%% ALPHA TRANSFORM

alph = [4, 8, 16];

I_max = double(max(image, [], 'all')); %get maximum grey level in the image

alpha_img_1 = I_max./(1 + exp(-alph(1)*(-0.5 + double(image)./I_max)));
alpha_img_2 = I_max./(1 + exp(-alph(2)*(-0.5 + double(image)./I_max)));
alpha_img_3 = I_max./(1 + exp(-alph(3)*(-0.5 + double(image)./I_max)));

%% IMAGE ADJUST AND COMPARE

img_adj = imadjust(image);

img_delta_1 = image_subtract(img_adj, alpha_img_1, 'mri.jpg', '\alpha = 4');
img_delta_2 = image_subtract(img_adj, alpha_img_2, 'mri.jpg', '\alpha = 8');
img_delta_3 = image_subtract(img_adj, alpha_img_3, 'mri.jpg', '\alpha = 16');

figure()
subplot(3,1,1)
imshow(uint8(alpha_img_1))
colorbar
xlabel('\alpha = 4')
ylabel(name)

subplot(3,1,2)
imshow(uint8(alpha_img_2))
colorbar
xlabel('\alpha = 8')
ylabel(name)

subplot(3,1,3)
```

```
imshow(uint8(alpha_img_3))
colorbar
xlabel('\alpha = 16')
ylabel(name)
sgtitle(strcat(['\alpha Transformed ' name]))

figure()
subplot(2,2,1)
imshow(uint8(img_adj))
colorbar
ylabel(name)
title('1% Saturated - High and Low')

subplot(2,2,2)
imshow(uint8(img_delta_1))
colorbar
ylabel(name)
title('1% Contrast - \alpha Transformed')
xlabel('\alpha = 4')

subplot(2,2,3)
imshow(uint8(img_delta_2))
colorbar
ylabel(name)
title('1% Contrast - \alpha Transformed')
xlabel('\alpha = 8')

subplot(2,2,4)
imshow(uint8(img_delta_3))
colorbar
ylabel(name)
title('1% Contrast - \alpha Transformed')
xlabel('\alpha = 16')
```

Listing 4: Code used to perform image subtraction

```
function delta = image_subtract(img1, img2, name, trial)

delta = abs(double(img1) - double(img2));

% figure()
% imshow(uint8(delta))
% colorbar
% title(name)
% ylabel('Adjusted Image - Custom Transform')
% xlabel(trial)
```

Listing 5: Code used to equalize the given histogram

```
function new_img = histo_eq(img, name)
%% DOCUMENTATION

% FUNCTION RECIEVES AN IMAGE AND APPLIES THE HISTOGRAM EQUALIZATION FUNCTION
% DERIVED FROM LAB 3 QUESTION 3
% FUNCTION DISPLAYS THE IMAGE AFTER EQUALIZATION

% MADE BY: DANIEL SHERMAN
% MARCH 4, 2020

%% START OF CODE

new_img = 255.*(4.*(double(img)./255).^3 - 3.*(double(img)./255).^4);

figure()
imshow(uint8(new_img))
colorbar
title('After Applying f_e_q(I)')
xlabel(name)

```

```
uint8(new_img);
```

Listing 6: Code used for the custom histogram calculation routine

```
function hist = my_own_histogram(img, bit, name)
%% DOCUMENTATION

% THIS FUNCTION ACCEPTS AN IMAGE AND RETURNS A HISTOGRAM WITHOUT USING MATLAB FUNCTIONS
% FUNCTION ALSO DISPLAYS THE IMAGE AFTER EQUALIZATION, AND PERFORMS IMAGE
% SUBTRACTION TO COMPARE RESULTS

% MADE BY: DANIEL SHERMAN
% MARCH 2, 2020

%% START OF CODE

%% FIND HISTOGRAM

grey_level = [1:2.^bit]; %store grey levels, corresponding from 0 to 255 for 8 bit
[len wid] = size(img); %take size of image
area = len*wid; %calculate area
I_max = double(max(img, [], 'all')); %get maximum grey level in the image

for k = grey_level %iterate through all possible grey levels
    %grab all indicies of pixels that correspond with the current grey level
    hist(k) = length(find(img == k - 1)); %count all the pixels that correspond,
    %to find the histogram value
end

pdf = hist./area; %calculate pdf

for k = grey_level %iterate through all grey levels
    cdf(k) = sum(pdf(1:k)); %calculate the cdf
end

tx_pixels = round(I_max.*cdf); %find intensity at pixel values corresponding to the cdf

%% CREATE HISTOGRAM EQUALIZATION FUNCTION

for k = grey_level %iterate through all grey levels
    histo_eq(k) = I_max*sum(hist(1:k - 1))/area; %apply the histogram equalization integral,
    %in discrete space
end

%%  PASS IMAGE THROUGH THE EQUALIZATION FUNCTION

reshaped_img = reshape(img, 1, area); %reshape to 1 by area vector for computational ease
img_eq = zeros(size(reshaped_img)); %initialize equalized image to zeros of the same size

for k = grey_level %iterate through all grey levels
    found_gl = find(reshaped_img == k - 1); %collect all indicies of image that
    %correspond to current grey level
    img_eq(found_gl) = histo_eq(k); %set grey level in new image to be the
    %value of the equalized histogram
end

img_eq = reshape(img_eq, len, wid); %reshape again to be the size of an image

%% PLOT NICELY

figure()
subplot(2,2,1)
imshow(img) %plot original image
colorbar
title(name)

subplot(2,2,2)
imshow(uint8(img_eq)) %plot equalized image
colorbar
```

```
title(strcat([name ' Passed Through f_e_q(I)']))

subplot(2,2,3)
plot(grey_level, hist) %plot the image's histogram
xlabel('Grey Level')
ylabel('Count')
title('Custom Histogram Routine')

subplot(2,2,4)
plot(grey_level, histo_eq) %plot the equalization function
xlabel('Original Grey Level')
ylabel('Transformed Grey Level')
title(strcat(['f_e_q(I) for ' name]))

%% COMPARISON TO IMADJUST()

img_adj = imadjust(img); %utilize imadjust()

img_delta_1 = image_subtract(img_adj, img_eq, name, '\gamma = 0.5'); %apply image
%subtraction for comparison

figure()
subplot(3,1,1)
imshow(uint8(img_adj))
colorbar
ylabel(name)
title('1% High and Low Contrast Reduction')

subplot(3,1,2)
imshow(uint8(img_eq))
colorbar
title('Equalized Image')

subplot(3,1,3)
imshow(uint8(img_delta_1)) %plot differences
colorbar
title('1% Contrast - Equalized Image')
```