ECE253 HW2

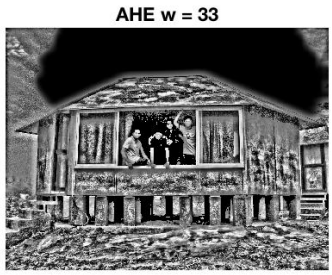
Name: Zhouhang Shao
PID: A99086018

Problem 1: Adaptive histogram equalization

The resulting image with different window is size is attached below

- The original image

- Image after global histogram equlization and AHE



- **How does the original image qualitatively compare to the image after AHE and HE respectively.**

  HE and AHE both enhance the contrast of the original image. We can observe more details in the enhanced images. For all the image after enhancement, they look brighter compared to the original one.However, all the images after histogram equalization seems unreal compared to the original one

- **Which strategy (AHE or HE) works best for beach.png and why? Is this true for any image in general?**

  AHE works better on the beach.png. I think the reason probably would be that the image contains regions that are significantly lighter or darker than than most of the image. Thus, the normal histogram equalization does not work very well.

  This conclusion will not always hold in general. Since AHE only operates on a small region, when the pixels in the neighborhood are very similar, the mapping function will map the narrow range of pixel values to the entire intensity range in the resulting image. In this case, AHE tends to enhance subtle area but are very sensitive to the noise. Thus, it will not work very well for the noisy image.

Source Code

- **AHE.m(the adaptive histogram equalization function)**

```
% This function is designed to perform the adaptive histgram equalization
% Before the operation, it will first pad the image based on the window
% size
% for each pixel, it will perform histogram equalization around the certain
% regions. The region is defined to be a square , it's size always equals
% to win_size
function [output] = AHE(image, win_size)
    %pad the image based on the window size
    pad_size = floor(win_size/2);
    paddedImage = padarray(image, [pad_size,pad_size], 'symmetric');
    [height,width] = size(paddedImage);
    output= uint8(zeros(size(image,1), size(image, 2)));
    %perform Adaptive histogram equalization
    for x =  1 + pad_size : height - pad_size
        for y = 1 + pad_size  : width - pad_size
            rank = 0;
            %iterate through the window around certer pixel
            for i = x - pad_size :  x + pad_size
                for j = y - pad_size : y + pad_size
                    if paddedImage(x,y) > paddedImage(i,j)
                        rank = rank + 1;
                    end
                end
            end
            intensity = 255 * (rank/(win_size * win_size));
            output(x- pad_size, y - pad_size) = intensity;
        end
    end
end
```

- **P1.m(script for problem 1)**

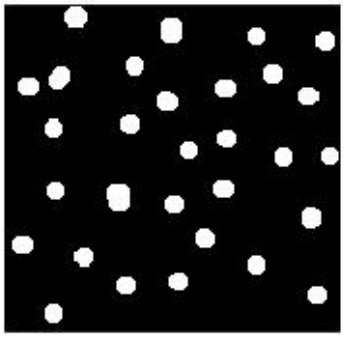
```
image = imread('beach.png');
imshow(image);
win_size = 129;
pad_size = floor(win_size/2);
disp(pad_size);
paddedImage = padarray(image,[pad_size,pad_size],'symmetric');
[height,width] = size(paddedImage);
output= uint8(zeros(size(image,1), size(image, 2)));

padImage33 = AHE(image,33);
padImage65 = AHE(image, 65);
padImage129 = AHE(image, 129);
histEq = histeq(image);
figure, subplot(2,2,1)
imshow(histEq);
title('hist')
subplot(2,2,2);
imshow(padImage33);
title('AHE w = 33')
subplot(2,2,3);
imshow(padImage65);
title('AHE w = 65')
subplot(2,2,4);
imshow(padImage129);
title('AHE w = 129')
```

Problem 2: Binary Morphology

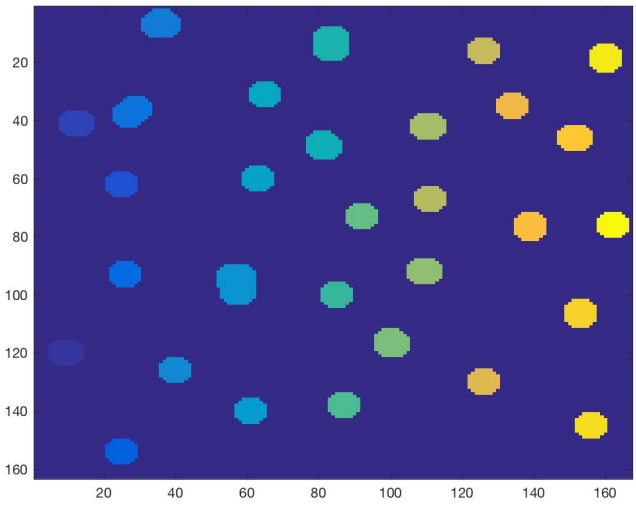
Saperate circles:

- **Structure element:** Disk with radius 5 pixel
- The original image

- The image after opening



* The image after connected componets labeling



* The following contains both the area and centroid data for each connected components

1	2	3	4	5	6
108	111	103	102	103	130
(120,9.5)	(40.89,12.62)	(62,25.23)	(153.525.5)	(93,26.23)	(36.88,28.11)
7	**8**	**9**	**10**	**11**	**12**
120	111	226	115	95	105
(6.5,36)	(126.1,39.37)	(97.32,57.15)	(140.38,61.41)	(59.61,62.87)	(31.83,64.86)
13	**14**	**15**	**16**	**17**	**18**
142	156	69	102	97	119
(49.24,82.92)	(13.09,83.79)	(64,82)	(99.75, 84.95)	(138,87)	(73.75,92.48)
19	**20**	**21**	**22**	**23**	**24**
105	117	137	98	95	95
(116.63, 100.4)	(91.79,109.21)	(41.36,112.34)	(66.5,111)	(15.38,126.13)	(130.87,126.4)
25	**26**	**27**	**28**	**29**	**30**
108	95	151	88	110	102
(35,133.5)	(76.87,138.61)	(44.64,149.87)	(106.5,153.5)	(144.35,156.13)	(18.74,159.95)

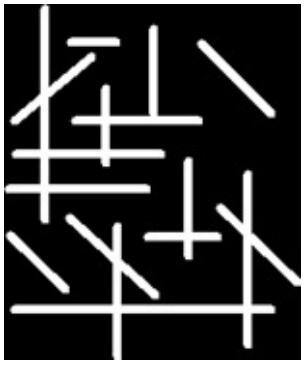
Source code

```
image = imread('circles_lines.jpg');
image = rgb2gray(image);        %do we need to convert to gray scale first?
image = imbinarize(image);
imshow(image);

SE = strel('disk',5);
circle = imopen(image,SE);
write = figure;
imshow(circle);
saveas(write,'circle_only.jpg')

%find the connected components
ccCircles = bwlabel(circle);
write = figure;
imagesc(ccCircles);
saveas(write,'connect_component_circles.jpg');

%calculate the area for each connected componenets
numElement = max(ccCircles(:));
circleArea = zeros(numElement,1);
circleCentroid = zeros(numElement,2);

%looping through the circle image and find the area for each connected
```

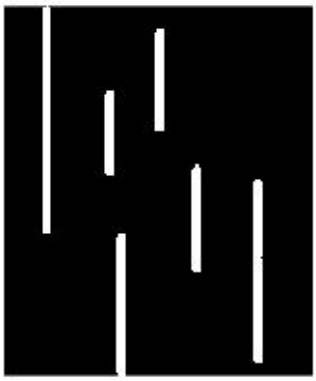
```
%componenets
for x = 1 : size(ccCircles,1)
    for y = 1 : size(ccCircles, 2)
        if ccCircles(x,y) ~= 0
            circleArea(ccCircles(x,y),1) = circleArea(ccCircles(x,y),1) + 1;
            circleCentroid(ccCircles(x,y),1) =  circleCentroid(ccCircles(x,y),1) + x;
            circleCentroid(ccCircles(x,y),2) =  circleCentroid(ccCircles(x,y),2) + y;
        end
    end
end

% calculating the centroid
for i = 1 : numElement
   circleCentroid(i,1) =  circleCentroid(i,1)/circleArea(i,1);
   circleCentroid(i,2) =  circleCentroid(i,2)/circleArea(i,1);
end
```

Seperate lines:

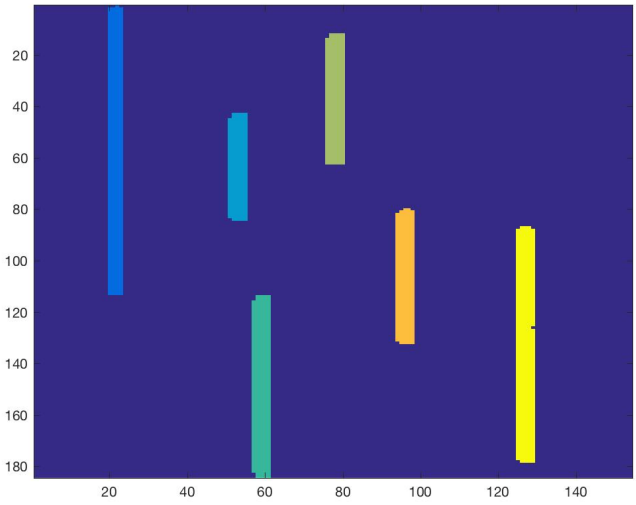
- The original image



* The image after opening



- The image after connected componets labeling


20
40
60
80
100
120
140
160
180
20
40
60
80
100
120
140