# Histogram Equalization

Jose Luciano

April 11, 2022

## Abstract

My goal in this lab is to perform histogram equalization on a greyscale image. To accomplish this I calculated the normalized histogram of the given grayscale image. The normalized histogram is used to calculate the transformation function which will be used to perform histogram equalization on the image. Lastly, the standard deviation and mean of the equalized and original image are computed for analysis.

**Technical Discussion**

The function equalize is able to take a greyscale image as an input and return a histogram equalized version of the image. The function calls a total of 3 other functions which are compute_histogram, plot_histogram, and histogram_transform. Furthermore, the histogram that is returned by the function histogram_transform is used to assign values to the output image which will result in the histogram equalized version of our original greyscale image.

The function compute_histogram takes in a greyscale image and calculates the normalized histogram of the image (PMF). The calculation is performed by the following equation: h = h/(r*c) where h represents the frequency of the pixel intensity values (0-255) and each value is divided by the number of pixels in the image.

The function histogram_transform takes in a histogram and calculates the transformation function required to perform histogram equalization on the greyscale image. The following formula is used to calculate the transformation function: $T(i) = (L-1)\sum h(i)$ where L represents the maximum number of pixel intensity values and h(i) represents the values of the normalized histogram.

The function plot_histogram takes a histogram and plots a bar graph where the x-axis represents the pixel intensity values and the y-axis represents the probability mass function (PMF).

**Results**

The first image that was used to test our equalize function is Figure 1a. The result is very noticeable when we compare Figure 1a to Figure 1b because there aren't as many lower intensity values in Figure 1b as there were in Figure 1a. Similarly, when the image in Figure 2a was tested the resulting image has better contrast which can be seen in Figure 2b. I believe that both Figure 2b and Figure 1b are better than the original image because in the original image there isn't as much contrast.

Figure 1a. Dark Image

Figure 1b. Histogram Equalized Version of Dark Image

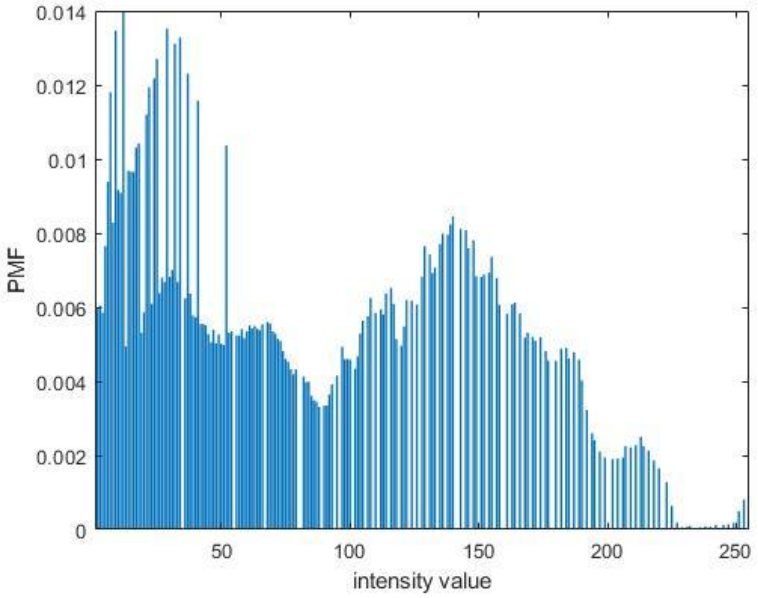


Figure 1c. Histogram of Dark Image

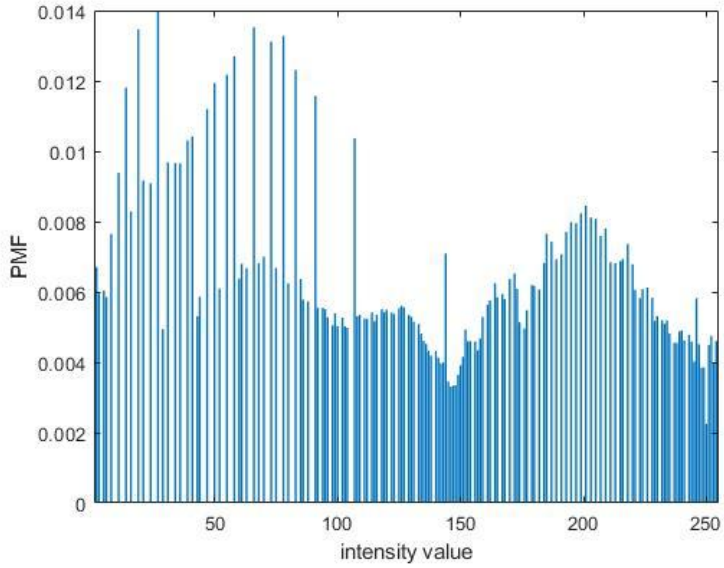


Figure 1d. Histogram of Equalized Dark Image

Dark Image: Standard Deviation = 60.353473  Mean = 82.816909
Equalized Image: Standard Deviation = 73.397536  Mean = 128.400086

Figure 2a. Light Image

Figure 2b. Histogram Equalized Version of Light Image

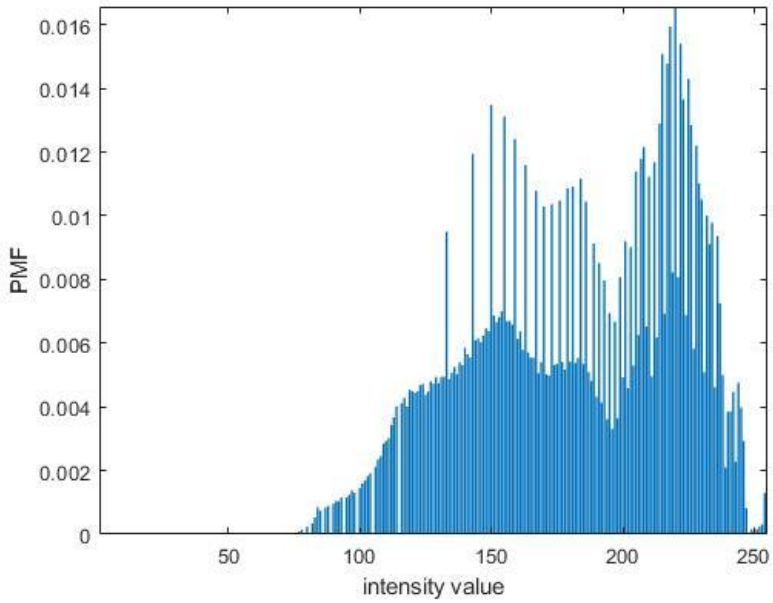


Figure 2c. Histogram of Light Image

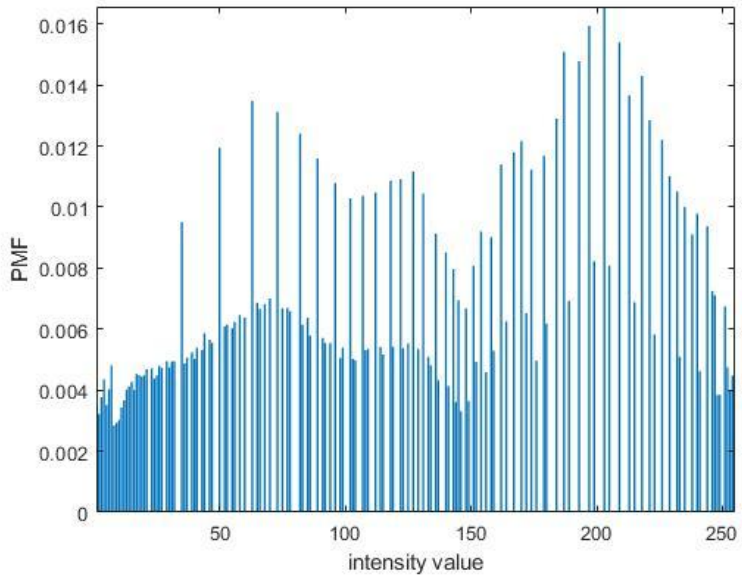


Figure 2d. Histogram of Equalized Light Image

Light Image: Standard Deviation = 39.150753  Mean = 182.023691
Equalized Image: Standard Deviation = 73.816858  Mean = 128.575921

# equalize.m

```
function out = equalize(input)
% equalize The function performs histogram equalization on a greyscale
%          image.
%
% Syntax:
%   out = equalize(input);
%
% Input:
%   input = greyscale image with intensity values from 0-255
%
% Output:
%   out = returns a greyscale image which is the histogram equalized
%         version of the original image
%
% History:
%   Jose Luciano - Created equalize function 4/10/2022
[r c] = size(input);
out = uint8(zeros(r,c));

% perform the histogram equalization of input image [0,255]
graph = compute_histogram(input);

plot_histogram(graph);

%compute the transformation vector
T = histogram_transform(graph);

% loop all the values, perform equalization for each pixel
for x = 1:r %each row
    for y = 1:c %each column
         index = input(x,y);
         if index == 0
             index = 1;
         end
         out(x,y) = T(index,1);%from T
    end
end
graph_eq = compute_histogram(out);
plot_histogram(graph_eq);

% get average and standard deviation using built-in functions
% change images to type double to calculate mean and std
input = double(input);
std_original = std(input(:));
mean_original = mean(input(:));

out = double(out);
std_eq = std(out(:));
mean_eq = mean(out(:));
```

```
fprintf('Original Image: Standard Deviation = %f   ', std_original);
fprintf('Mean = %f\n', mean_original);

fprintf('Equalized Image: Standard Deviation = %f   ', std_eq);
fprintf('Mean = %f\n', mean_eq);

out = uint8(out);
end
```

*Published with MATLAB® R2020b*

# compute_histogram.m

```
function h = compute_histogram(A)
% compute_histogram Compute the normalized histogram of the image
%
% Syntax:
%   h = compute_histogram(A);
%
% Input:
%   A =  greyscale image with intensity values from 0-255
%
% Output:
%   h = a vector of length 256 which is the normalized histogram of
 the
%       greyscale image
% History:
%   Jose Luciano - Created compute_histogram function 4/9/2022
[r c] = size(A);
%initialize histogram
h = double(zeros(256,1));
for x = 1:r
    for y = 1:c
        index = A(x,y);
        if index == 0
            index = 1;
        end
        %count how many times each value appears
        h(index,1) = h(index,1) + 1;

    end
end
%normalization
for x = 1:256
    h(x,1) = h(x,1)./(r*c);
end
end
```

*Published with MATLAB® R2020b*

# histogram_transform.m

```
function T = histogram_transform(A)
% histogram_transform Computes the transformation function for
 histogram
%                     equalization
%
% Syntax:
%   T = histogram_transform(A);
%
% Input:
%   A = a normalized histogram
%
% Output:
%   T = a length 256 vector which represents the transformation
 function
%       for histogram equalization
%
% History:
%   Jose Luciano - Created histogram_tranform function 4/9/2022
T = uint8(zeros(256,1));
sum = 0;
for x = 1:256
    sum = sum + A(x,1);
    T(x,1) = 255*sum; %use the equation from textbook
end
end
```

*Published with MATLAB® R2020b*

# plot_histogram.m

```
function plot_histogram(data)
% plot_histogram Plots the normalized histogram
%
% Syntax:
%   plot_histogram(data);
%
% Input:
%   data = normalized histogram
%
% Output:
%   plots the histogram
%
% History:
%   Jose Luciano - Created plot_histogram function 4/09/2022
figure;
bar(data);

%change axis, xlabel = intensity value and ylabel = PMF
xlabel('intensity value');
xlim([1 255]);
ylabel('PMF');
y = max(data(:,1));
ylim([0 y]);

end
```

*Published with MATLAB® R2020b*

```
%test script
in_img = imread('Lab_03_image2_light.tif');
temp = equalize(in_img);
figure;
imshow(temp);
imwrite(temp, 'light_image.png');
imwrite(in_img,'Lab_03_image2_light(png).png');

in_img2 = imread('Lab_03_image1_dark.tif');
temp2 = equalize(in_img2);
figure;
imshow(temp2);
imwrite(temp2, 'dark_image.png');
imwrite(in_img2,'Lab_03_image1_dark(png).png');
```

*Published with MATLAB® R2020b*