University ID : N17385760
Net ID : iv447

B) Practical programming assignments:

B1) Compute a Histogram and CDF

Write code that reads a 2D image as input and returns a 1D array of the relative frequencies of occurrence of greylevels in your image. Provide a choice for quantizing a binning of the greylevels into n quantized bins between 0 and the maximum value (please remember that for an 8bit image, this is the range 0 … 255 for the range 0 … L-1).

• Calculate the histogram of an image of your choice, please note that a color image first needs to be converted into black-and-white

Histogram of Image :

Histogram : Histogram for an image calculates the image intensity. When we are creating a histogram we want we want to make a chart that tells us about some very good information about how data is spread out. This can be done using binning.

I have implemented histogram creation code in Matlab. For this i am performing following steps :

1. Reading image from matlab console using imread. (Image can be in jpg or png format).
2. Converting the image to grayimage using rgb2gray function in Matlab.
3. I have created histogram using 2 bins.

- 100
- 255

4. I am calling histogram function :

```
histogram ( grayimage, numberofbins , minvalue , maxvalue)
```

Histogram function Implementation :

```
function [ Image_histogram] = histogram ( image, nbins, min, max)

% Create a array of size n.
binsize = (max - min)/nbins;
Image_histogram = zeros(256,1);
for i=1:1:size(image,1)
```

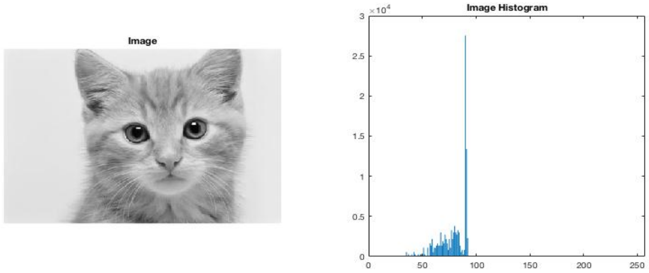
```
     for j=1:1:size(image,2)
        val=image(i,j);
        %read input image level
        if val >= min && val <= max
            for i=1:1:nbins
                if val <= i*binsize+min && val > (i-1)*binsize+min
                % original histogram in pixels
                   Image_histogram(i)=Image_histogram(i)+1;
                end
            end
        end
    end
end

subplot(2,3,1) , imshow(image);
title("Image");
subplot(2,3,2),bar(Image_histogram);
title("Image Histogram");
```

Matlab Console :

```
I = imread(‘test.jpg’);
A = rgb2gray(I);
histogram(A,100,0,255);
```

Output :

1. Implemented for a colored image of cat with bin size 100 :



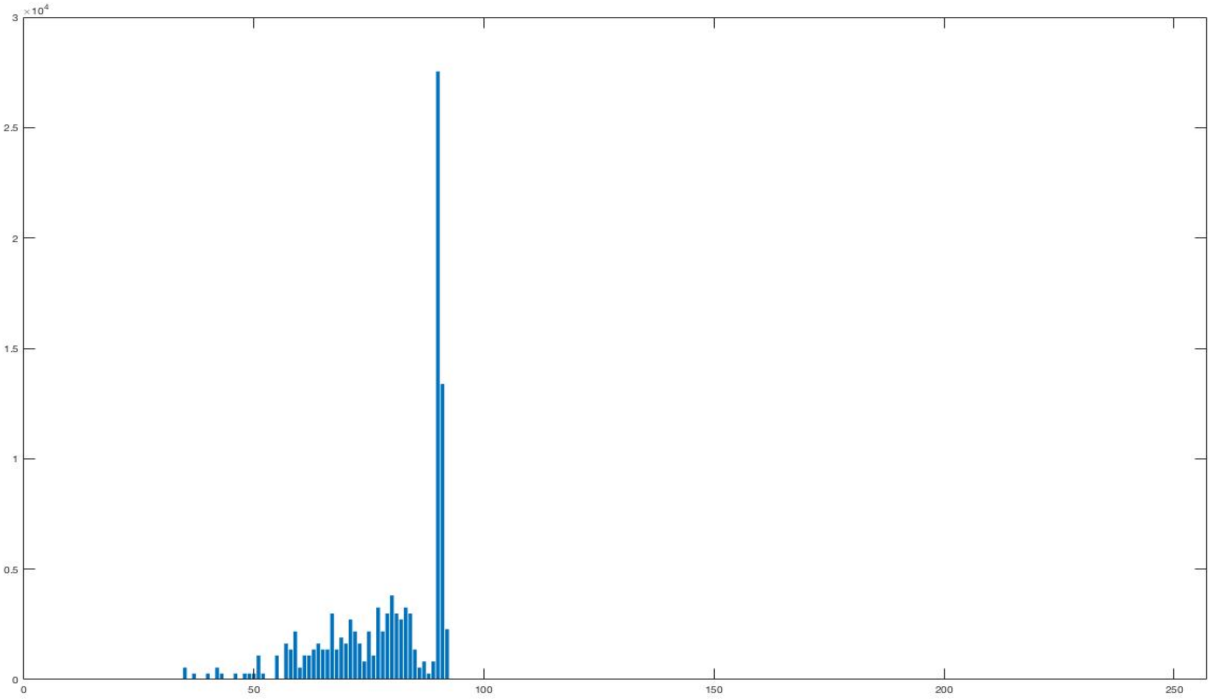
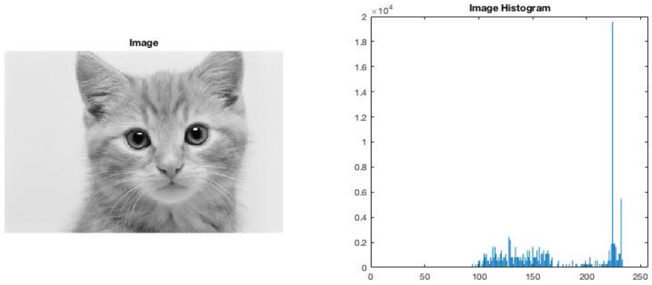


Image Histogram

2. Implemented with colored cat Image with bin size 255 :



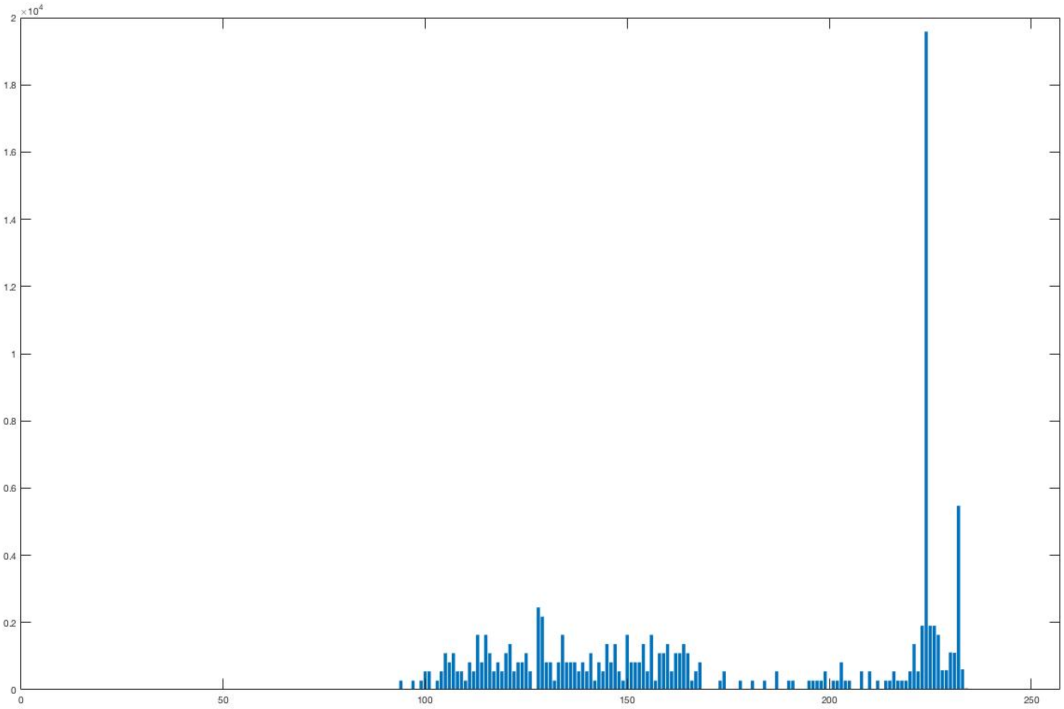
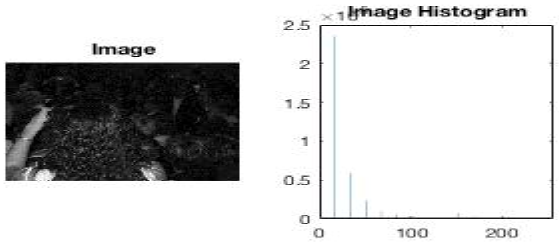


Image Histogram

3 . Implemented with black and white image with bin size 255 :

```
 J = imread(‘crowd.png’);
histogram(J , 255 , 0 ,255);
```



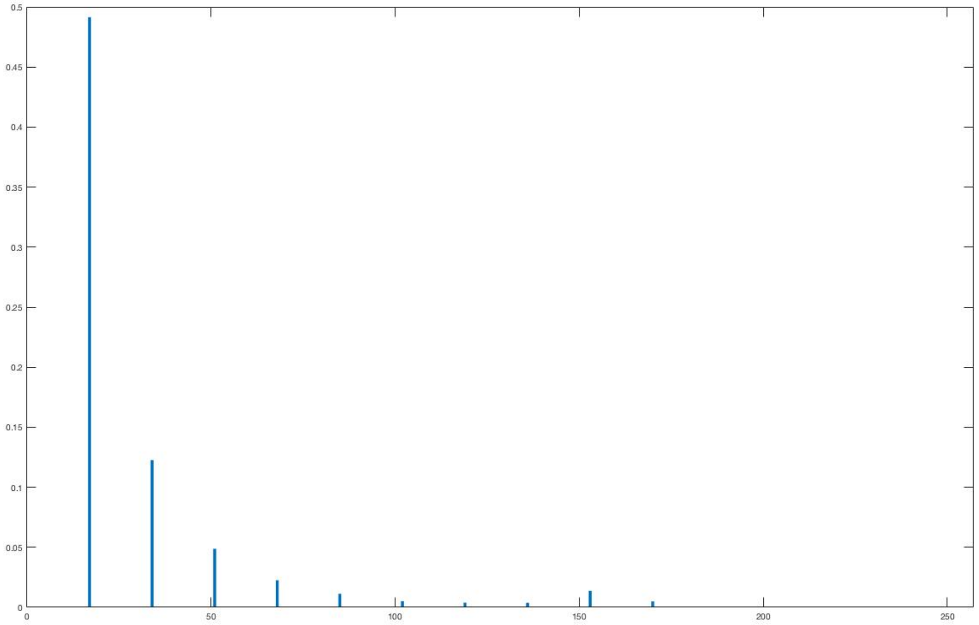


Image Histogram

• Normalize the histogram by the image size to present a probability density function (pdf), plot the pdf.

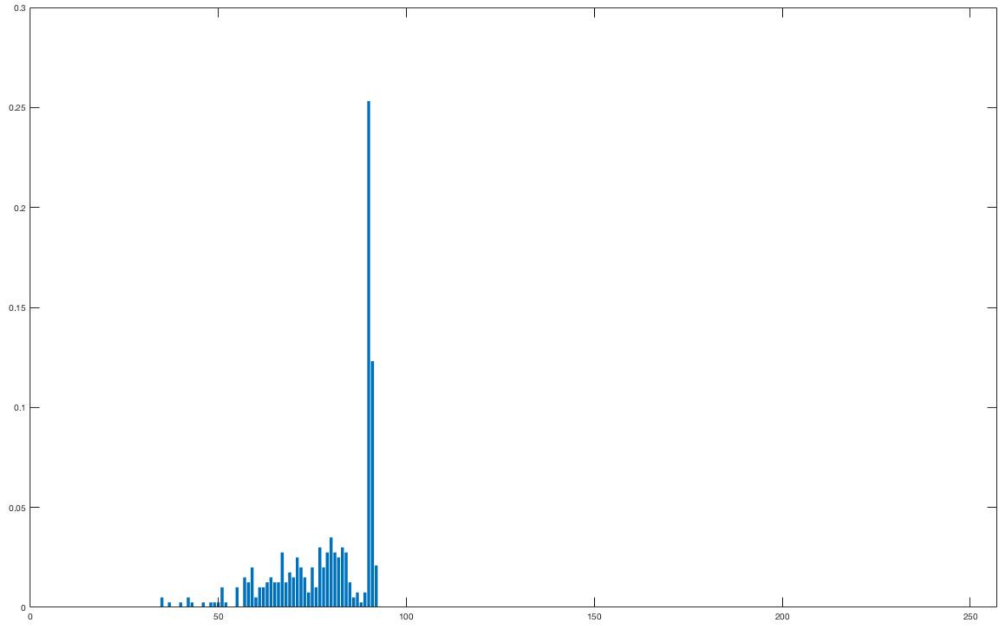
Normalization of Histogram :

Normalization is a process that changes the range of pixel intensity values.

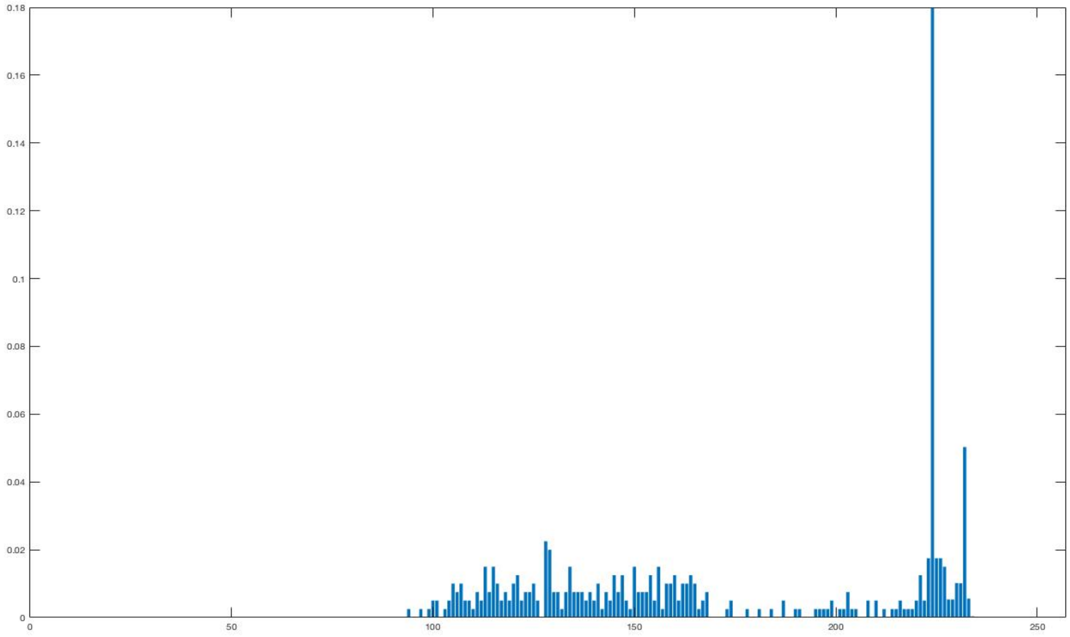
Normalization implementation in Matlab :

```
function [ Norm_InputIm_histogram] = normhistogram ( InputImage,Image_histogram)

Norm_InputIm_histogram = zeros(256,1);
r = size(InputImage,1);
c = size(InputImage,2);
resolution = r * c;
for i=1:size(Image_histogram)
   % normalized histogram pdf
   Norm_InputIm_histogram(i)=Image_histogram(i)/resolution;
end

figure, bar(Norm_InputIm_histogram);
```

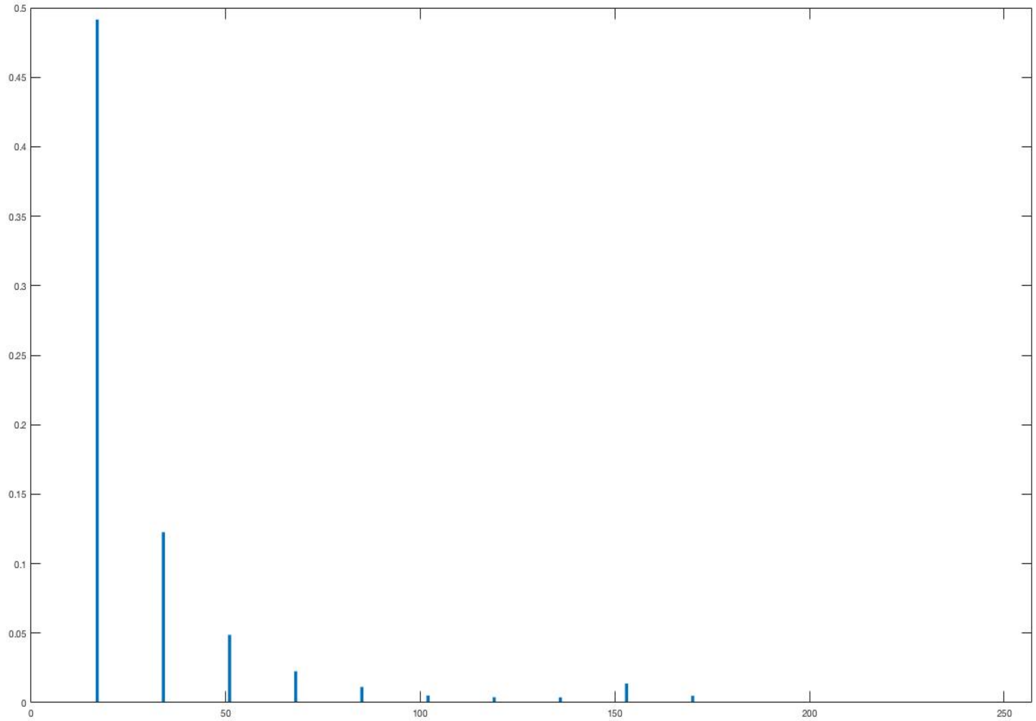
1. Normalized histogram for cat image with bin size 100.



2. Normalized Histogram for cat image with bin size 255 :



3 . Normalized Histogram for Crowd Image with bin size 255 :



- **Calculate the cumulative distribution function CDF from your pdf and plot the function.**

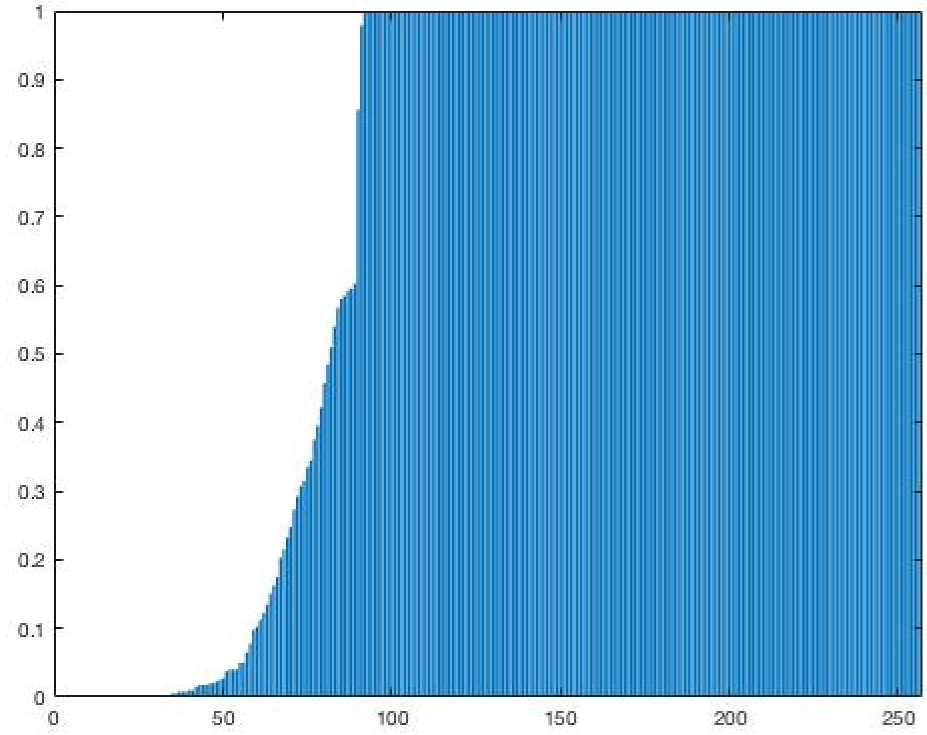
Cumulative Distribution Function (CDF) :

Implementation in Matlab :

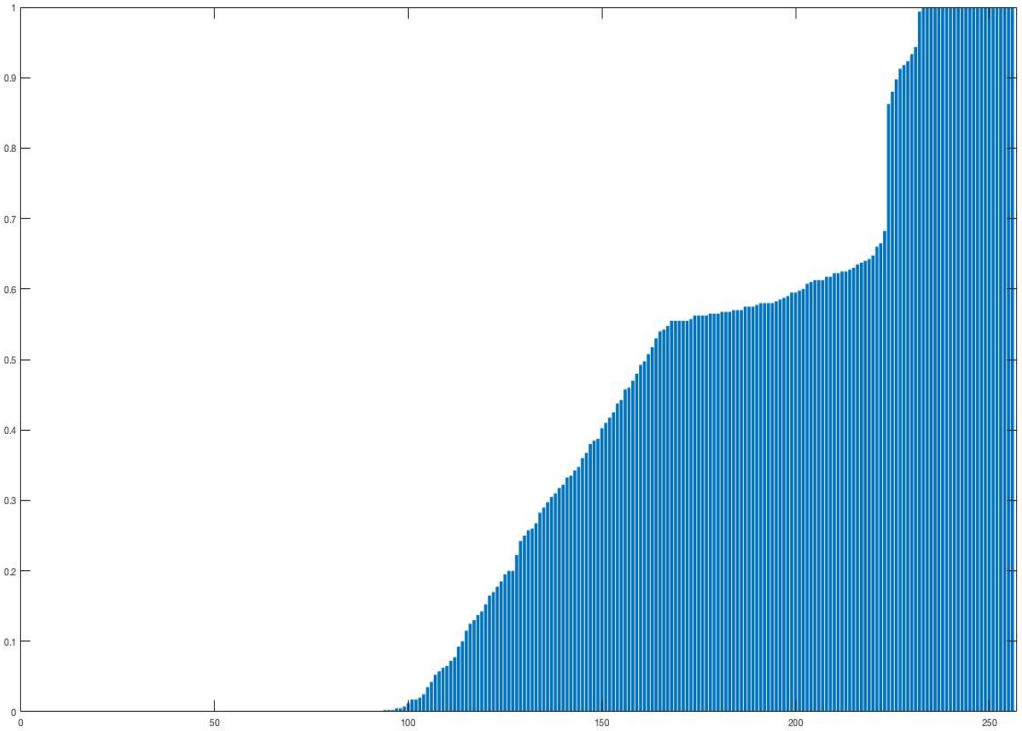
```
function [cumulative_vector] = cdf ( Norm_InputIm_histogram )

cumulative_vector = zeros(256,1);
cumulsum = 0;
for i=1:size(Norm_InputIm_histogram)
   cumulsum = cumulsum + Norm_InputIm_histogram(i);
   cumulative_vector(i)= cumulsum;
end
```

figure, bar(cumulative_vector);

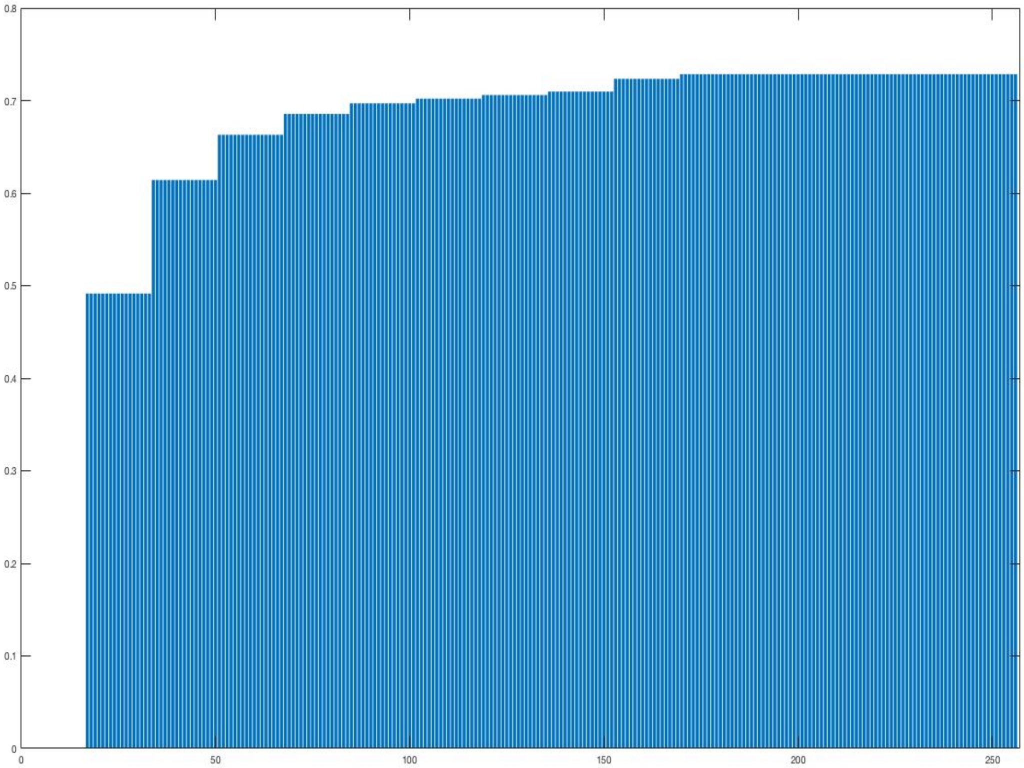
1. CDF of cat image with bin size 100 :



2. CDF of cat image with bin size 255 :



3. CDF of crowd image with bin size 255 :



B2) Histogram Equalization :

Use the histogram code as developed above, and provide an additional function for histogram equalization.

Histogram equalization : It is a technique for adjusting image intensities to enhance contrast. Let f be a given image represented as a mr by mc matrix of integer pixel intensities ranging. from 0 to L − 1. L is the number of possible intensity values, often 256.

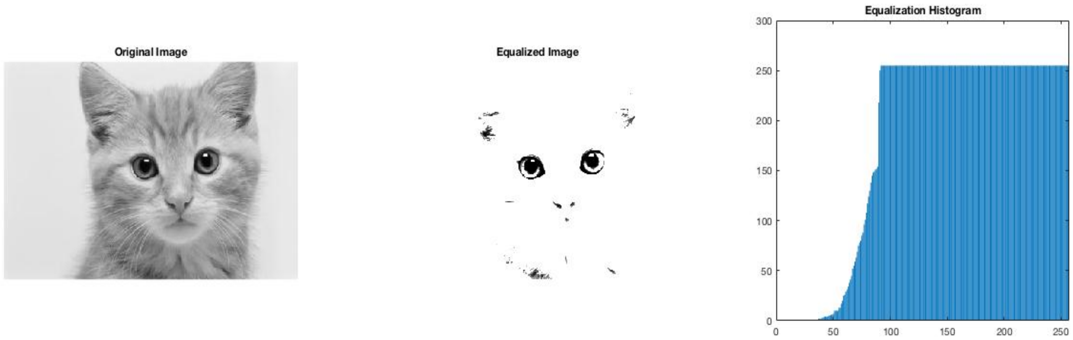
Image Equalization implementation in Matlab:

```
function [OutputImage] = imgeq( InputImage, Norm_InputIm_histogram )

r = size(InputImage ,1 );
c = size(InputImage ,2);
OutputImage = uint8(zeros(r,c));
```

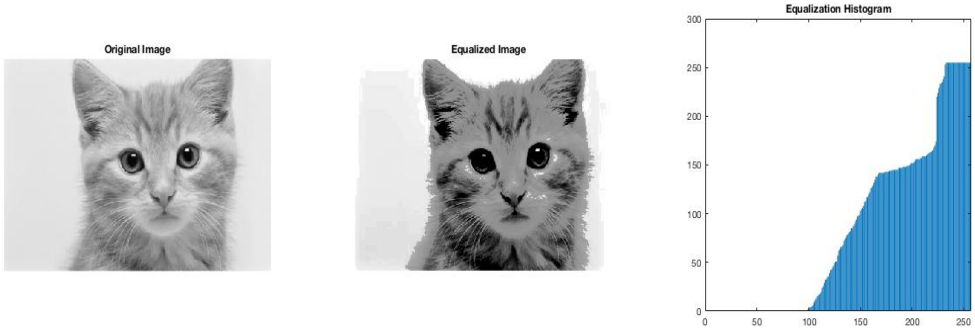
```
norm_cumulsumvec = zeros(256,1);
norm_cumulsum = 0;
output = zeros(256,1);
levels = 255;
for i=1:size(Norm_InputIm_histogram)
   norm_cumulsum = norm_cumulsum + Norm_InputIm_histogram(i);
   norm_cumulsumvec(i)= norm_cumulsum;
   %formula: (L-1)*cdf%
   output(i)= round(norm_cumulsumvec(i)*levels);
end

for i=1:size(InputImage,1)
   for j=1:size(InputImage,2)
      OutputImage(i,j)=output(InputImage(i,j)+1);
   end
end

subplot(2,3,1) , imshow(InputImage);
title("Original Image");
subplot(2,3,2) , imshow(OutputImage);
title("Equalized Image");
subplot(2,3,3) , bar(output);
title("Equalization Histogram");
```

- **Apply your histogram equalization code to the images used before. Calculate and plot the new histogram after equalization.**

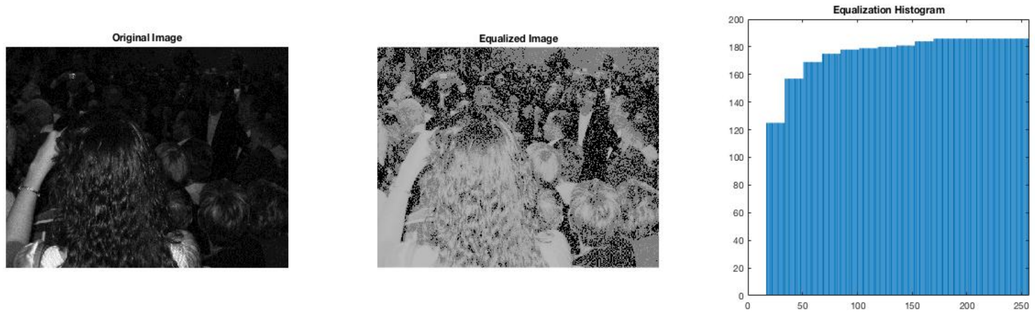
1 . Image Equalization for cat image with bin size 100 :



2. Image Equalization for cat image with bin size 255 :



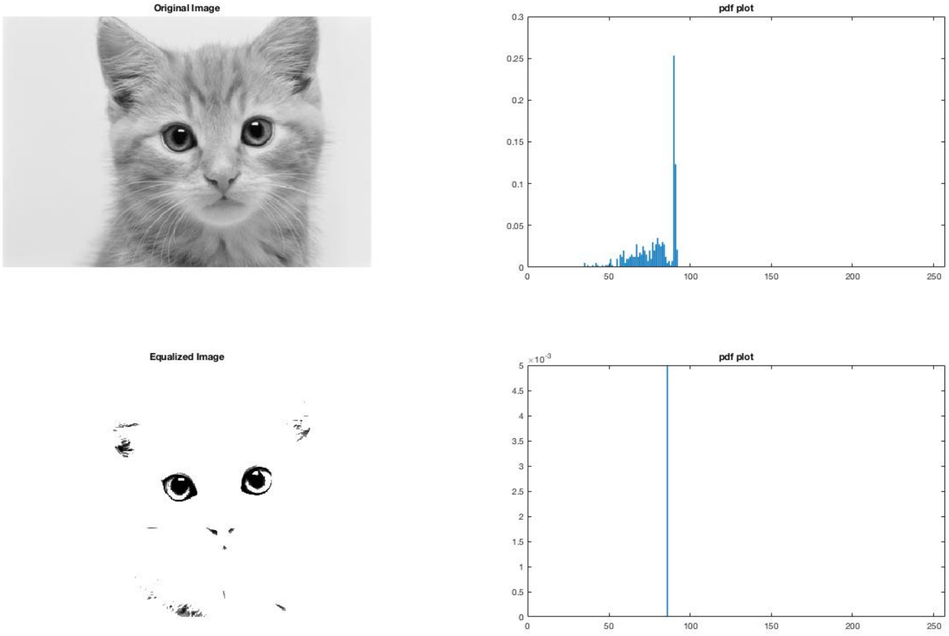
3. Image Equalization for crowd.png image with bin size 255 :



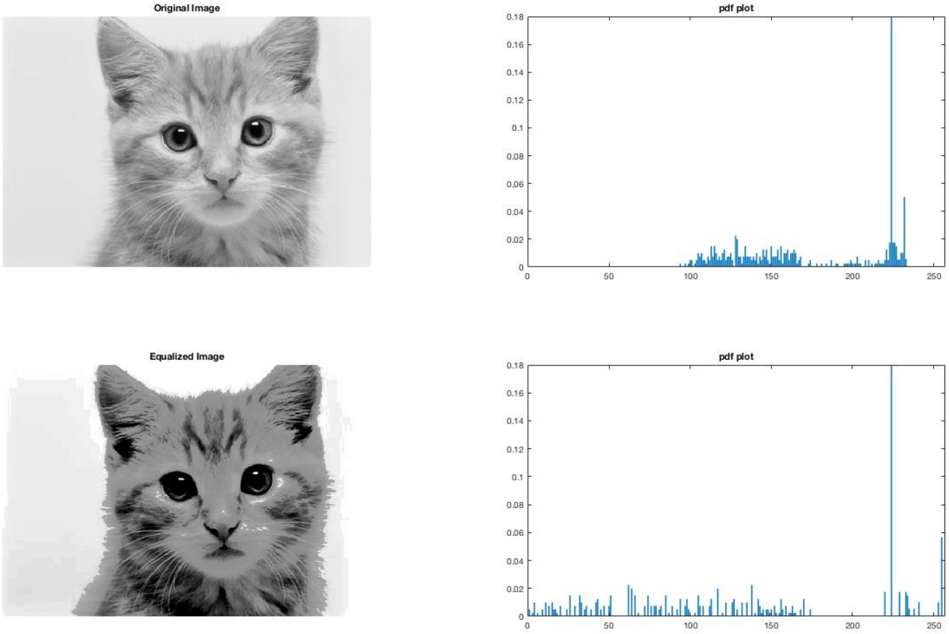
- **Add an additional section to the report by showing images, pdf’s and CDF’s before/after equalization. Briefly discuss what you see in the histogram equalized images and the corresponding plots of pdf’s and CDF’s.**

Pdf Before and After Image Equalization :

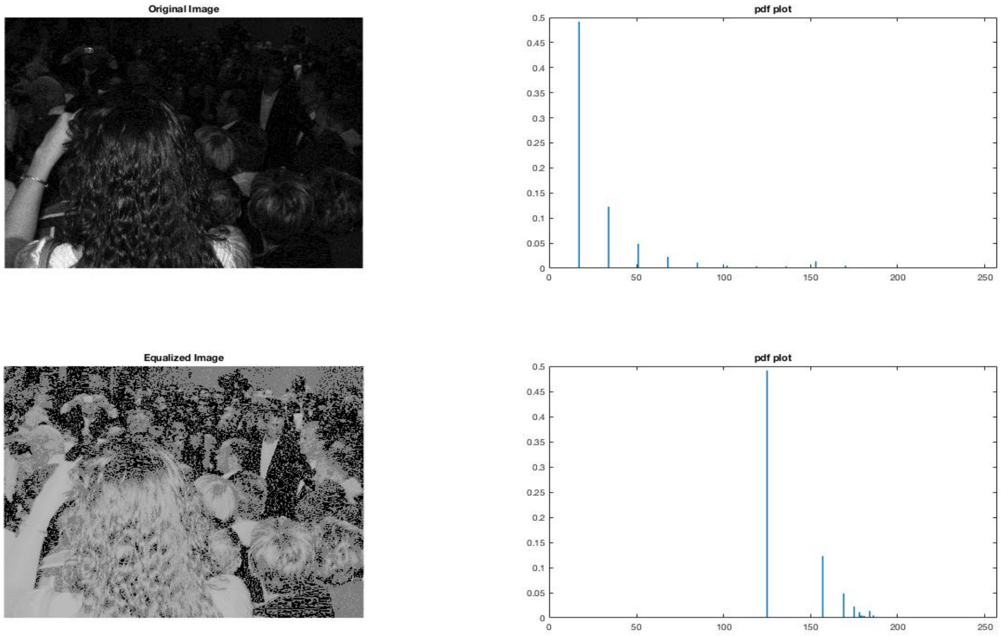
1. Cat Image with bin size 100 :



2. Cat Image with bin size 255 :

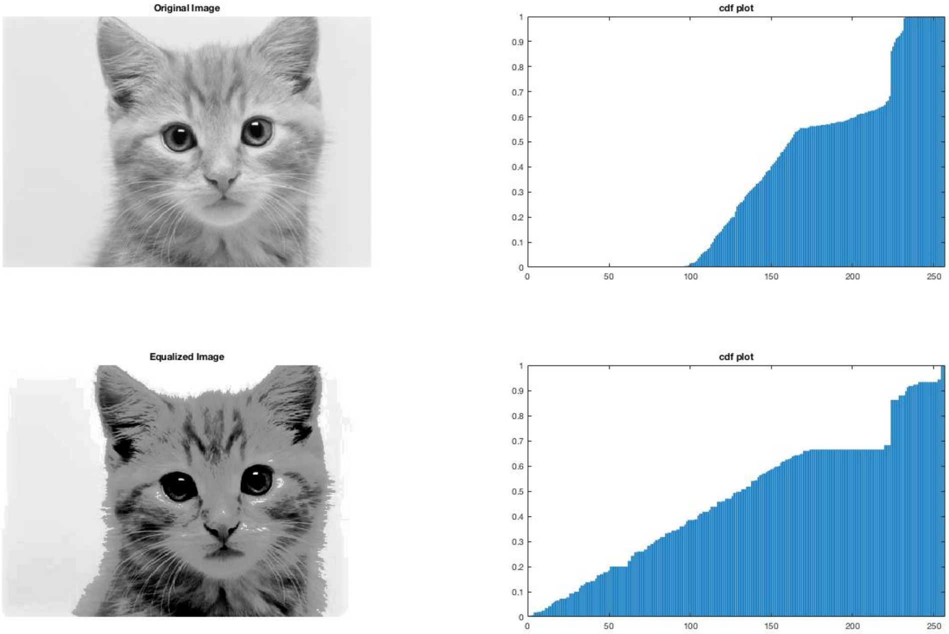


3. Crowd Image with bin size 255 :

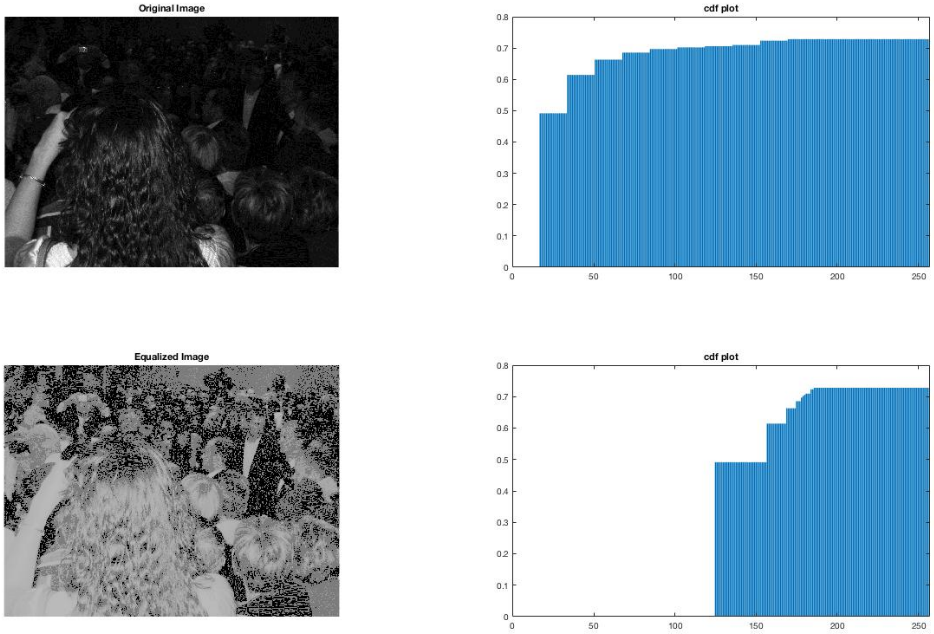


Cdf Before and After Image Equalization :

1. Cat Image with bin size 255 :



2. Crowd Image with bin size 255 :



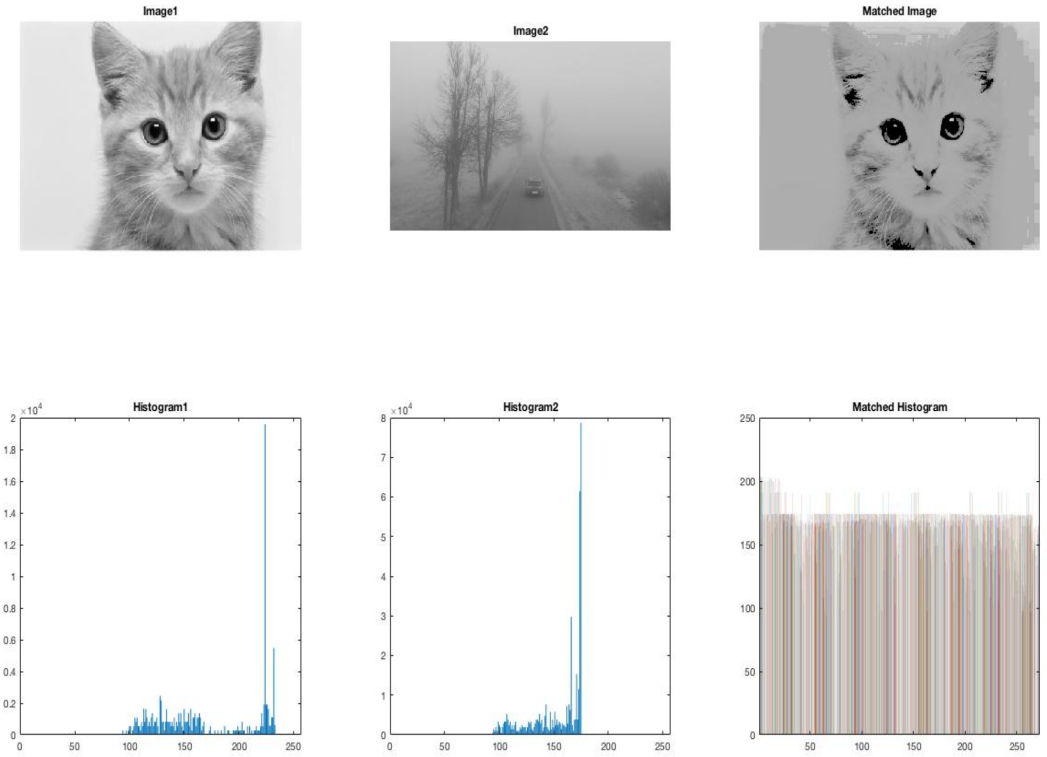
B3) Histogram Matching Following the course notes, develop code that maps intensity values of a preferably bad image into intensity distribution of a good looking image.

Histogram Matching : In Image Processing , histogram matching or histogram specification is the transformation of an image so that its histogram matches a specified histogram.

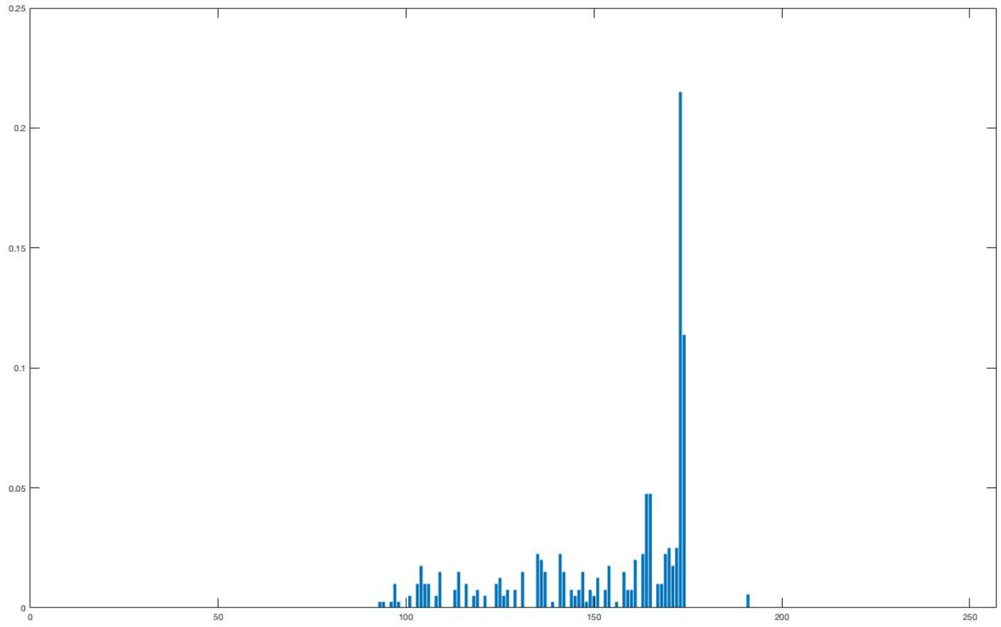
Histogram Matching can be implemented in Matlab using following code :

```
function [out ] = histogram_match ( img1, img2 )

a = histogram(img1, 255 , 0 ,255);
b = histogram(img2, 255 , 0 ,255);

pdf1 = normhistogram(img1,a);
pdf2 = normhistogram(img2,b);

a_cdf = cdf(pdf1);
b_cdf = cdf(pdf2);

M = zeros(256,1,'uint8');
for i = 1:256
    diff = abs(a_cdf(i) - b_cdf);
    [~,ind] = min(diff);
    M(i) = ind - 1;
end

out = M(double(img1)  + 1);
```

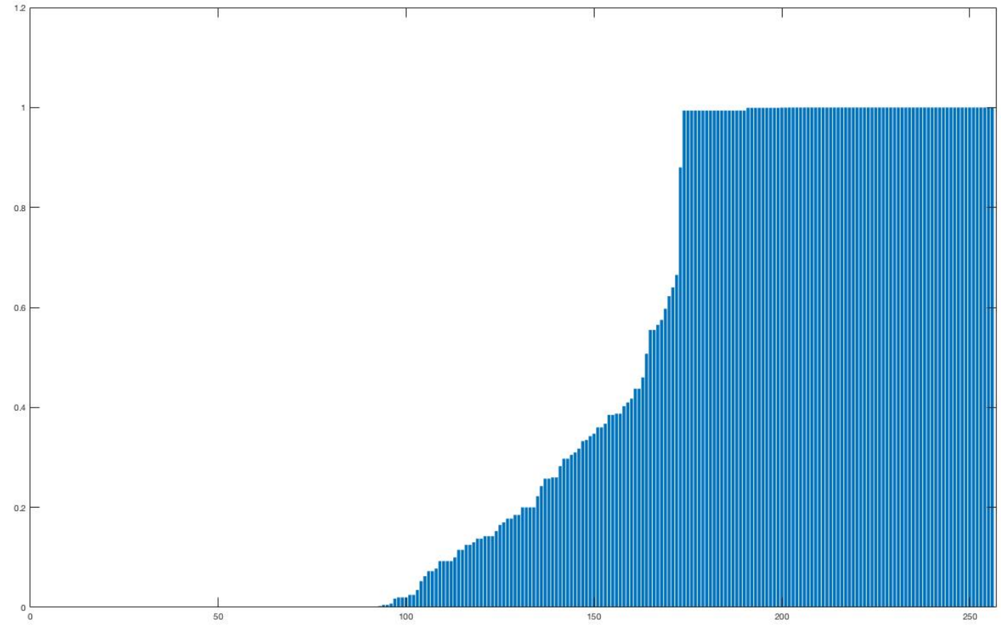
Output of above code would be :



PDF plot of matched Image :



CDF plot of matched Image :



Observation :

We took two images for histogram matching . One with good contrast and another with poor contrast or visibility. The matched image matches the image which has good contrast . As can be seen in the examples above, the matching introduces gaps in the histograms. This is to be expected since the histograms are being distorted.