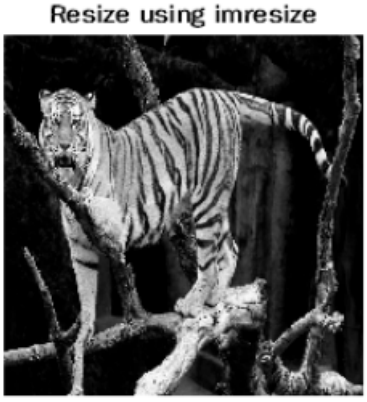
ISYE 8803 Homework 2

Shreecharran Sundar

Question 1 Part I. Image Resizing

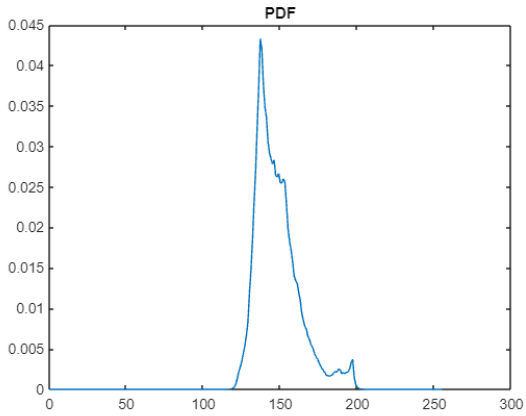
Image resize techniques try to interpolate the image intensity values for the pixels of the resized image that do not directly mapped to the original image. In this part we want to resize an image using bilinear interpolation. The original size of "tiger.jpg" is 853×1280. Write your own code to resize the image to 1500×1500. Plot your original and resized images (You cannot use "imresize" or other equivalent built- in functions).

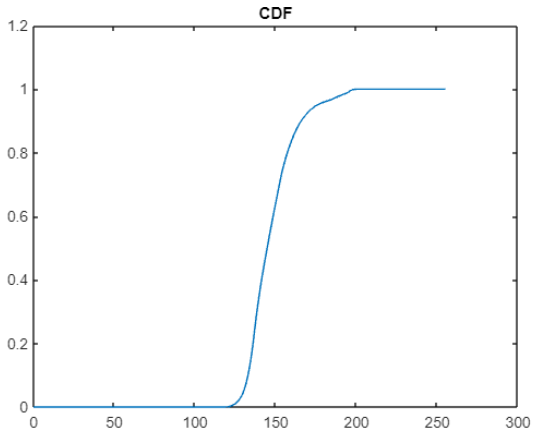
The code for my bilinear interpolation implementation is in the attached matlab file "Q1A.m". The resized image from my implementation:

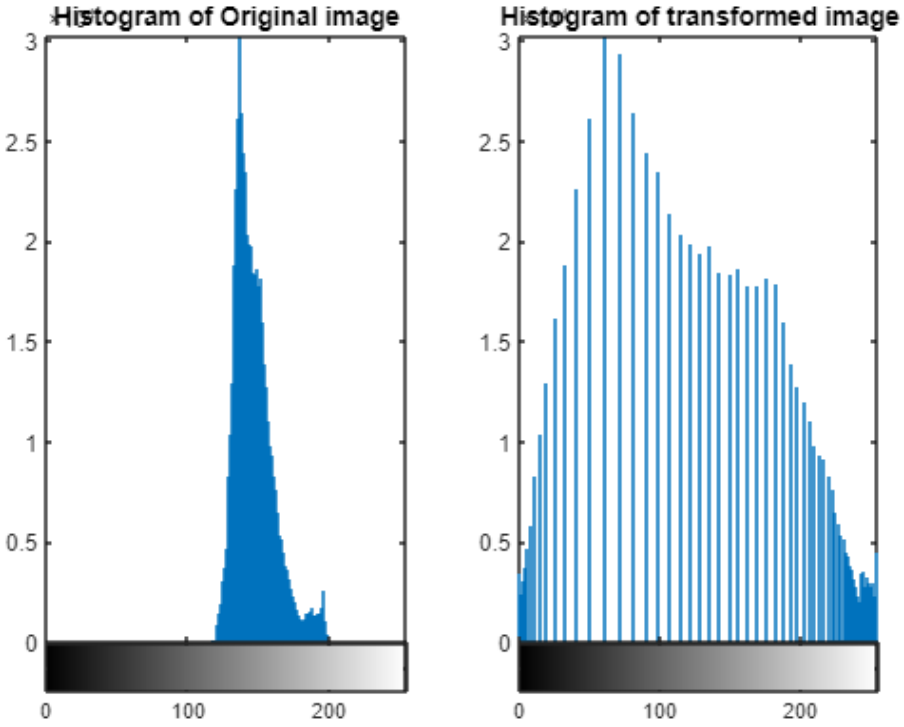


Question 1 Part II. Image intensity transformation

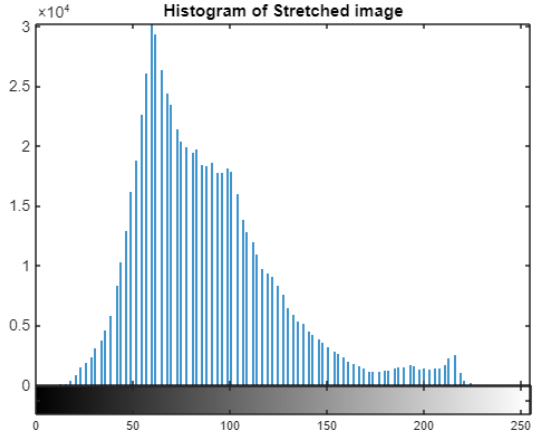
(a)







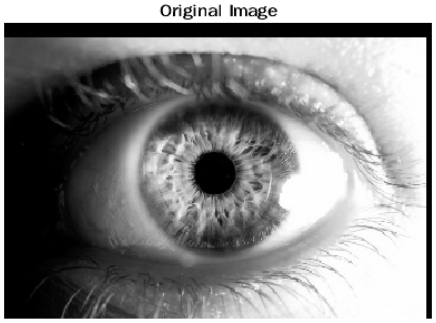
(b)



Both the methods are able to enhance the image (increase the contrast). In the case of histogram equalisation, the pixels are spread out more evenly and uniformly throughout the range (0-255) in comparison to histogram stretching. The histogram equalization image has more proportion of high pixels than histogram stretching. (You can observe that the first image is brighter in comparison to the second)

Question 2. Hybrid Images

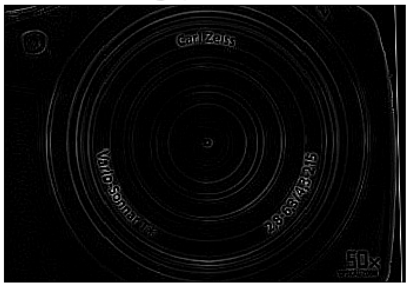
Original Image



Gaussian filter

Orginal image

High pass filter



Hybrid image

High frequency tends to dominate perception at small distances and only the low frequency part of the image can be seen from far. By blending the high frequency portion of one image with the low-frequency portion of another, we get a hybrid image that leads to different interpretations at different distances. From very close it looks more like a camera and from far away it looks like an eye.

Question 3

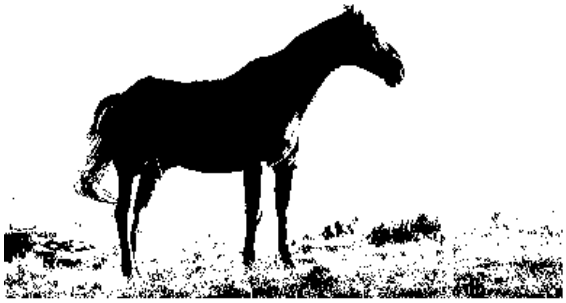
(a and b) Size of original image is 563 x 1000 x 3 ; Size of image after resize is 188 x 334 x 3

Orginal image

Resize image

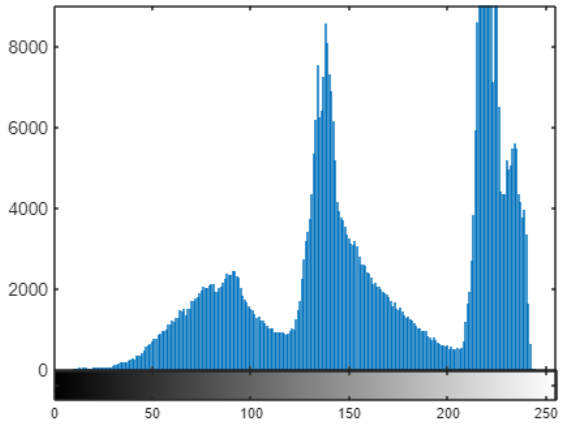
(c)

Gray image



Black and white image (level = 0.5)

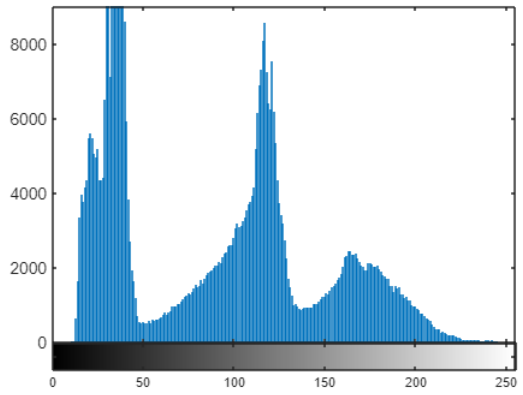
(d) From the histogram you can observe that the image has multimodal distribution. The image also has higher number of bright pixels as compared to darker pixels



Histogram of gray image

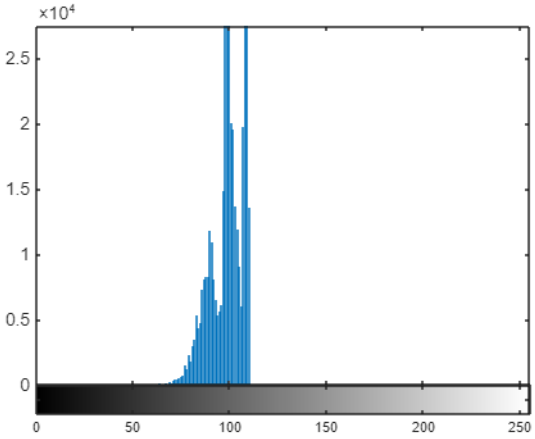
(e) Linear tranformation for negative images.

Linear transfomation



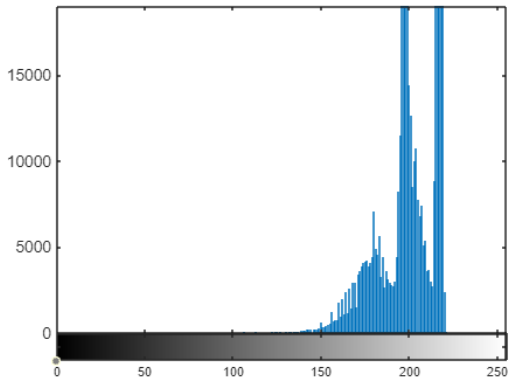
Histogram of linear transformation

Log-Transformation; c=20



Histogram of log-transformation; c=20

Log-Transformation; c=40

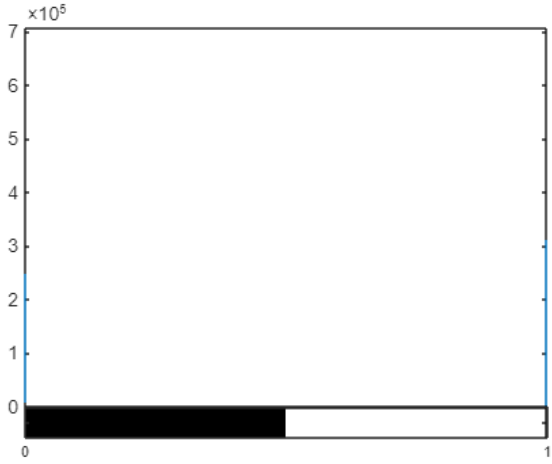


Histogram of log-transformation; c=40

Threshold (100)

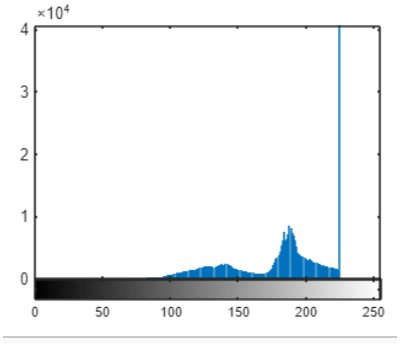
Histogram of Threshold (100)

Threshold (150)



Histogram of threshold (150)

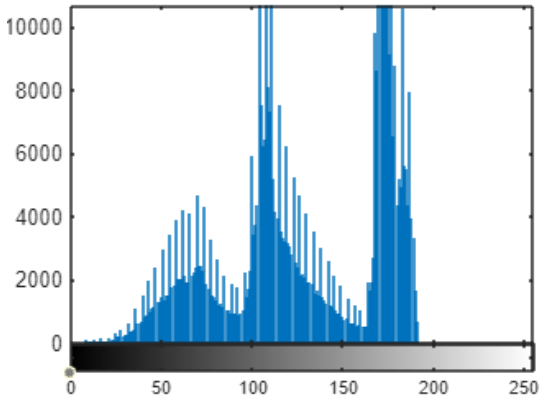
Histogram Shifted image



Histogram of histogram shifted image

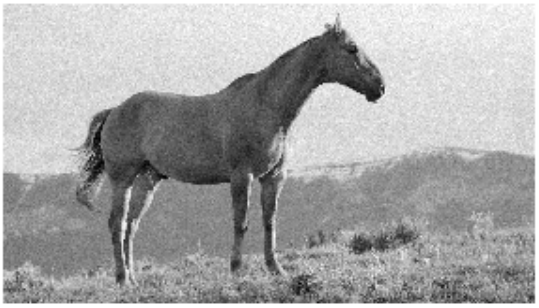
5. Contrast of original image is 253; contrast after stretching is 200

Histogram stretch image



Histogram of stretched image

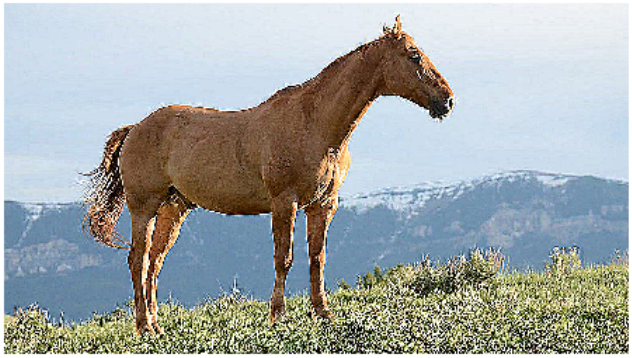
(f)



Noisy image

Denoised image

(g)



Sharpened image

Table: Purpose of each Transformation

Transformation	Purpose
Linear transformation	Obtaining negative image
Log transformation	Image enhancement controlled by c value (Dark pixels get brighter)
Threshold	Converting gray scale to black and white image
Histogram shift	To control the brightness of the image
Histogram stretch	To control the contrast of the image

Listing: Code

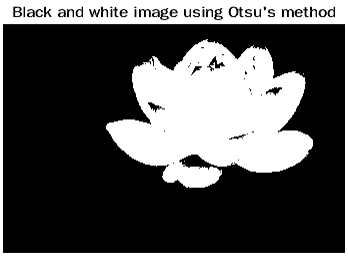
```
%3A
I=imread("horse1-2.jpg"); figure; imshow(I)
%3B
res=imresize(I,1/3); figure; imshow(res)
%3C
gray=rgb2gray(I); figure; imshow(gray)
bw=im2bw(I,0.5); figure; imshow(bw)
%3D
figure; imhist(gray)
%3E
e=double(gray);
e1= uint8(255-e); figure; imshow(e1)
figure; imhist(e1)
e21= uint8(20*log(e+1)); figure; imshow(e21); figure; imshow(e21)
e22= uint8(40*log(e+1)); figure; imshow(e22); figure; imhist(e22)
e31= e>100; figure; imshow(e31); figure;imhist(e31)
e32= e>150; figure; imshow(e32); figure; imhist(e32)
e4 = e + 50;
e4(e>175) = 225;
e4(e<=-25) = 25;
figure; imshow(uint8(e4)); figure; imhist(uint8(e4))
e5= (e-min(min(e)))/(max(max(e))-min(min(e)));
e5=e5*200;
figure; imshow(uint8(e5));figure; imhist(uint8(e5))
old_cont=max(max(e))-min(min(e));
%3F
filter1 = ones(3,3)/9;
f1 = uint8(double(gray) + normrnd(0,10,563,1000));
f2 = imfilter(f1,filter1);
figure;imshow(f1)
figure;imshow(f2)
%3G
filter2 = [-1 -1 -1;-1 9 -1;-1 -1 -1];
g = imfilter(I,filter2);
figure; imshow(g)
```

Question 4

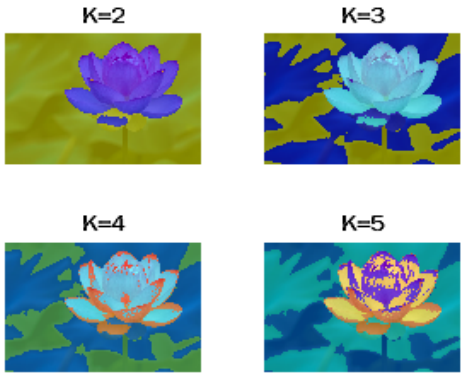
(a) Transform the colored image to a gray scale image.

(b) For the gray image, use Otsu's method to determine the optimal threshold for a black and white image. Report the threshold value you obtained.

Optimal threshold from Otsu's method is 95.3725 (threshold level of 0.3725)



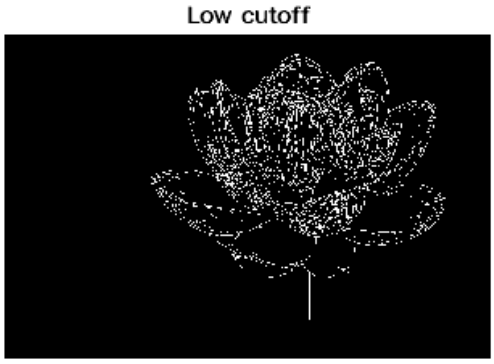
(c) For the colored image, use K-means clustering algorithm to segment the image. You can use any number of clusters.

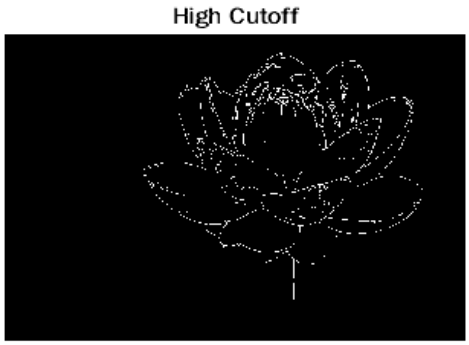


K-Means clustering for 4 different k values

(d) Apply the Sobel operator to the gray image to detect the edges in the image. Use different cutoff values. What happens when the cutoff increases?

Lower cutoff values are noisy but can detect more edges. Higher cutoff values are less noisy but do not capture some of the edges.





(e) For the colored image, use another convolution mask to detect the edges in the image. Report the convolution mask of your choice and compare the result with the Sobel's

The convolution mask I used for edge detection is: (Laplacian edge detection)

$$M = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

It uses a single symmetric kernel unlike the sobel operator which uses different kernels for horizontal and vertical direction. It is more sensitive to noise than sobel operator since it is using second derivative.