Digital Image Processing

CPE – 415
CEP

Name	Rohaid Ahmed Mirza
Registration Number	FA19-BCE-006
Class	BCE – 8A
Instructor's Name	Dr. Ahsan Khawaja

Effects of Histogram Equalization followed by Contrast Stretching towards the Vascular Segmentation of Retinal Fundus Images

Introduction:

Vessel detection in fundus images is a difficult task. The complex vascular structure is difficult to preserve in the fundus image. Image acquisition process introduces noise to the image. The noise in the image hides or blurs the vessels.

Image enhancement techniques play a crucial role in medical imaging. It helps remove the noise and clear out the vessel structure Histogram equalization and contrast stretching are popular methods used to improve the visual quality of images. This report explores the application of histogram equalization followed by contrast stretching as a combined approach for image enhancement.

Histogram Equalization:

Histogram equalization is a technique that redistributes the pixel intensities of an image to achieve a more uniform histogram. The goal is to enhance the overall contrast of the image by expanding the dynamic range of pixel values. The process involves the following steps:

a. ***Computing the Histogram:*** The first step is to calculate the histogram of the input image. The histogram represents the frequency distribution of pixel intensities.
b. ***Cumulative Distribution Function (CDF):*** The cumulative distribution function is calculated by summing up the histogram values. It represents the accumulated probability of each intensity level.
c. ***Histogram Equalization Transformation:*** The next step involves transforming the pixel values of the input image based on the CDF. Each pixel value is replaced with its corresponding CDF value multiplied by the maximum intensity level.
d. ***Resulting Image:*** The transformed image obtained through histogram equalization exhibits an improved contrast by redistributing the pixel intensities.

Contrast Stretching:

Contrast stretching is a technique used to further enhance the dynamic range of an image. It works by linearly mapping the original pixel values to a new range of intensities. The process involves the following steps:

a. ***Specifying the Target Range:*** The desired range of intensities is defined. This range typically spans the entire dynamic range from 0 to the maximum intensity level.
b. ***Minimum and Maximum Intensity Values:*** The minimum and maximum pixel intensity values in the input image are determined.

c. ***Stretching Transformation:*** The pixel values are linearly mapped to the target range using the formula: new_pixel_value = (pixel_value - min_intensity) * (new_max - new_min) / (max_intensity - min_intensity) + new_min

d. ***Resulting Image:*** The output image obtained through contrast stretching exhibits an expanded range of intensities, resulting in improved contrast and enhanced visual details.

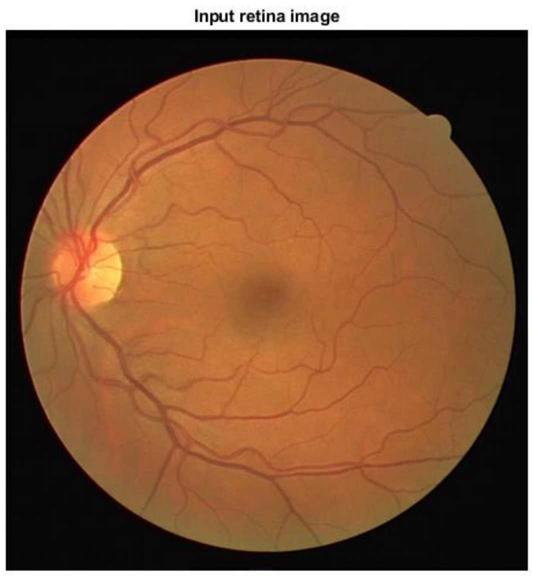
Combined Approach:

Histogram Equalization Followed by Contrast Stretching The combination of histogram equalization and contrast stretching provides a powerful technique for image enhancement. By applying histogram equalization first, the overall contrast of the image is improved, and the dynamic range is expanded. However, this process may result in some loss of detail in regions that originally had high contrast.

To mitigate this issue, contrast stretching is applied after histogram equalization. Contrast stretching further expands the dynamic range, allowing finer control over the intensity distribution. This approach helps to preserve the details in high-contrast areas while simultaneously enhancing the overall image quality.

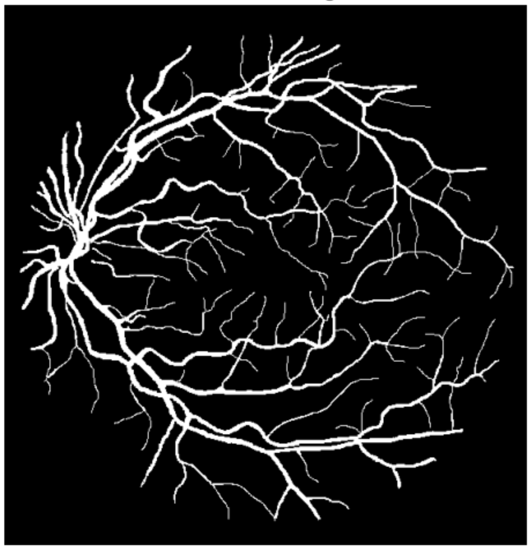
Enhancement with The Approved Technique:

```
clear all;
clc;
I = imread('retina_images\1.tif');
I = imresize(I,.8);
figure, imshow(I);title('Input retina image');
```



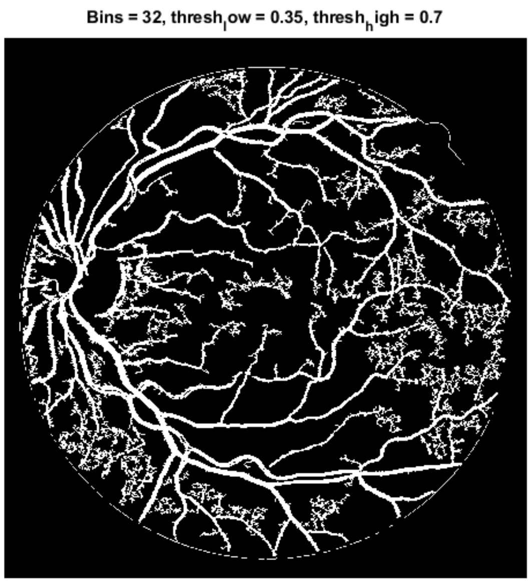
```
input = rgb2gray(I);
gt1 = imread('label_images\1.tif');
gt1 = imresize(gt1,.8);
gt_image = im2bw(gt1);
figure, imshow(gt1);title('Ground truth image');
```

Ground truth image

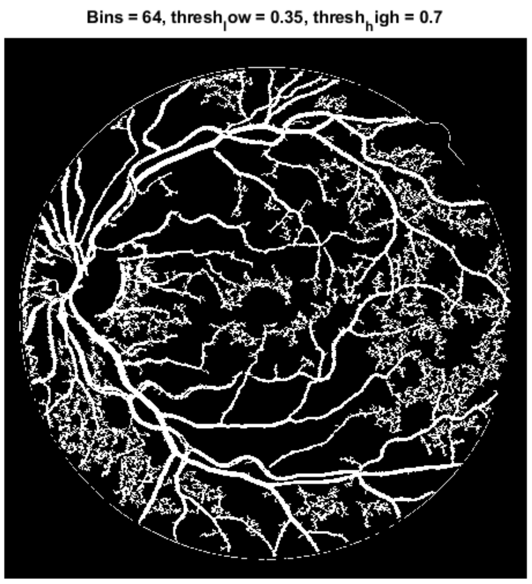


```
% segmented_image = segmentRetina(input);
segmented_image1 = segmentRetina(input,32,0.35, 0.7);
segmented_image2 = segmentRetina(input,64,0.35, 0.7);
segmented_image3 = segmentRetina(input,32,0.2, 0.8);
segmented_image4 = segmentRetina(input,64,0.2, 0.8);
segmented_image5 = segmentRetina(input,128,0.2, 0.8);
```

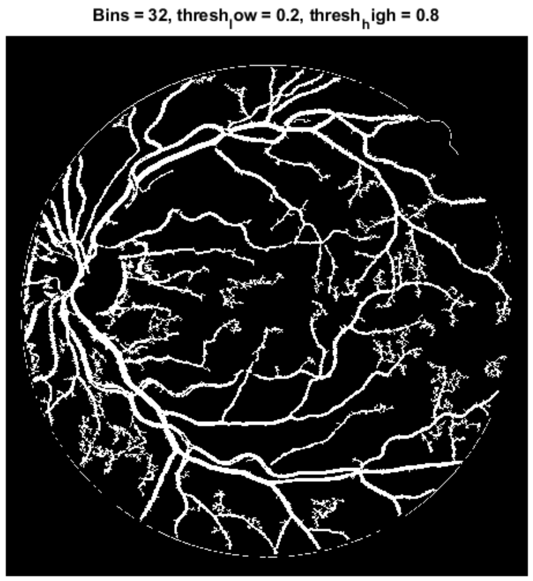
```
figure,imshow(segmented_image1);title('Bins = 32, thresh_low = 0.35,
thresh_high = 0.7');
```



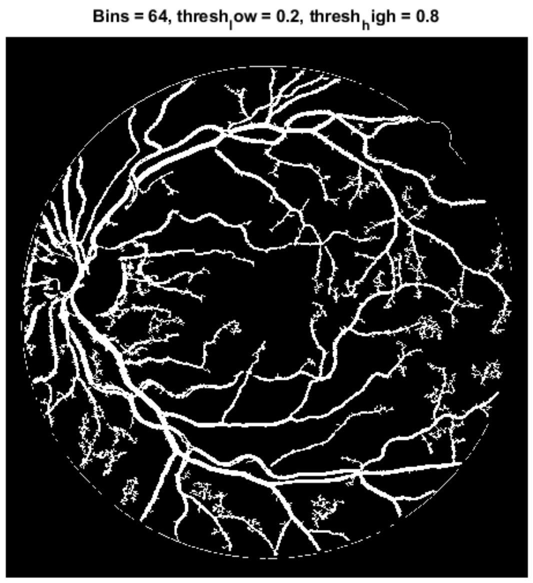
```
 figure,imshow(segmented_image2);title('Bins = 64, thresh_low = 0.35, 
thresh_high = 0.7');
```



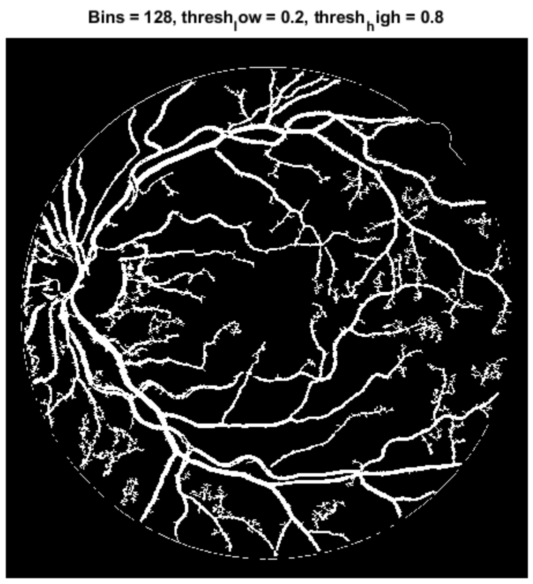
```
 figure,imshow(segmented_image3);title('Bins = 32, thresh_low = 0.2, thresh_high 
= 0.8');
```



```
 figure,imshow(segmented_image4);title('Bins = 64, thresh_low = 0.2, thresh_high
= 0.8');
```



```
 figure,imshow(segmented_image5);title('Bins = 128, thresh_low = 0.2,
thresh_high = 0.8');
```



```
% Calculate evaluation metrics for each segmented image
[P1, N1, T1] = calculatePNT(segmented_image1, gt_image);
[P2, N2, T2] = calculatePNT(segmented_image2, gt_image);
[P3, N3, T3] = calculatePNT(segmented_image3, gt_image);
[P4, N4, T4] = calculatePNT(segmented_image4, gt_image);
[P5, N5, T5] = calculatePNT(segmented_image5, gt_image);

% Display the evaluation metric values
fprintf('Bins = 32, thresh_low = 0.35, thresh_high = 0.7\n');
```

```
Bins = 32, thresh_low = 0.35, thresh_high = 0.7
```

```
fprintf('Value of P is: %.2f\n', P1);
```

```
Value of P is: 81.78
```

```
fprintf('Value of N is: %.2f\n', N1);
```

```
Value of N is: 92.34
```

```
fprintf('Value of T is: %.2f\n', T1);
```

```
Value of T is: 91.39
```

```

fprintf('\nBins = 64, thresh_low = 0.35, thresh_high = 0.7:\n');
```

```
Bins = 64, thresh_low = 0.35, thresh_high = 0.7:
```

```
fprintf('Value of P is: %.2f\n', P2);
```

```
Value of P is: 83.07
```

```matlab
fprintf('Value of N is: %.2f\n', N2);
```

```
Value of N is: 90.75
```

```matlab
fprintf('Value of T is: %.2f\n', T2);
```

```
Value of T is: 90.07
```

```matlab

fprintf('\nBins = 32, thresh_low = 0.2, thresh_high = 0.8:\n');
```

```
Bins = 32, thresh_low = 0.2, thresh_high = 0.8:
```

```matlab
fprintf('Value of P is: %.2f\n', P3);
```

```
Value of P is: 79.68
```

```matlab
fprintf('Value of N is: %.2f\n', N3);
```

```
Value of N is: 94.91
```

```matlab
fprintf('Value of T is: %.2f\n', T3);
```

```
Value of T is: 93.55
```

```matlab

fprintf('\nBins = 64, thresh_low = 0.2, thresh_high = 0.8:\n');
```

```
Bins = 64, thresh_low = 0.2, thresh_high = 0.8:
```

```matlab
fprintf('Value of P is: %.2f\n', P4);
```

```
Value of P is: 79.42
```

```matlab
fprintf('Value of N is: %.2f\n', N4);
```

```
Value of N is: 95.22
```

```matlab
fprintf('Value of T is: %.2f\n', T4);
```

```
Value of T is: 93.80
```

```matlab

fprintf('\nBins = 128, thresh_low = 0.2, thresh_high = 0.8:\n');
```

```
Bins = 128, thresh_low = 0.2, thresh_high = 0.8:
```

```matlab
fprintf('Value of P is: %.2f\n', P5);
```

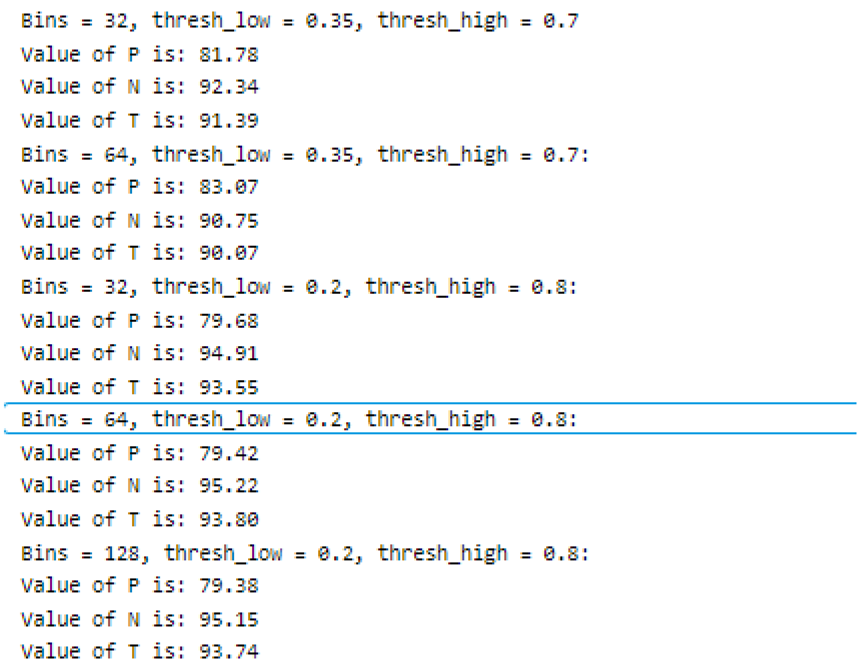
```
Value of P is: 79.38
```

```matlab
fprintf('Value of N is: %.2f\n', N5);
```

```
Value of N is: 95.15
```

```
fprintf('Value of T is: %.2f\n', T5);
```

```
Value of T is: 93.74
```



```
Bins = 32, thresh_low = 0.35, thresh_high = 0.7
Value of P is: 81.78
Value of N is: 92.34
Value of T is: 91.39
Bins = 64, thresh_low = 0.35, thresh_high = 0.7:
Value of P is: 83.07
Value of N is: 90.75
Value of T is: 90.07
Bins = 32, thresh_low = 0.2, thresh_high = 0.8:
Value of P is: 79.68
Value of N is: 94.91
Value of T is: 93.55
Bins = 64, thresh_low = 0.2, thresh_high = 0.8:
Value of P is: 79.42
Value of N is: 95.22
Value of T is: 93.80
Bins = 128, thresh_low = 0.2, thresh_high = 0.8:
Value of P is: 79.38
Value of N is: 95.15
Value of T is: 93.74
```

Discussion:

The variant of the combined technique with ***64 bins*** and ***lower threshold value of 0.35 & higher threshold value of 0.8*** proves to be performing the best among the other 4 variants.