Sebastian Zdarowski

October 20, 2019

CSC 481

Assignment 3 (100 points)

From the text:

7.1 *(2/2)*

Give the percentages of red(x), green(y) and blue (Z) light required to generate the point labeled warm white in Fig 7.5

Ans: x +y +z = 1, X axis has a value of .43 (that will be red color), green color(y axis has a value of .40

.43 + .40 + z = 1

z = .17

Breakdown of Warm White:

43 % Red

40 % Green

17% Blue

7.2 *(7/7)*

Consider any two valid colors c1 and c2 with coordinates (x1,y1) and (x2,y2) in the chromaticity diagram of fig. 7.5 Derive the necessary general expressions for computing the relative percentages of colors c1 and c2 composing any color that is known to lie on the straight line joining these two colors.

Ans: Use Euclidian distance to obtain distances between c1 and c2, assume c3 is the color line between the colors.

C3 = (x3,y3)

distance1 = square root((x2-x1)^2 + (y2-y1)^2)

distance2 = square root((x3-x1)^2 + (y3-y1)^2)

Percentage of C1 in C3 would be P = (distance1 – distance2) / distance1

To obtain how much C2 is in C3, we would just subtract 1 from p(calculated above)

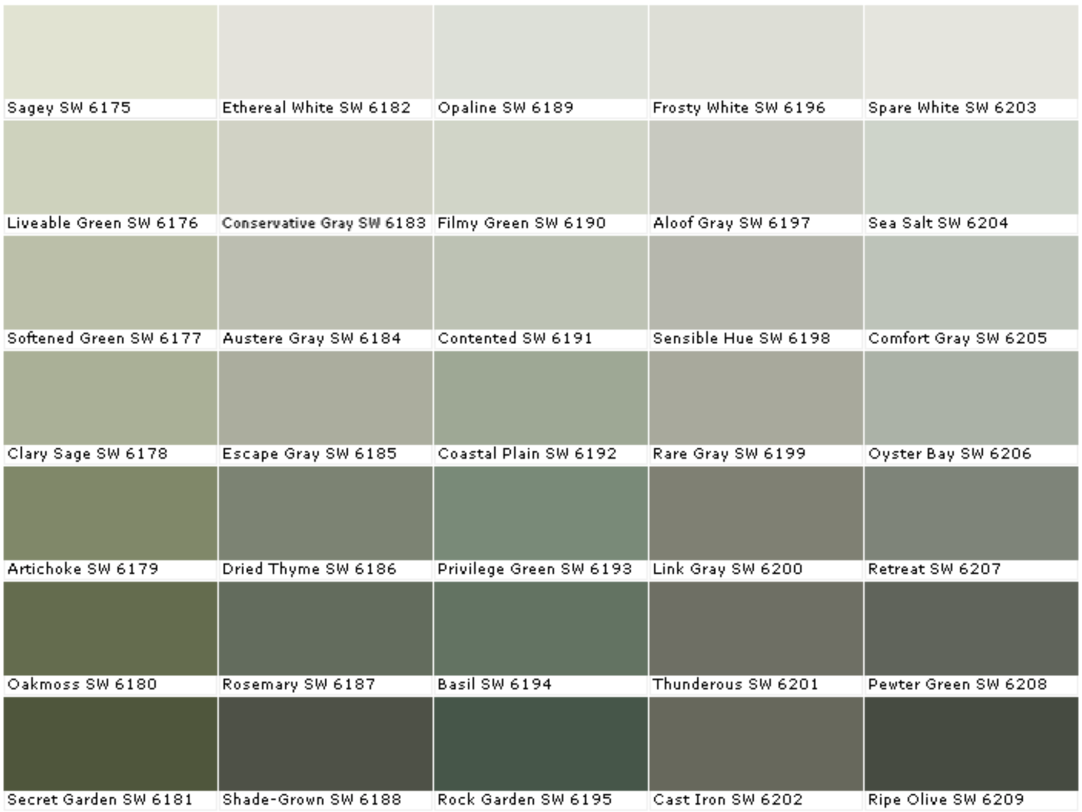
1-P= P2

7.5 *(4/4)*

The R, G and B component images of an RGB image have the horizontal intensity profiles shown in the following diagram, What Color would a person see in the middle column of this image?

Ans: Within the chart, all other values are .5 so I will assume that the image is gray when all values are set to .5. So if the Green value is projecting a 1, I would assume that the image would be a gray green.

Sherman Williams style paint would be between Artichoke SW and OakMoss



7.8b ***(7/7)***

Suppose that we replace every color in the RGB cube by its CMY color. Label with a color name the weight vertices of the new cube that you would see on the screen.

Ans

Use transformation formula: RGB = 1 – CMY

CMY = 1 - RGB

Original Cartesian Coordinates

Blue (0,0,1)

Magenta (1,0,1)

Red(1,0,0)

Yellow(1,1,0)

Green(0,1,0)

Cyan(0,1,1)

Black(0,0,0)

White(1,1,1)

Transformed to CMY model cube would have these coordinates

Blue (1,1,0)

Magenta (0,1,0)

Red (0,1,1)

Yellow(0,0,1)

Green(1,0,1)

Cyan(1,0,0)

Black(1,1,1)

White(0,0,0)

481 Students: 7.14a *(5/0)*

Give the gray-level values for all regions in the hue image

Equal proportions of Red, Blue and Green so its hue is zero, Black no proportions of red, blue and green, so its hue is also zero.

Gray level images are 8 bit images, Hue values range from 0 to 360 so this means hue value are now 360/255.

 # 43 is Yellow (43 x (360/255)) roughly 60 Degrees

 # 85 is Green (85 x (360/255)) roughly 120 degrees

 # 128 is Blue= (128 x(360/255)) roughly 180 degrees

 # 0 = (0x(360/255)) = 0 so the image is white

Programming:

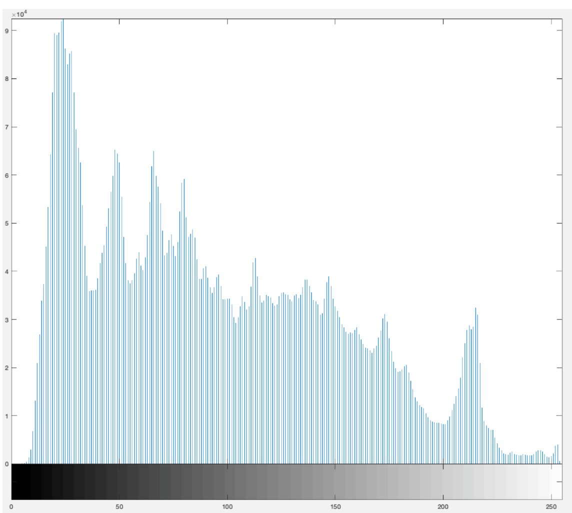
Histogram Equalization *(20/30)*

a) Read and display an image of your choice. (To help you with the next problem, look for an image that has interesting objects of different intensities.)

```
% Homework 3 Sebastian Zdarowski
X = imread('collins3.jpg');
Y = rgb2gray(X);
imshow(Y);
```

b) Calculate and display the histogram of this image.

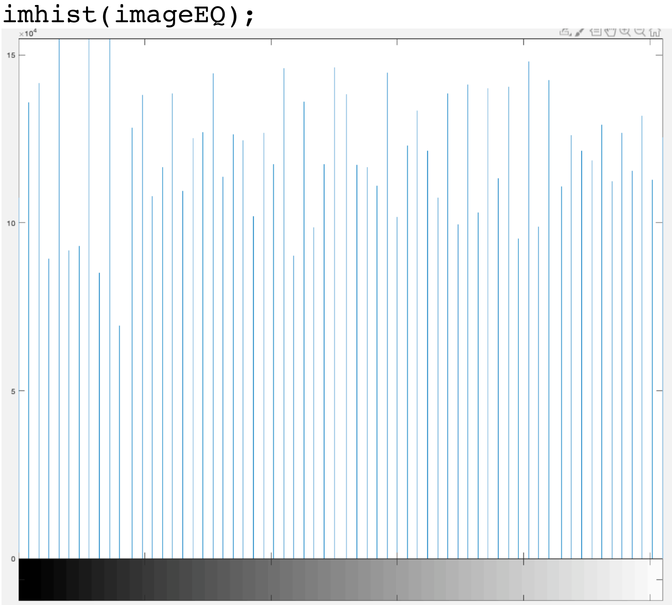
```
imhist(Y);
```



c) Enhance the contrast of the intensity image using histogram equalization and display both the uniform histogram and the new enhanced intensity image.

```
imageEQ = histeq(Y);
imshow(imageEQ);
```

```
imhist(imageEQ);
```



d) Explain why the two histograms (of the original image and of the enhanced image) are different.

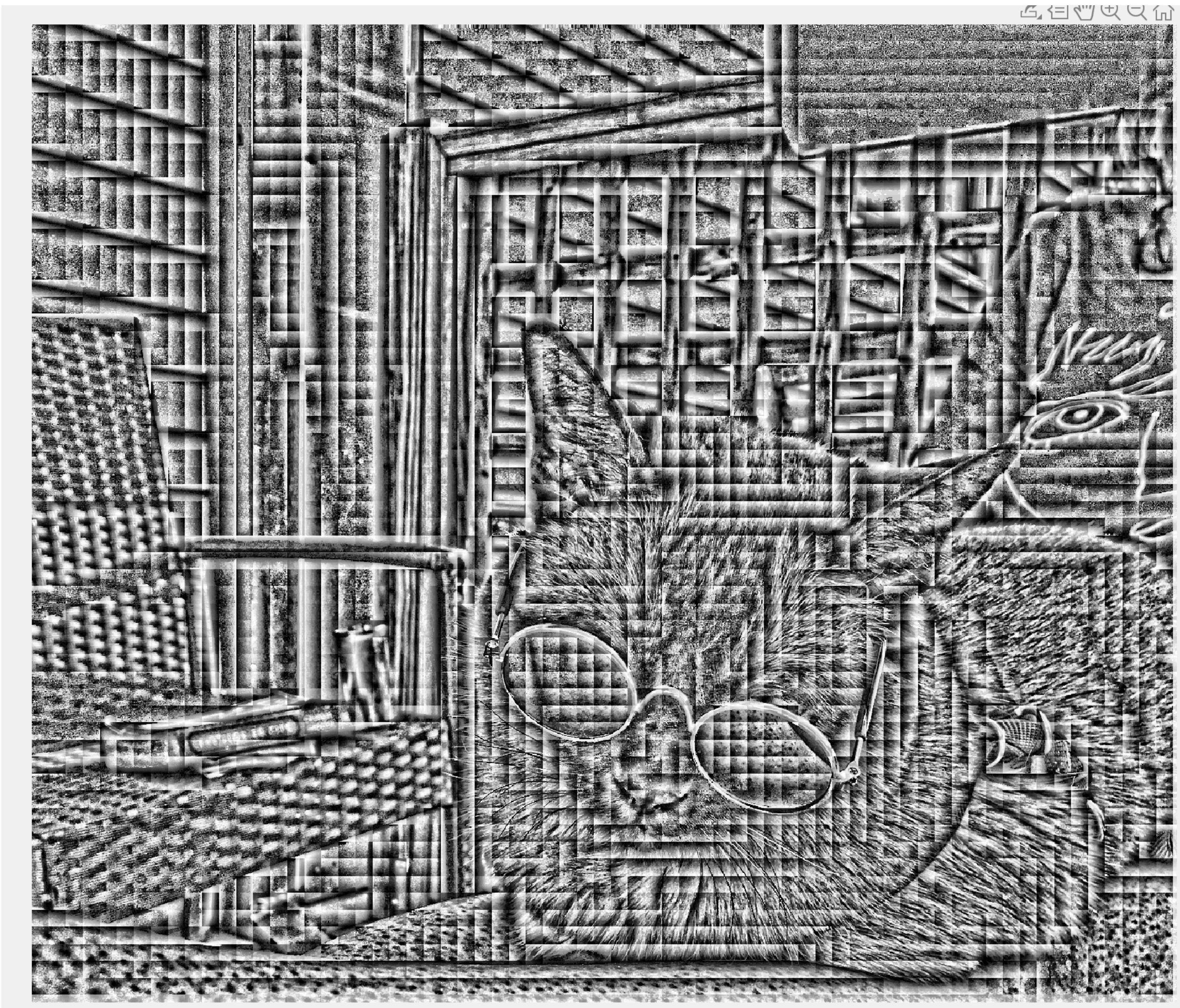
   Bottom histogram went through histogram equalization, the equalization increased the contrast of the image by evenly distributing the pixels across the histogram. The total number of pixels didn't change within the image, they were just grouped together to flatten out the histogram. The original image histogram had 6 peaks they were tightly grouped together, while the EQ histogram, has spread these pixel across the histogram.

481 Students: ***(10/0)*** Apply a local enhancement approach on this image and show your results. Before you start, consider how your image might call for a particular window size. For fun, you might want to try a few different window sizes.

Picture size is 2585 x 3024

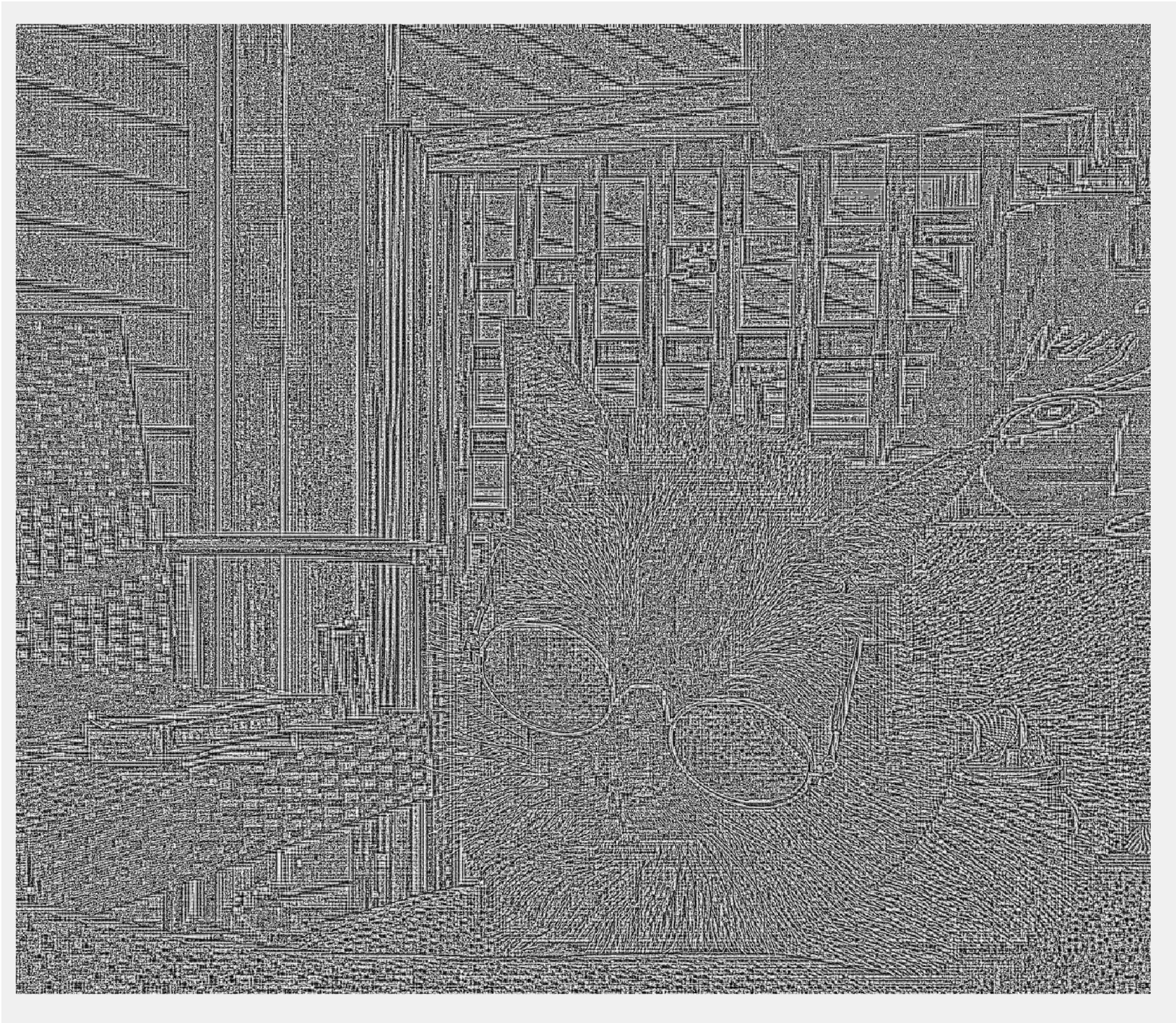
```
%Local Enhancement
%fun can be a function_handle created using @.
%This example uses blkproc to compute the 2-D DCT of each 8-by-8 block to
%the standard deviation of the elements in that block
fun = @histeq;
localEn = blkproc(Y,[45 45],fun);
imshow(localEn);
```

45 x 45 window size. (Ran in 2 seconds, most likely due to large window size)
Tried running 3x3 and matlab was taking too long, so assume a larger window would be required for my image.

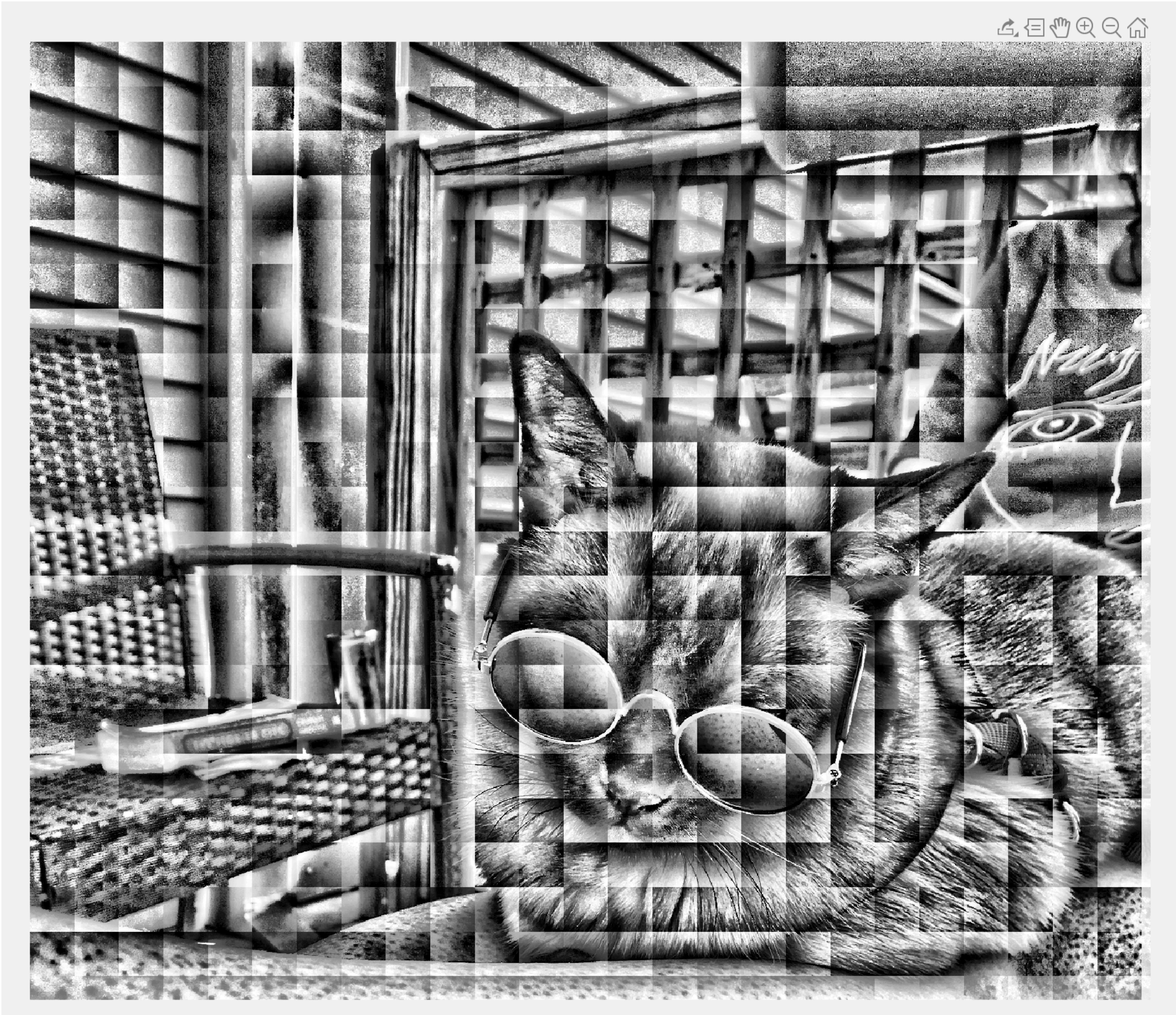


9x9 Window (Took 4 mins to run)

```
fun = @histeq;
localEn = blkproc(Y,[9 9],fun);
imshow(localEn);
```



120 x 120 window (Took a second)



Histogram-based segmentation ***(30/50)***

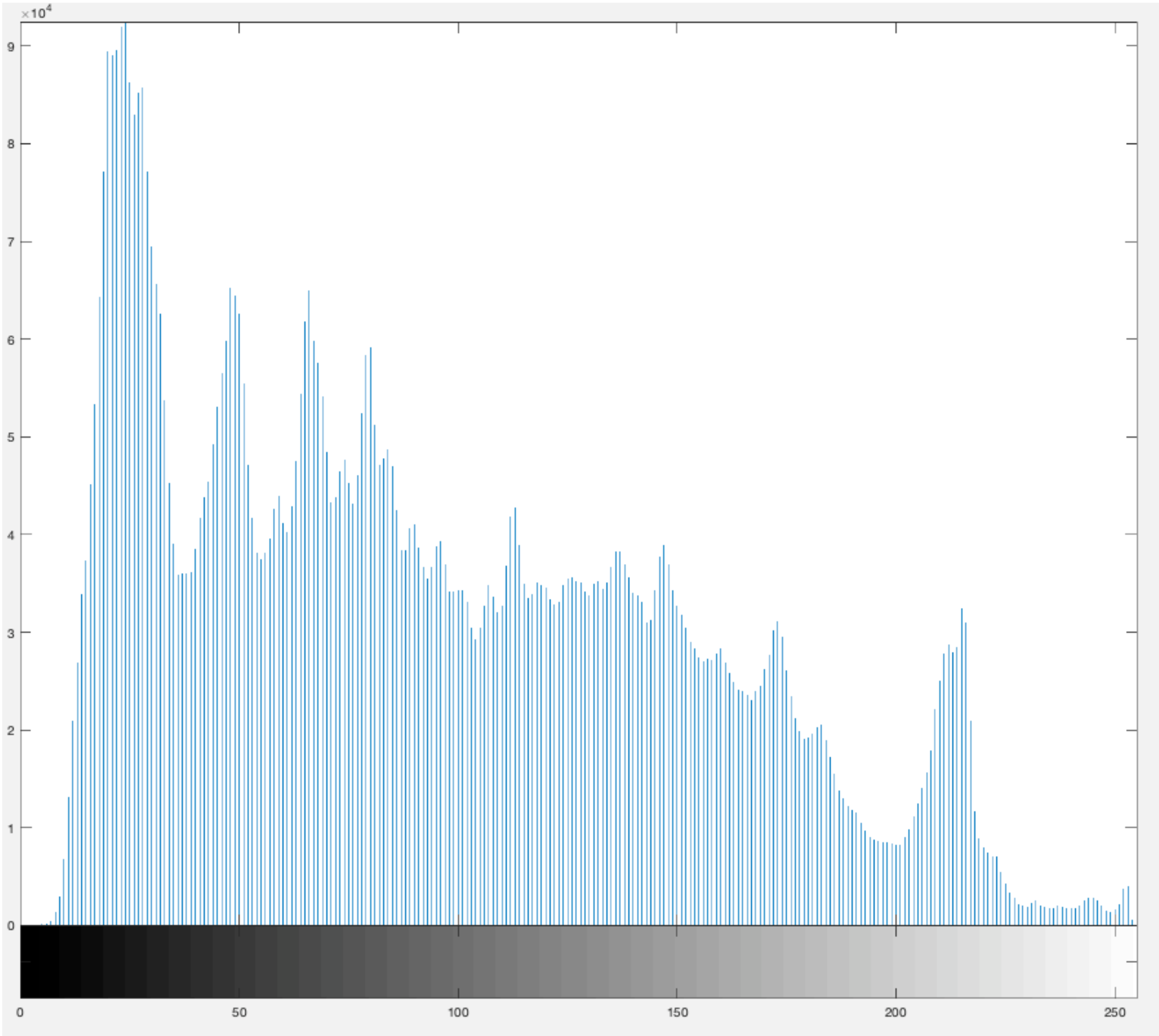
Implement histogram based segmentation on your image as follows:

a) Show your image.

```
% Homework 3 Sebastian Zdarowski
X = imread('collins3.jpg');
Y = rgb2gray(X);
imshow(Y);
```

b) Display the histogram and identify the peaks of your histogram with the “objects” that they correspond to.

```
imhist(Y);
```



Seems that there 8 main peaks within the image. The largest peak belongs to the chair in the back and probably some of the fur on Collins head. Second Peak would be probably be lighter fur but still dark around the glasses. The last peak observed in the histogram, would belong to the cats cream fur.

c) Specify the ranges that you will use to identify the binary objects.

0 to 36 I would use if I wanted to identify the black chair in the background image

200 to 255, If I wanted to identify the house paneling

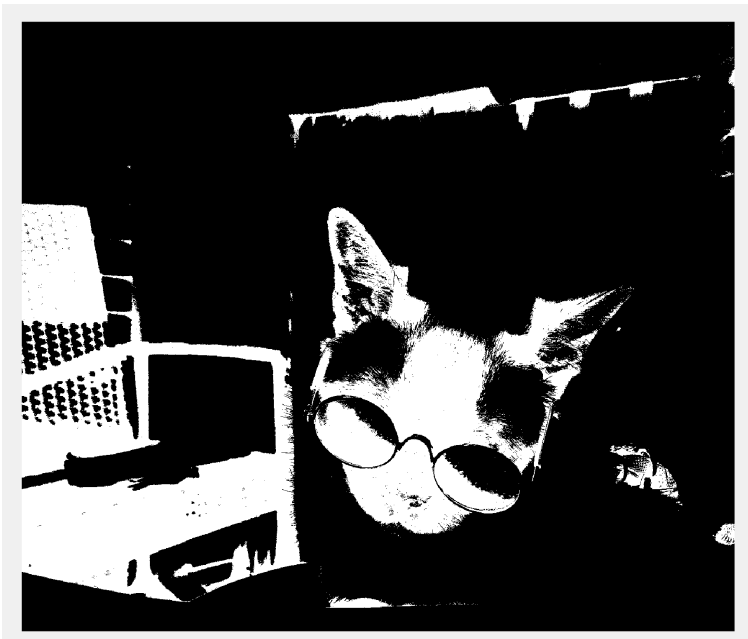
40 to 50 If I wanted to identify the Top of Collins head, the black patches of fur on his head.

d) Show the identified objects as binary images for each range. (Remember to scale the images for display so that objects can be seen.)

```
binary = zeros(size(Y));
```

```
binary(Y<36) = 1;
imshow(binary);
```

0 to 36 I would use if I wanted to identify the black chair in the background image

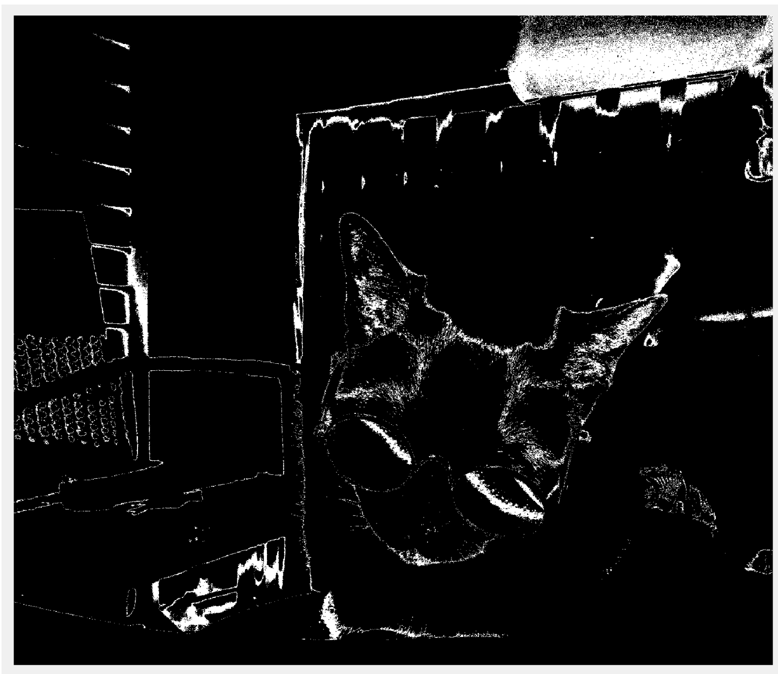


```
binary = zeros(size(Y));
binary(Y>200 & Y<255) = 1;
imshow(binary);
```

identify house paneling

Was used to identify the top of the cats head, but it seams I was wrong in that assumption, it picked up on the outline of the cats head,

```
binary = zeros(size(Y));
binary(Y>40 & Y<50) = 1;
imshow(binary);
```



e) Finally construct the histogram-based segmented image, by combining the binary images. What gets constructed after only adding 3 of the histogram peaks….

```
% Construct the histogram image based segmented image
mean = sum(sum(binary.*double(Y)))/sum(sum(binary));
mean2 = sum(sum(binary2.*double(Y)))/sum(sum(binary2));
mean3 = sum(sum(binary3.*double(Y)))/sum(sum(binary3));

histogramImage = mean*binary + mean2*binary2 + mean3*binary3;
imshow(histogramImage);
```

Tried capturing 6 segments of the histogram, image was able to pick up on more detail within this image. Seems that the additional peaks consisted of the t shirt in the back ground, small patches of fur on the back side of the cat.

481 Students: Histogram equalization, interpolation and noise reduction ***(25/0)***

a) On the histogram equalized image from the first problem, use the `imnoise` function to generate two noise corrupted images as follows:

```
noise_gaussian =imnoise(Image,'gaussian',0,0.05);
noise_saltAndpepper=imnoise(Image,'salt & pepper', 0.02);
```

```
imshow(noise_gaussian);
```

```
imshow(noise_saltAndpepper);
```

b) Use `subplot` to display the original image, the histogram enhanced image, and the two noise corrupted images.

```
subplot(2,2,1);
imshow(Y);
subplot(2,2,2);
imshow(imageEQ);
subplot(2,2,3);
imshow(noise_gaussian);
subplot(2,2,4);
imshow(noise_saltAndpepper);
```

c) Use the function `fspecial` to design averaging filters of size (3x3), (5,5), and (7x7). Use `subplot` to display the `noise_saltAndpepper` image and the three averaged filtered results. Do the same for the `noise_gaussian` image.

```
i3 = fspecial('average',[3 3]);
i5 = fspecial('average',[5 5]);
i7 = fspecial('average',[7 7]);

image1 = imfilter(noise_saltAndpepper,i3);
image2 = imfilter(noise_saltAndpepper,i5);
image3 = imfilter(noise_saltAndpepper,i7);


subplot(2,2,1);
imshow(noise_saltAndpepper);
subplot(2,2,2);
imshow(image1);
subplot(2,2,3);
imshow(image2);
subplot(2,2,4);
imshow(image3);
```

```
i3 = fspecial('average',[3 3]);
i5 = fspecial('average',[5 5]);
i7 = fspecial('average',[7 7]);

image1 = imfilter(noise_gaussian,i3);
image2 = imfilter(noise_gaussian,i5);
image3 = imfilter(noise_gaussian,i7);


subplot(2,2,1);
imshow(noise_gaussian);
subplot(2,2,2);
imshow(image1);
subplot(2,2,3);
imshow(image2);
subplot(2,2,4);
imshow(image3);
```

d) Use the `medfilt2` function to perform median filtering on the `noise_saltAndpepper` image. Design the median filters to work with window sizes of (3x3), (5x5), and (7x7). Use your filters on the `noise_gaussian` image also and display as in part c).

```
%%%%% Median Filter
image1 = medfilt2(noise_gaussian,[3 3]);
image1 = medfilt2(noise_gaussian,[5 5]);
image1 = medfilt2(noise_gaussian,[7 7]);


subplot(2,2,1);
imshow(noise_gaussian);
subplot(2,2,2);
imshow(image1);
subplot(2,2,3);
imshow(image2);
subplot(2,2,4);
```

```
imshow(image3);
```

```
%%%%% Median Filter
image1 = medfilt2(noise_saltAndpepper,[3 3]);
image1 = medfilt2(noise_saltAndpepper,[5 5]);
image1 = medfilt2(noise_saltAndpepper,[7 7]);


subplot(2,2,1);
imshow(noise_saltAndpepper);
subplot(2,2,2);
imshow(image1);
subplot(2,2,3);
imshow(image2);
subplot(2,2,4);
```

```
imshow(image3);
```

General submission instructions:

(a) Be kind to your aging, over-worked professor and submit only a single document. This can be pdf, MS Word, OpenOffice, etc. Do not submit a zip file.

(b) Your single document should include the input image for your problem, if required, and answers to each of the sub-problems (text, image or both, as appropriate). Your document should also include code that you wrote to generate your answers.

(c) You may use any images you like for the programming; I encourage you to use images that might be useful/interesting for your final project.

(d) Feel free to use whatever functions MatLab supplies. Also feel free to write your own, if you are so inclined; it will take more time, but you will gain a deeper understanding of the material.

(e) Point values for each question are indicated as ***(x/y)*** in which ***x*** is the point value for 481 students and ***y*** is the point value for 381 students.