Image Processing | Lab4

Aim: Understanding Image Histograms and implementing histogram equalization and matching.

Image Histogram:

- An image histogram is a plot of the gray-level frequencies (i.e., the number of pixels in the image that have that gray level).
- Divide frequencies by total number of pixels (m x n image size) to represent as probabilities.

Histogram equalization:

- To improve the contrast of an image.
- To transform an image in such a way that the transformed image has a nearly uniform distribution of pixel values.

Histogram matching:

- Histogram equalization yields an image whose pixels are uniformly distributed among all gray levels.
- Sometimes, this may not be desirable. Instead, we may want a transformation that yields an output image with a pre-specified histogram. This technique is called histogram specification/matching.

Important Functions:

1. Plot histogram of RGB image. (*imhist(im) : Produce histogram counts of image im.)*

```
function void = plot_hist(r)
  red_channel = r(:,:,1);
  green_channel = r(:,:,2);
  blue_channel = r(:,:,3);

  [yRed, x] = imhist(red_channel);
  [yGreen, x] = imhist(green_channel);
  [yBlue, x] = imhist(blue_channel);

  plot(x, yRed, x, yGreen, x, yBlue);
  legend("Red","Green","Blue");
endfunction
```

2. Histogram equalization of grayscale image.

```
function [s,final] = imequalizehist(r)
  L=256;
  [m,n] = size(r);
  hist = zeros(size(L-1));
  for i=0:(L-1),
    hist(i+1) = sum(sum(r==i));
  endfor
  pdf = hist/(m*n);
  total(1) = pdf(1);
  for i=1:(L-1),
    total(i+1) = total(i)+pdf(i+1);
  endfor
  s = (L-1)*total;
  s = round(s);
  final = zeros(m,n);
  for i=0:(L-1),
    final = final + (r==i)*s(i+1);
  endfor
  final = uint8(final);
endfunction
```

3. Histogram equalization of RGB image.

```
function [s, final] = imequalizecolorhist(r)
  [m,n,d] = size(r);
  L=256;

  red_channel = r(:,:,1);
  green_channel = r(:,:,2);
  blue_channel = r(:,:,3);

  [sr, finalr] = imequalizehist(red_channel);
  [sg, finalg] = imequalizehist(green_channel);
  [sb, finalb] = imequalizehist(blue_channel);

  s=zeros(d,L);
  s(1,:)=sr;
  s(2,:)=sg;
  s(3,:)=sb;
  s = uint8(s);

  final = zeros(size(r));
  final(:,:,1) = finalr;
  final(:,:,2) = finalg;
  final(:,:,3) = finalb;
  final = uint8(final);
endfunction
```

4. Histogram matching of grayscale image.

```
function [s,final] = imhistmatch(r,ref)
  L=256;
  [m,n] = size(r);
  [s ,final1] = imequalizehist(r);
  [G, final2] = imequalizehist(ref);
  for i=0:(L-1),
    [val ind(i+1)] = min(abs(G-s(i+1)));
  endfor
  ind = ind-1;
  s=ind;
  final  = zeros(m,n);
  for i=0:(L-1),
    final = final + (r==i)*ind(i+1);
  endfor
  final = uint8(final);
endfunction
```

Tasks:

1. Can two visually different images have the same histogram? If yes, synthesize two grayscale images which are visually different but have the same histogram and also show the histogram. If no, justify your answer.

Yes, two visually different images can have the same histogram. Each column in the histogram represents how many pixels in the photograph have the pixel value represented by the column. Histogram does not tell you where those pixels are located within the image. As a result, two different images can result in the same histogram.

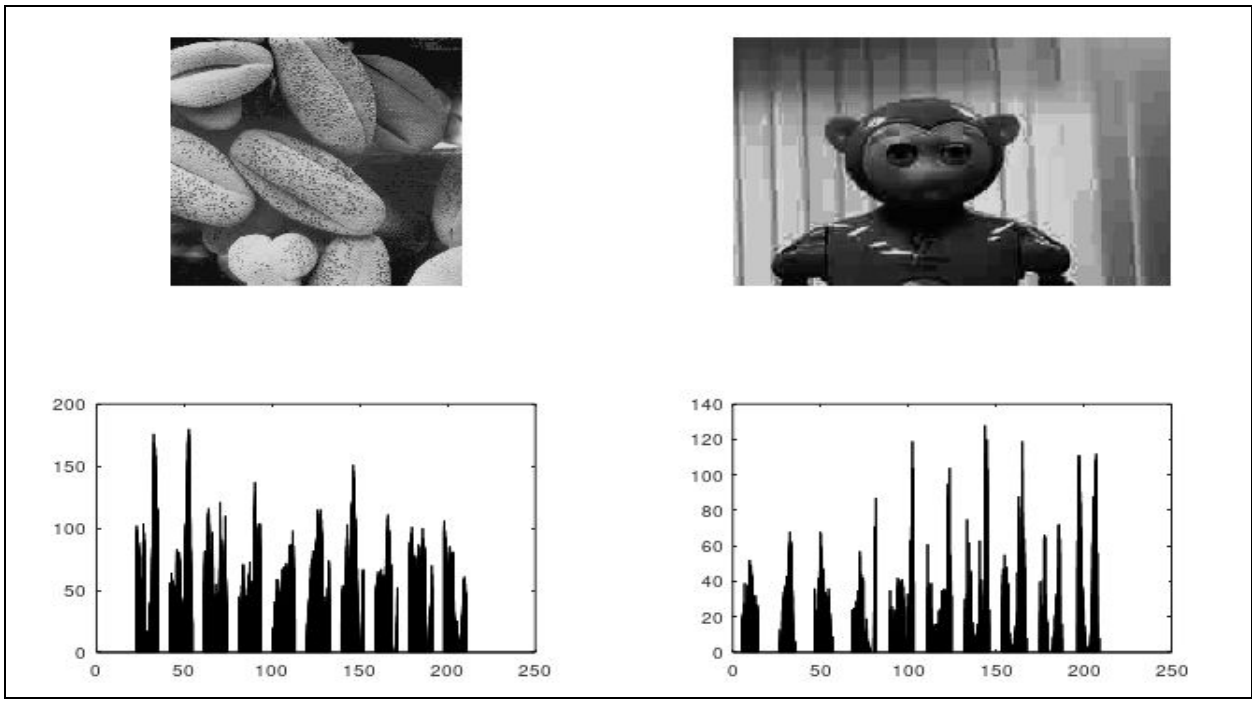
Below two images are created by histogram matching. It seems two images have the same histogram though they are slightly different.

Example:

Code:

```
#Can two visually different image have same histogram? If yes, synthesize two
#grayscale images which are visually different but having same histogram and
#also show the histogram. If no, justify your answer.

i1 = imread('lab4Images/test3.tif');
i2 = imread('lab4Images/1.jpg');
ref = imread('lab4Images/2.jpg');

[s1,r1] = imhistmatch(i1,ref);
[s2,r2] = imhistmatch(i2,ref);

subplot(2,2,1);
imshow(r1);

subplot(2,2,2);
imshow(r2);

subplot(2,2,3);
hist(r1);

subplot(2,2,4);
hist(r2);
```

Output:

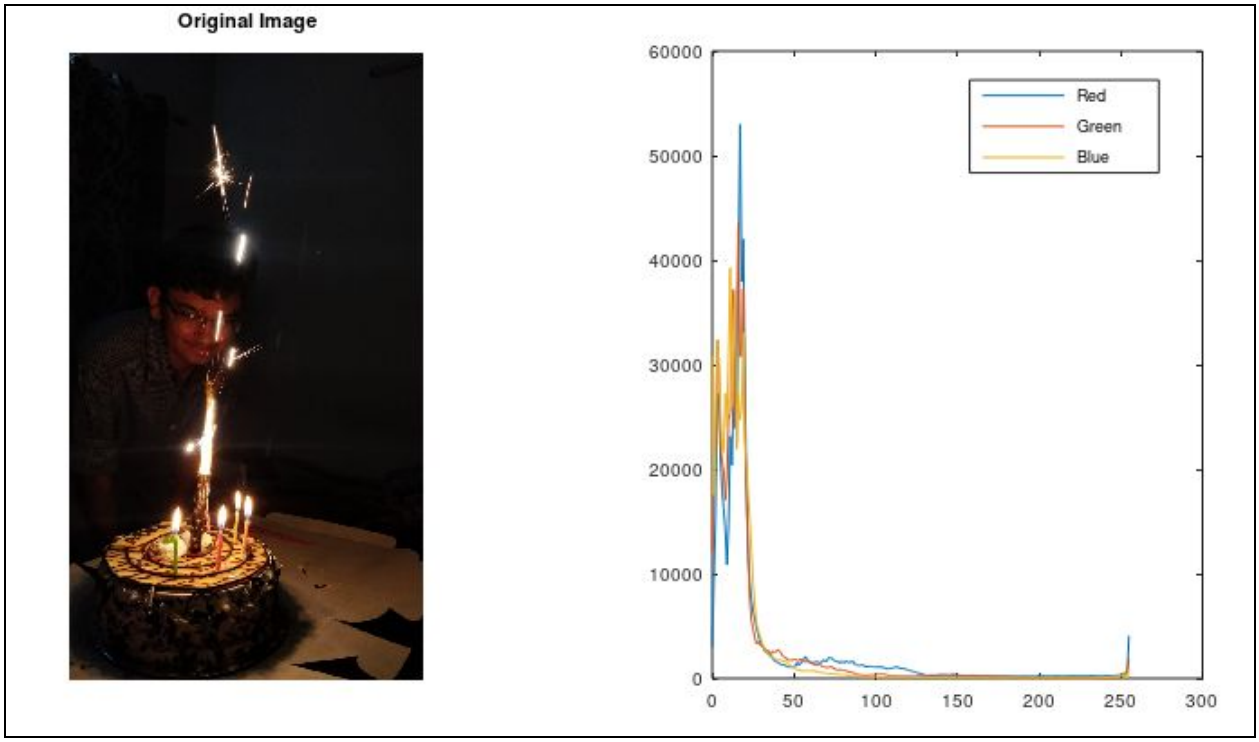


2. Take your color photograph taken in the dark. Equalize it's histogram.

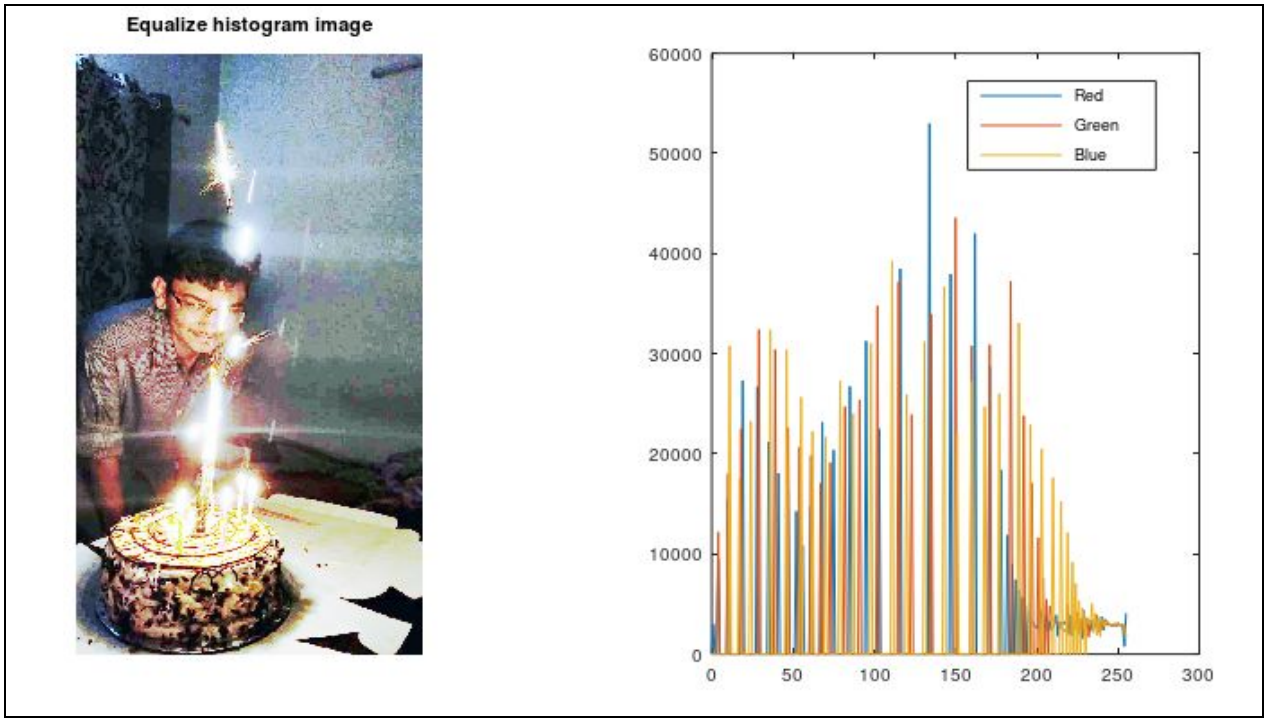
Code:

```
#Take your color photograph taken in dark. Equalize it's histogram.
r = imread('lab4Images/my_dark_photo.jpg');
subplot(1,2,1);
imshow(r);
title('Original Image');

red_channel = r(:,:,1);
green_channel = r(:,:,2);
blue_channel = r(:,:,3);

[yRed, x] = imhist(red_channel);
[yGreen, x] = imhist(green_channel);
[yBlue, x] = imhist(blue_channel);

subplot(1,2,2);
plot(x, yRed, x, yGreen, x, yBlue);
legend("Red","Green","Blue");

[sr, finalr] = imequalizehist(red_channel);
[sg, finalg] = imequalizehist(green_channel);
[sb, finalb] = imequalizehist(blue_channel);

[yRedf, xf] = imhist(finalr);
[yGreenf, xf] = imhist(finalg);
[yBluef, xf] = imhist(finalb);

figure;

s = zeros(size(r));
s(:,:,1) = finalr;
s(:,:,2) = finalg;
s(:,:,3) = finalb;

subplot(1,2,1);
imshow(uint8(s));
title('Equalize histogram image');

subplot(1,2,2);
plot(xf, yRedf, xf, yGreenf, x, yBluef);
legend("Red","Green","Blue");
```

Original image:



Histogram equalized image:

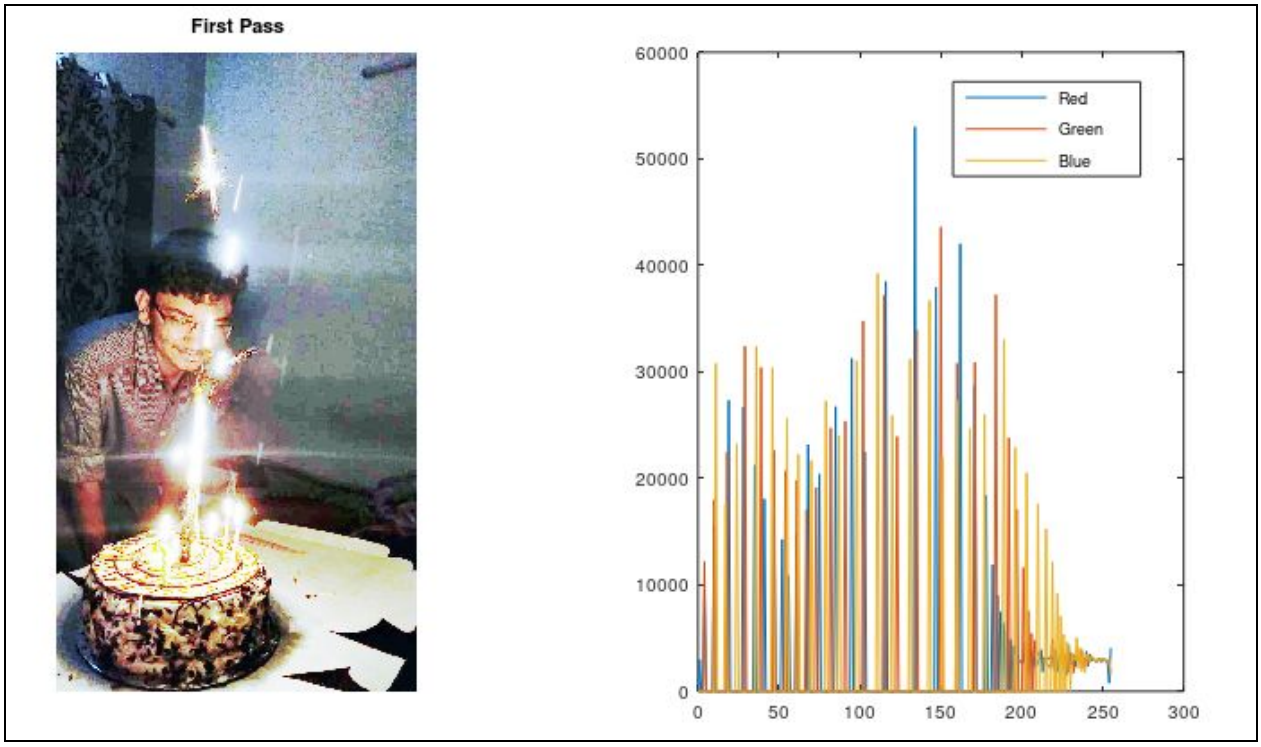


3. Perform histogram equalization of equalized image obtained. Is the second pass of the histogram equalization process useful? Justify your answer.

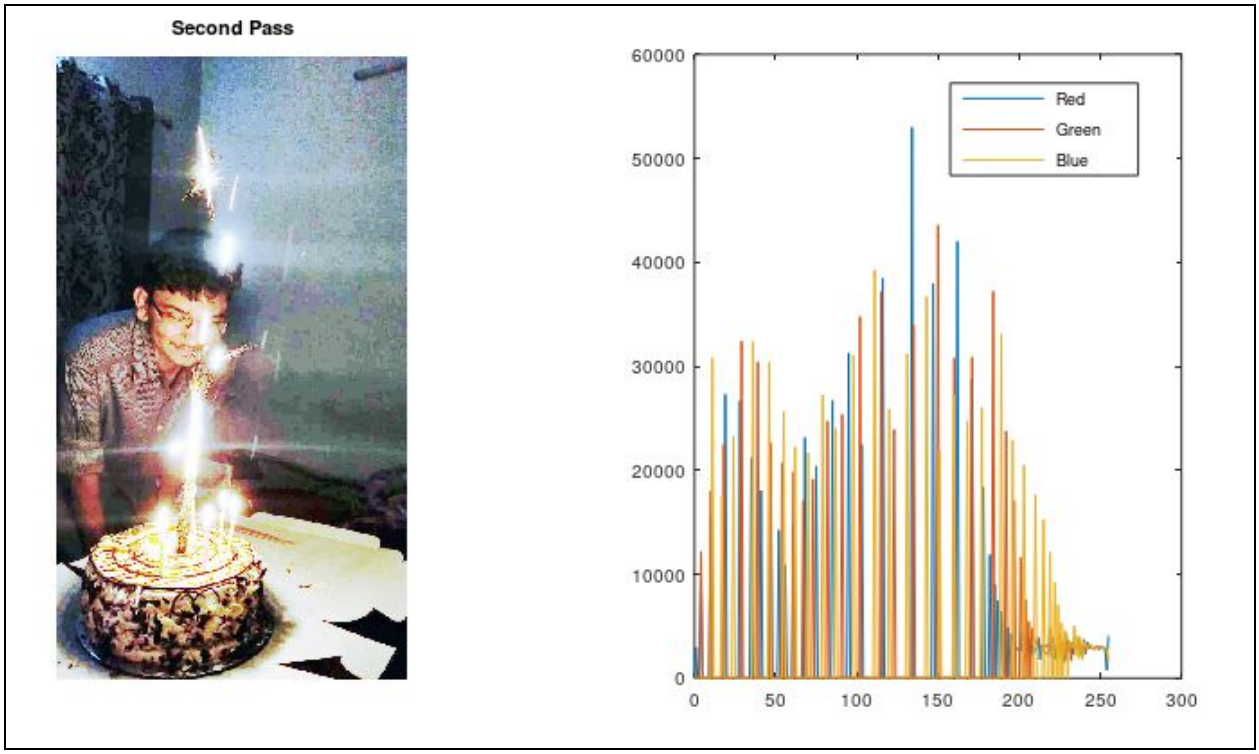
Code:

```
#Perform histogram equalization of equalized image obtained.
# Is second pass of the histogram equalization process useful? Justify your
#answer.
r = imread('lab4Images/my_dark_photo.jpg');
[s1, final1] = imequalizecolorhist(r);
[s2, final2] = imequalizecolorhist(final1);

subplot(1,2,1);
imshow(final1);
title("First Pass");
subplot(1,2,2);
plot_hist(final1);

figure;

subplot(1,2,1);
imshow(final2);
title("Second Pass");
subplot(1,2,2);
plot_hist(final2);
```

First-pass image:



Second-pass image:



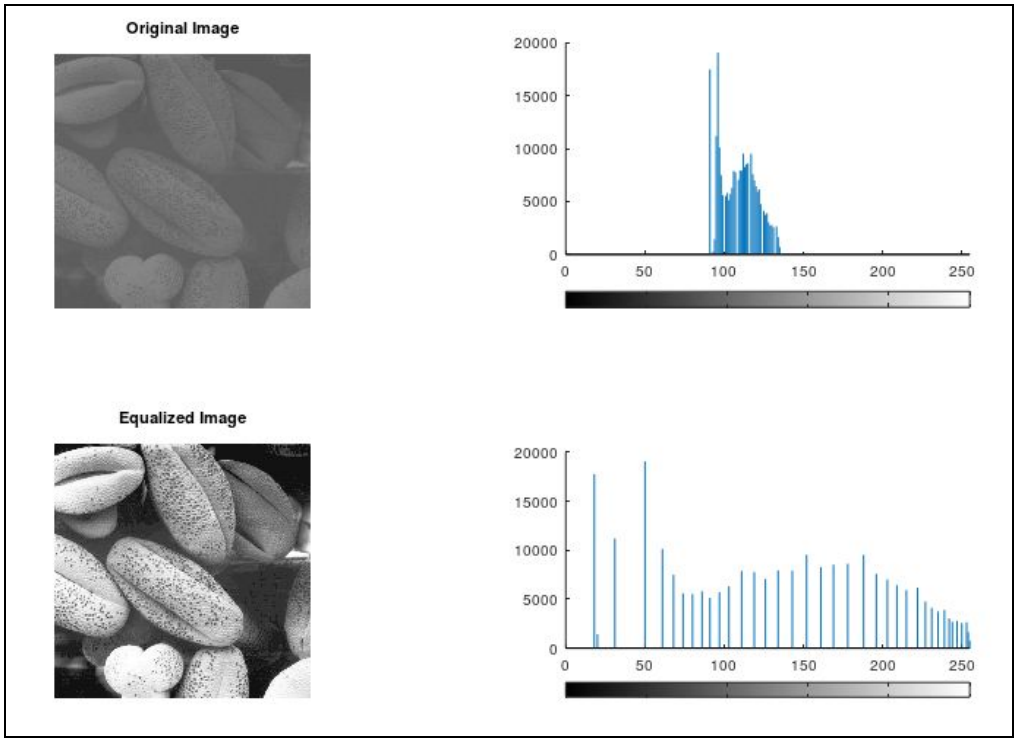
As we see , In second pass of equalized histogram image has no importance and both image and histogram does not change. Because the image is already equalized, equalizing again is not going to make any kind of changes.

4. Perform histogram equalization for image ‘test3.tif’.

Code:

```
#Perform histogram equalization for image 'test3.tif'.
r = imread('lab4Images/test3.tif');
subplot(2,2,1);
imshow(r);
title("Original Image");
subplot(2,2,2);
imhist(r);

[s, final] = imequalizehist(r);
subplot(2,2,3);
imshow(final);
title("Equalized Image");
subplot(2,2,4);
imhist(final);
```

Output:

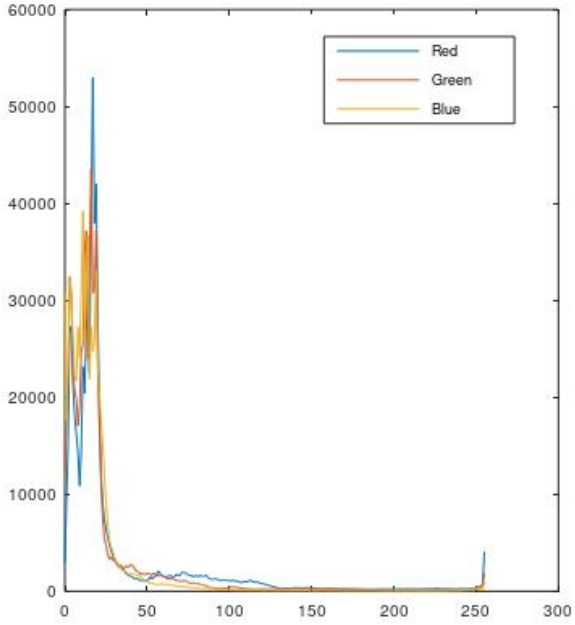


5. Take any of your photographs, match it's histogram with the histogram of image 'test4.jpg'. plot histogram of original image, template and matched image for all three channels.

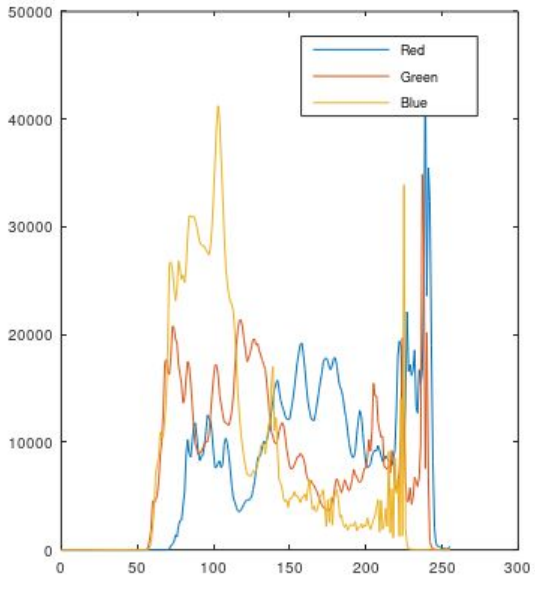
Code:

```
#Take any of your photograph, match it's histogram with the histogram of image
#'test4.jpg'. plot histogram of original image, template and matched image for
#all three channels.

r = imread('lab4Images/my_dark_photo.jpg');
ref = imread('lab4Images/test4.jpg');

subplot(1,2,1);
imshow(r);
subplot(1,2,2);
plot_hist(r);

figure;

subplot(1,2,1);
imshow(ref);
subplot(1,2,2);
plot_hist(ref);

figure;

L=256;
[m,n,d] = size(r);

[sf1, final(:,:,1)] = imhistmatch(r(:,:,1),ref(:,:,1));
[sf2, final(:,:,2)] = imhistmatch(r(:,:,2),ref(:,:,2));
[sf3, final(:,:,3)] = imhistmatch(r(:,:,3),ref(:,:,3));

final = uint8(final);

subplot(1,2,1);
imshow(uint8(final));
subplot(1,2,2);
plot_hist(final);
```

Original Image:



Template Image:



Matched Image: