EE-569
INTRODUCTION TO DIGITAL IMAGE PROCESSING

NAME: Harikrishna Prabhu
Email ID: hprabhuusc.edu
Issue Date: 12th January 2018
Due Date: 4th February 2018

Problem 1: Basic Image Manipulation

In this problem, conduct a series of simple manipulations on Greyscale and Color images to get familiarize with accessing and processing of the given image data and producing desired output.

(a)Color Space Transformation

1) Color -> Greyscale transformation:

Convert the given color image (Tiffany.raw) to its greyscale image using the three techniques given below.

(i) The Lightness method
(ii) The Average method
(iii) The Luminosity method

Show the three greyscale images and discuss which method works out the best overall.

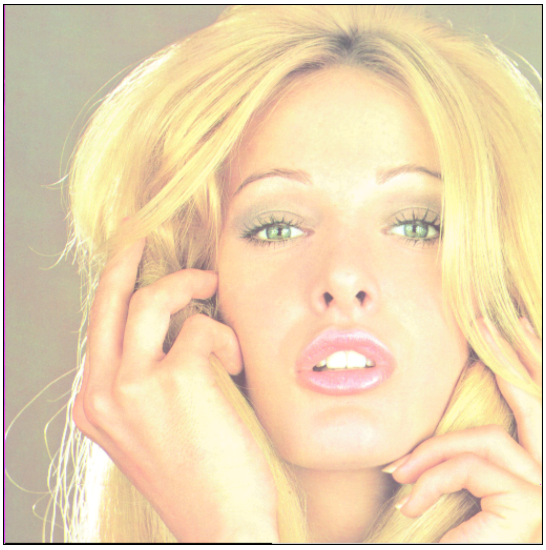


Figure 1: Tiffany.raw

2) CMY(K) Color Space- (Cyan Magenta Yellow(Black))

Represent the given test color images (Dance.raw and Bear.raw) in CMY, for each input color produce three grey scale images corresponding to Cyan, Magenta and Yellow channels.

Figure 2: Dance.raw

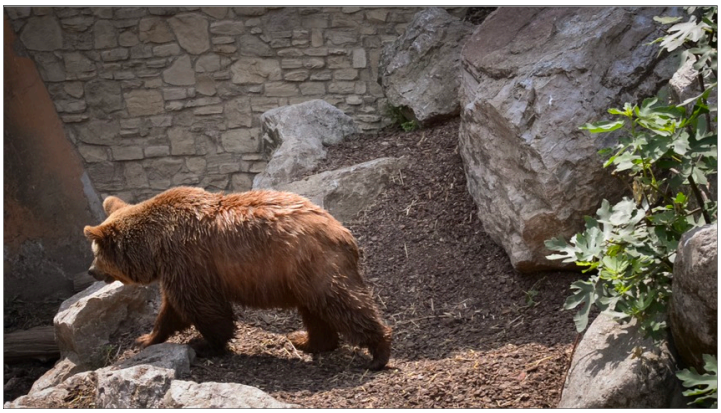


Figure 3: Bear.raw

(b) Image Resizing via Bilinear Interpolation

Use Bilinear-Interpolation to resize the input color image (airplane.raw) of size 512x512 to an output image of size 650x650.

Figure 4: airplane.raw

ABSTRACT AND MOTIVATION:

Color space can be defined as the organization of colors in a specific way or color model, may it be for analog or digital representation. Some of the common color spaces are RGB, CMY(K), Y'CbCr, HSV etc. In general, the rule of thumb is that RGB is used in web, digital display purposes and CMY(K) is used in printing purposes. There are various transformation techniques to convert one color space to another. In a digital Image, each color is represented by three channels of its color space and each channel can be viewed as a grey scale image. A Grey Scale image stores information on the intensity of each channel in the image. Most of the Image processing tools and applications work on grey scale images.

Why Grey scale?

There are so many reasons to as why grey scale images are used.

(i) Though it is counterintuitive to imagine an image in grey scale, from the perspective of the problem, it is the easiest way to visualize it.

(ii) Complexity of coding: It's easier to code for greyscale first and then color than to directly code the project for color.

(iii) When grey scale images are used for processing, the computation speed is very much enhanced

(iv) Color images tend to be deceptive, i.e. from the bird's eye view it's difficult to differentiate different shades/tones of a color.

Fun Fact: Digital Image processing was first applied to a Grey-Scale Image.

Where and Why is CMY(K) used?

CMY(K) is a color model especially used in applications that revolve around printing purposes. This is because of its inherent nature, that is subtractive. (while RGB is Additive in nature). When it comes to printing, RGB colors are dark to start with and mixing these as inks only forms more darker colors. But in case of CMY(K), If Cyan (Green-Blue) is mixed with Yellow (Green-Red) then Green is the result as it is common to both. That is, CMY(K) can be used to get even lighter shades than RGB.

Image Resizing: Often or almost in all cases we require resizing a given image into application specific or system specific size. Say for instance, you click a photograph using your camera and want to use that picture as your PC's wallpaper. There, Image resizing plays its part to quench the need. Image resizing can mean enlargement and shrinkage.

APPROACH AND PROCEDURE:

(a)Color Space Transformation

a 1) Color -> Greyscale transformation:

Here we convert the given color image in RGB to a greyscale image which encompasses all three-color channels in one way or another. We use 3 different techniques to do so. They are,

(i) Lightness method:

This method averages the most prominent and least prominent of the colors.

Formula used is grey_Scale = (max (R, G, B) + min (R, G, B))/2

(ii) Average method:

This method takes the average of all three-color channels.

Formula used is grey_Scale = (R + G + B)/3

(iii) Luminosity method:

It is a more complex and detail specific version of the average method. Here we use the weighted formula: grey_Scale = 0.21 R + 0.72 G + 0.07 B

The first part is to read the given image and store it as an array so that it becomes easy to access each pixel. Here for accessing the image we use iterative method with three variables, where the first two (Width (j=) x Height (i=)) denotes the pixel location and the third one denotes the color channel ((k=) 0-RED, 1-GREEN and 2-BLUE). Using the above three formulas we produce the grey scale images for the given color image.

a 2) CMY(K) Color Space:

Here an input color image represented in RGB is converted to an output color image in CMY(K) color space. This is done using the formula,

C = 255 – R or C = 1 – R’

M = 255 – G or M = 1 – G’

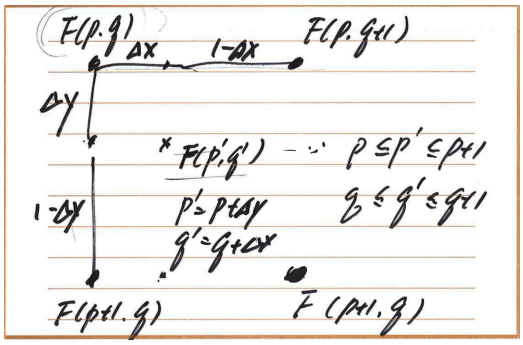
Y = 255 – B or Y = 1 – B’

Where R’, G’ and B’ are R/255, G/255, B/255 respectively.

Initially the input image is read into an array, then second of all it is standardized i,e 0…255 is converted to 0…1. Then using the above-mentioned formula RGB is converted to CMY and each channel of CMY is displayed as greyscale.

(b) Image Resizing via Bilinear Interpolation

Image resizing can be implemented using various methods, namely
(i) Nearest Neighbor
(ii) Bilinear Interpolation
(iii) Bicubic Interpolation
Here we will be implementing image resizing using Bilinear Interpolation. Mathematically Bilinear Interpolation is an extension of linear interpolation for 2 variables.
First of all, before resizing, we know by what ratio or by how much are we resizing the given image. Here we are resizing a 512x512 image into 650x650 image. (ratio= 650/512 = 1.26) i.e the new image is 1.26 times the original image.



As this image explains, we read the pixel from the original image and then based on the ratio we manipulate and save it as output file. When it comes to boundaries, we mirror the rows and columns respectively and then manipulate it on the basis of ratio to relocate. We see that in most of the cases, the pixel locations are not conveniently placed (i.e not integer values) so that manipulation is done easier (eg. 256x256 to 512x512 or 1024x1024 is easy as we just are doubling or quadrupling it) here in case of 512x512 to 650x650, we don't get all positions as integers, so we use the 4 nearest neighbors to allocate value to that pixel.

EXPERIMENTAL RESULT:

a 1) Color -> Greyscale transformation:

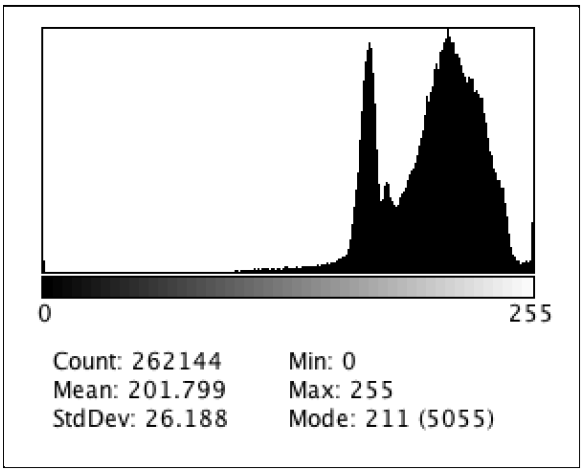
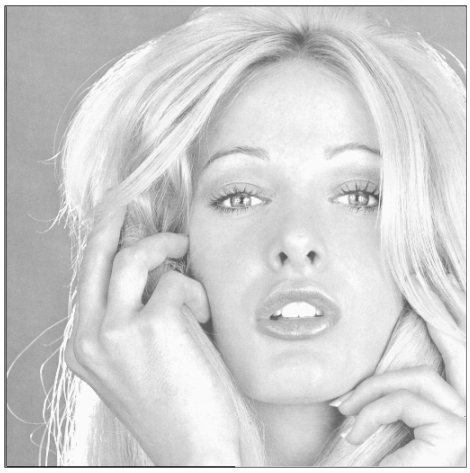


Figure: AVG



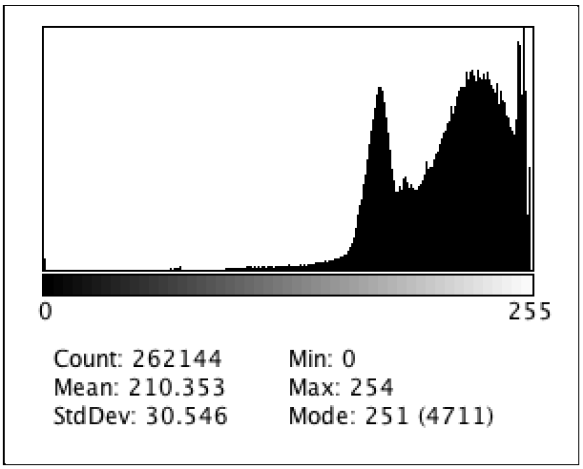


FIGURE: Luminous

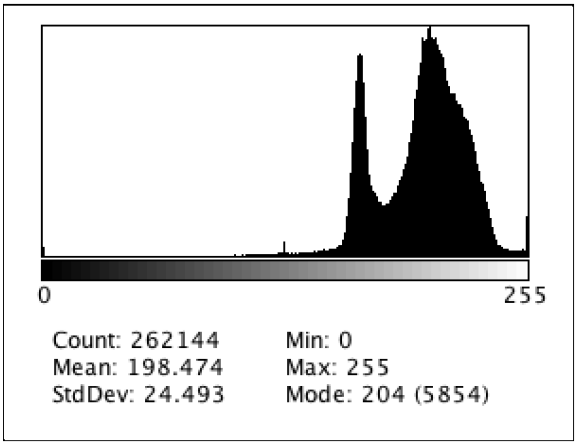
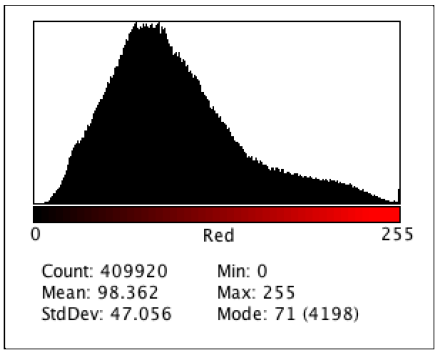
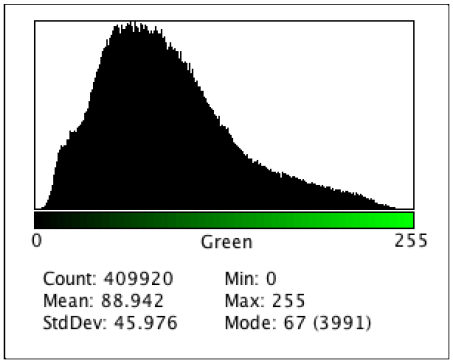


FIGURE: Light

a 2) CMY(K) Color Space:





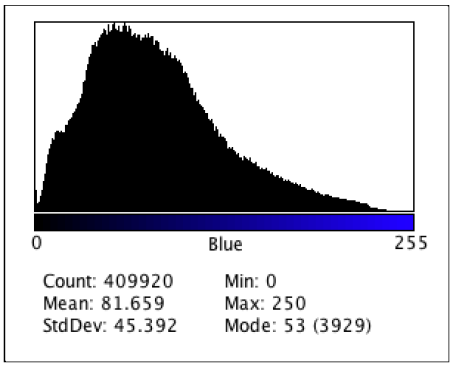


Figure: Bear and its RGB component

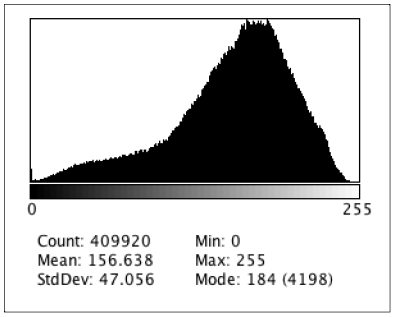


Figure: Bear Cyan and its Histogram

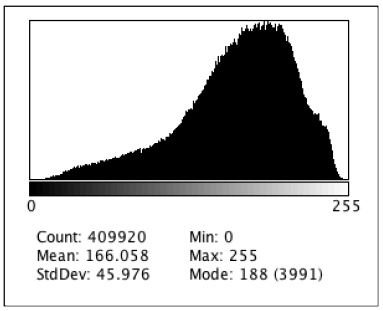


Figure: Bear Magenta and its Histogram

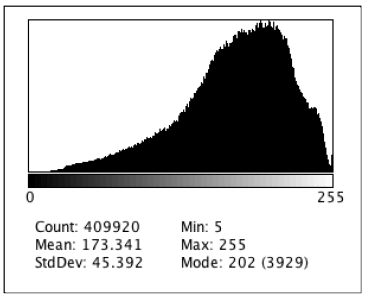
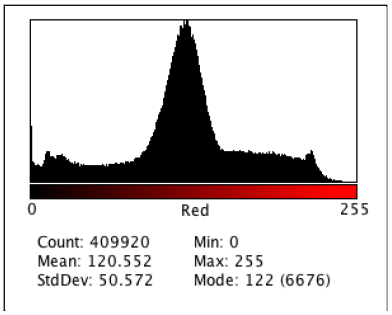
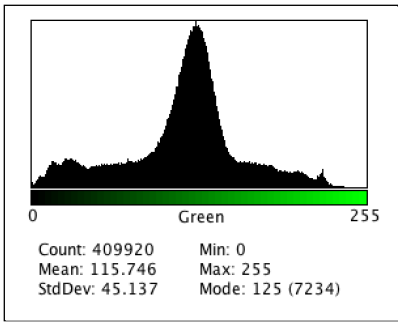


Figure: Bear Yellow and its Histogram





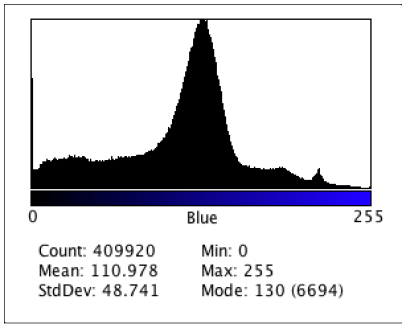
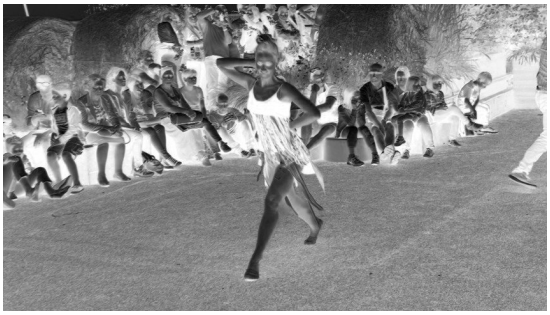


Figure Dance and its RGB components



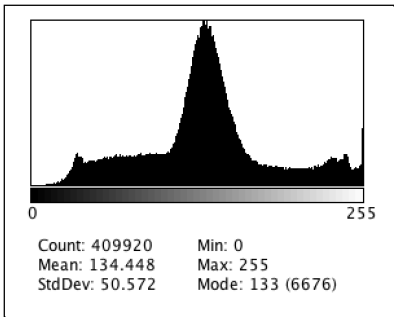
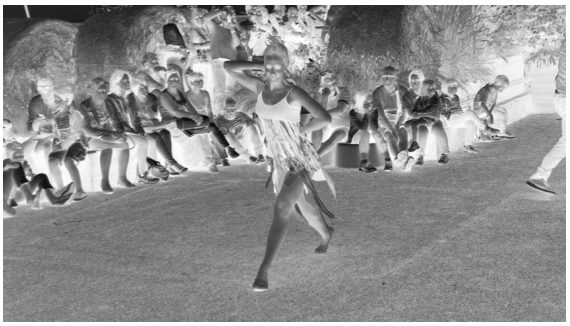


Figure: Dance Cyan and its Histogram



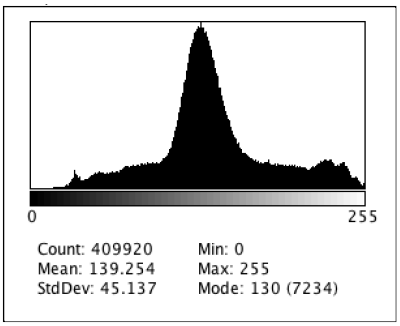


Figure: Dance Magenta and its Histogram

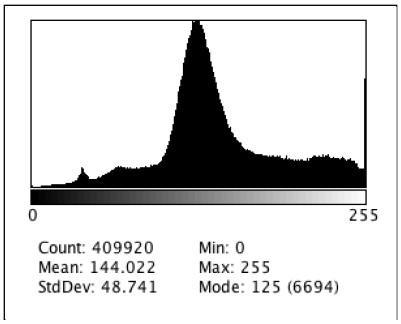


Figure: Dance Yellow and its Histogram

(b) Image Resizing via Bilinear Interpolation

Figure: Airplane_Original 512x512

Figure: Airplane_Resized 650x650

DISCUSSION:

(a)Color Space Transformation

a 1) Color -> Greyscale transformation:

(ii) Averaging Method:

This method is the most common conversion techniques of all. It is very fast and simple because it just averages the three-color channels. However fast and simple it is, it has bad representation of the colors.

(i) Lightness method:

This method is better than the averaging method as it takes into account the prominent of the colors. It looks smoothened and dull than the image represented using the averaging method but there may be a case it gets over-smoothened.

(iii) Luminous method:

This method is little complex than the other two methods that are mentioned here. This method is based in the relative color perception of the human eye. It is a fact that the human eye perceives green stronger than red and red stronger than value.

G>R>B

From the above three color->grey scale conversion techniques, luminosity method is the best of the lot. Its representation is more vivid and clear.

a 2) CMY(K) Color Space:

Even once the Image is converted to CMY(K) from RGB, at the display level we can't figure out much tonal difference but when we study the histograms, we can see a significant difference in the structure. Especially when we compare the RGB and CMY(K) histograms of Bear.raw, we can see that in RGB histogram the structure is left-skewed and the CMY(K) histogram is right-skewed. In both Bear.raw and Dance.raw, the RGB and CMY(K) histograms have close to same standard deviation in the pixel distribution but the mean pixel value is different from one another.

One of the major problem with CMY is that, it cannot achieve darker black or true black. Hence black (K) is introduced and added to each channel to achieve darker tones of a color.

(b) Image Resizing via Bilinear Interpolation

We have successfully resized 512x512 to 65x650. But we must keep in mind that as we increase the size of the output image, the quality of image drops in terms of sharpness and intensity. We witness that as the ratio for the output image increases (Upscale) the image gets more pixelated and it doesn't look good. The same goes with downscaling, as we downscale more, the color information might get lost.

Problem 2: Histogram Equalization

a) Histogram equalization:

To implement the two Histogram equalization techniques, namely

(i) Transfer Function based Histogram equalization.

(ii) Cumulative Probability based histogram equalization.

to enhance the contrast of the given image. (Desk.raw)

Discuss the observations on these two enhancement techniques and suggest any idea on improving further.

b) Image Filtering – Creating Oil painting effect:

An exemplary Oil-Painting effect for the barn.raw is shown. In order to achieve this, follow the given steps,

Figure: Barn.raw

(1) Quantize all the colors of the input image, to an image that has only 64 colors (i.e R-4, G-4 and B-4 colors). Sn example for barn.raw is shown in the figure barn64.raw.

Figure: Barn_64.raw

(2) (Filter)Select the most frequent color in its NxN neighborhood, where N ranges from 3 to 11. An example for 5x5 widow for barn.raw is shown in barn_oil.raw

Figure: Barn_oil.raw

So far, we have dealt with 64 color images only. What happens to the filter if 512 color image is provided instead.

Compare the outputs of both the cases and kindly brief your observations.

c) Image Filtering – Creating Film Special effect:

Given are the two images below, original.raw and film.raw.

FIGURE: original.raw

FIGURE: film.raw

Investigate the given images and find the relation between them, i.e. the effect applied on the original image. Study the histograms to see what type of manipulation is done with the image.

ABSTRACT AND MOTIVATION:

Histogram is a graphical representation of pixel intensity or denotes the frequency with which the pixel appears in the given Image. They come handy in spotting background details and noise in general. And Histogram equalization is a method to process images in order to adjust the contrast of an image by adjusting the pixel intensity distribution. The objective of Histogram equalization method is to bring in a linear trend in the cumulative distribution of the pixel intensities. Here we will be discussing about two methods in histogram equalization, namely,
(i) Transfer function based Histogram equalization technique.
(ii) Cumulative Probability based histogram equalization

One important term to learn alongside Histogram is 'Contrast' as it goes hand in hand and comes more often in image enhancements, it is the difference in luminance or color that makes the object distinguishable.

Filtering is a technique to modify or enhance the given image, such as emphasizing certain features or removing certain features. It is one the preliminary things to know in the field of image processing. Filters are always in use or action in Image processing applications. Here to get acclimatize to the field of Image processing, we learn to deal with filters and its impact on the given original image.

(i) Oil painting effect: We are all familiar with oil-painting in the arts section, this is same as that, but we implement it in digital frame. This method, rather than the effects to be added, it gives us an insight on how to deal with windows for filtering and types of effects that can conveniently be added using window. In most practical applications, Images that we get are of very huge size and applying the effect to each and every pixel becomes tedious and may mess it up or unevenly distribute the effect. Hence, we bring the concept of window, where we can generalize the effect adding process to a fixed frame size and through iterative measures we can evenly add the effect to how much ever big images we get. In simple words, irrespective of the image size, windowing helps to ease the complexity in applying the effect evenly out there in the spatial domain, still as usual the process time may increase due to the inherent size of the input image.

(ii) Film Special effect: This part is an extension to the previous part. The motto of this part is to crack supervised, unprecedented Image processing effects and develop an algorithm accordingly. Here based on the given original image and the effect manipulated output image, we figure out what has happened with the image.

APPROACH AND PROCEDURE:

a) Histogram equalization:

The objective is to enhance the contrast of the given image.

(i) Transfer Function based Histogram equalization technique

In this method, the given input image is read and non-physically split in three greyscale channels (R, G, B). From each of these channels, counters are used to keep track on the number of pixels

for each gradient of the color (0..255), i.e to store the pixel intensity values bin-wise. From this generated pdf, corresponding cumulative distribution function's (cdf) are constructed and it is obvious that the cdf values range from 0 to (width x height) as they keep track/count on the number of pixels in the image. This then is proportioned to match a range varying from 0 to 255 bin-wise. This newly created cdf that which has values ranging from 0..255 is assigned to the pixels bin wise respectively.

(ii) Cumulative Probability based Histogram equalization technique:

In this technique, like the previous one the pdf is generated. The main aim of this technique is to equally distribute the pixels. So, it uses an algorithm called bucket-filling where each bucket in the channel has equal number of pixels in them. It is as simple as that. Each bin will have (height x width)/255 number of pixels, so we assign a counter variable to control the filling of pixels in each bin. Once the bin is filled, the counter is re-initialized and the pixels will start filling in the next bin and this action re-iterates until the last bin is filled.

b) Image Filtering – Creating Oil painting effect:

In this oil painting effect, which seems to be quite complex can actually be broken down into parts and cumulatively solved in an easy manner.

First of all, we need to quantize the image, the word 'quantize' refers to restrict the number of tones for each channel to a particular integer value. So here as mentioned in part (i) of the question, we need a 64-color image to be quantized from the original image, i.e each color channel must have only 4 variants in its tone. (R- will have only 4 tones in the picture and so will G and B have). In order to achieve this, we proceeded with the following steps,

(1) Firstly, calculate the total number of pixels present in the image. Then we divide the histogram of the image into 4 bins (that will hold the 4-quantized color). From these two pieces of information we come to know the lower and upper limits of each bin.

(2) Then take the weighted average of the pixel color value for each bin, i.e find the average color of the pixels present in the bin. This can be done using the formula:

quantized pixel value = ((# pixels in each pixel value) x pixel value)/Total # of pixels in the bin

Once the image is quantized, we need to apply filter to it in order to add the oil painting effect. For applying the effect proceed with the following steps,

(1) Fix a window size (say 3x3 or 5x5 or … NxN, where N=odd) (in 3x3 we will have 9 pixels in the window), then for every pixel we access in the actual image, the widow is taken and the most frequently appearing pixel value is assigned to the present location for the output image. Using iteration, each pixel value of the given image is modified with the help of window and stored in the output image file. By replacing the pixel value with the most frequently appearing pixel, mathematics means MODE of the pixels present in the window. And the above same steps are followed for 5x5, 7x7 and 9x9 windows.

c) Image Filtering – Creating Film Special effect:

Here first the given image and the image with the effect is studied and a relation between them is deduced. Based on what type of relation it has, we implement the algorithm accordingly. Here we see that the image is flipped from left to right or about the y-axis, so we assign 0..255 to 255..0, hence the image is flipped. Then we see that the image is inverted, so we subtract the present bin value from 255, therefore whatever was black becomes white and vice versa. Apart from flip and inversion we see that there is difference is histogram structure. The effect added picture has the histogram that is shrinked in size.
This shrinkage can be brought about by ratio multiplication or linear interpolation, but since the shrinked histogram has min value and max value limit, we can formulate the shrinkage as
Present pixel value = ((Present pixel value – Min Pixel value)/(Max pixel value – Min pixel value)) + min pixel value

EXPERIMENTAL RESULT:

a) Histogram equalization:

FIGURE: Desk.raw given image

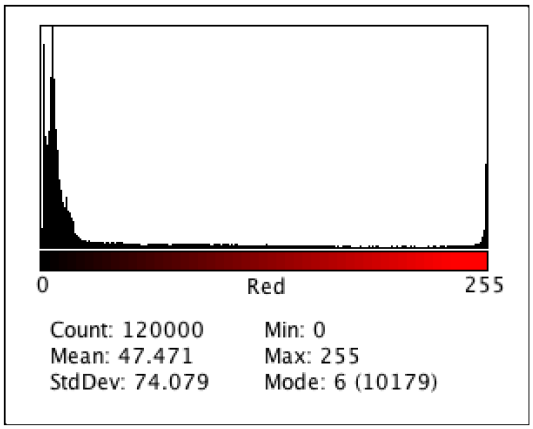


FIGURE: R – Channel histogram of Desk.raw

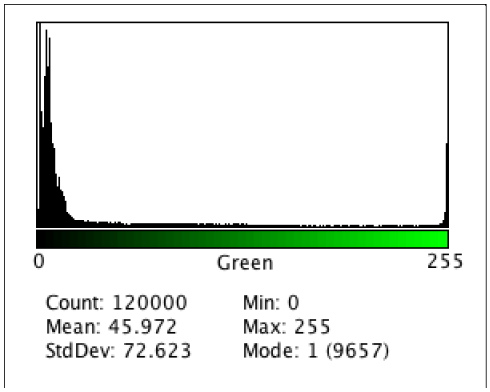


FIGURE: G – Channel histogram of Desk.raw

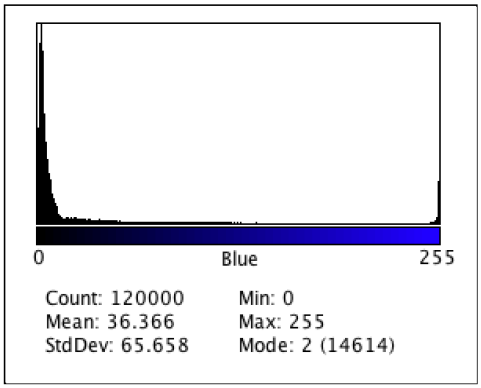
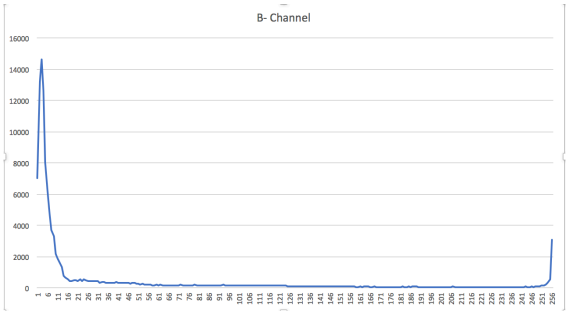
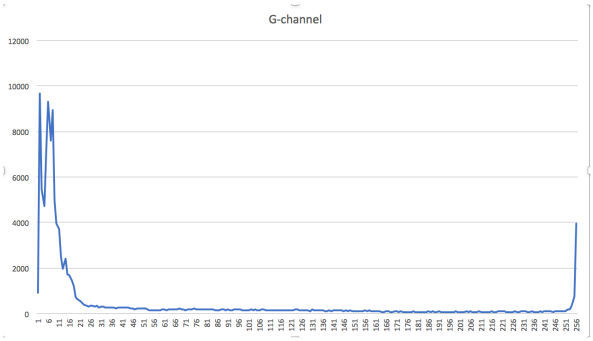


FIGURE: B – Channel histogram of Desk.raw





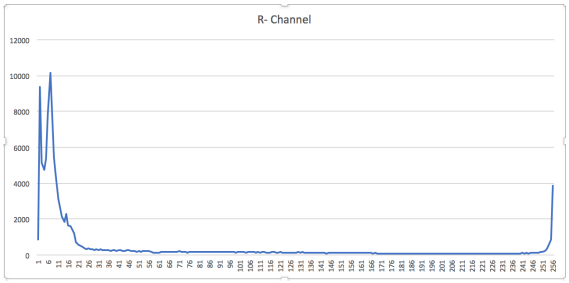
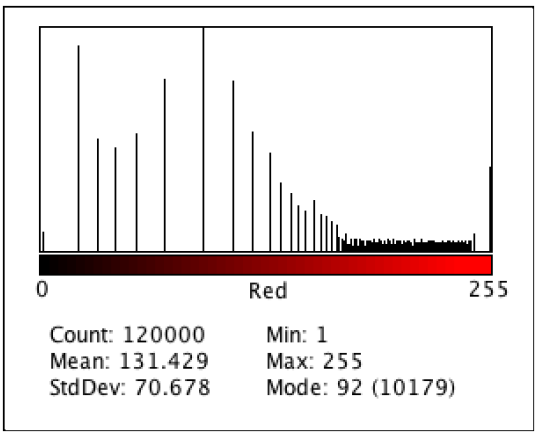


Figure: Histogram: B,G,R.

FIGURE: Image enhanced: Transfer function based histogram equalisation



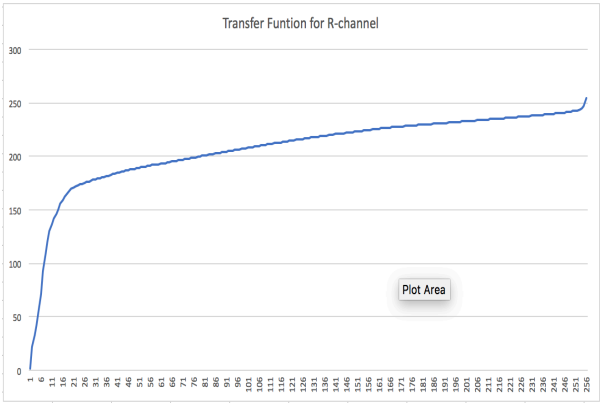
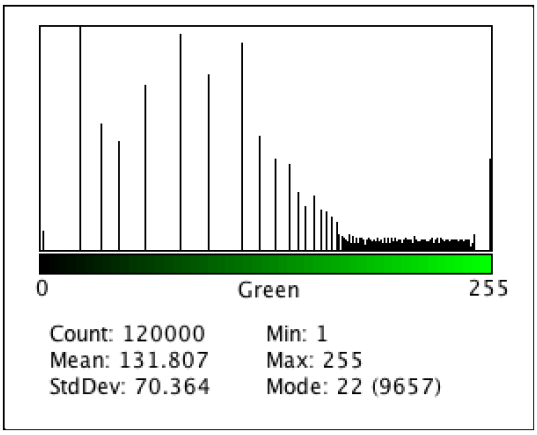


FIGURE: PDF and CDF of channel - R



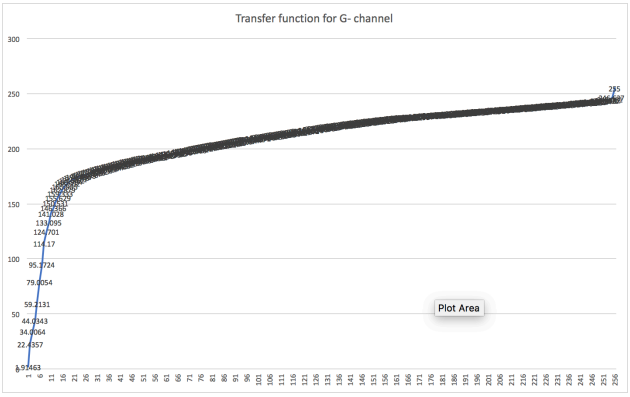
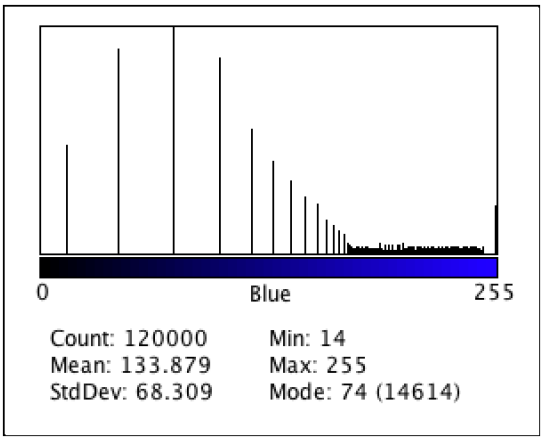


FIGURE: PDF and CDF of channel - G



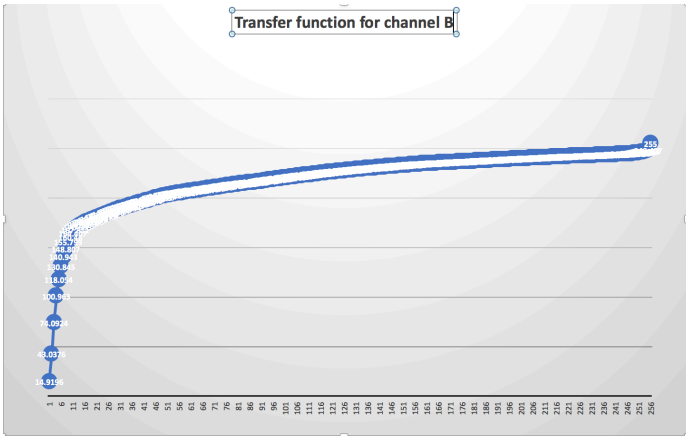


FIGURE: PDF and CDF of channel – B

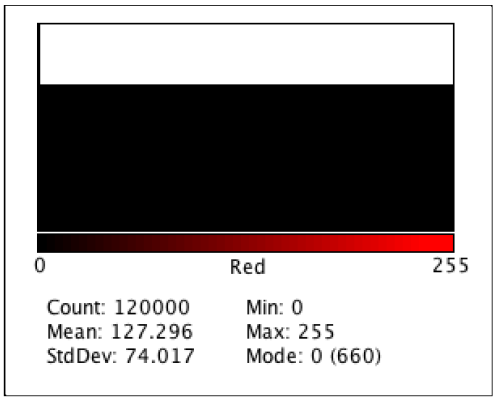
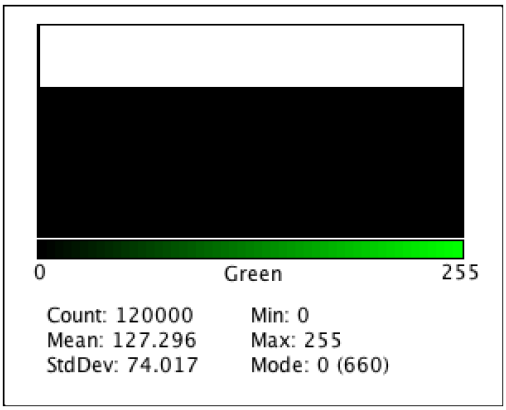


FIGURE: Enhanced Image: Cumulative Probability based histogram equalization, R channel PDF.



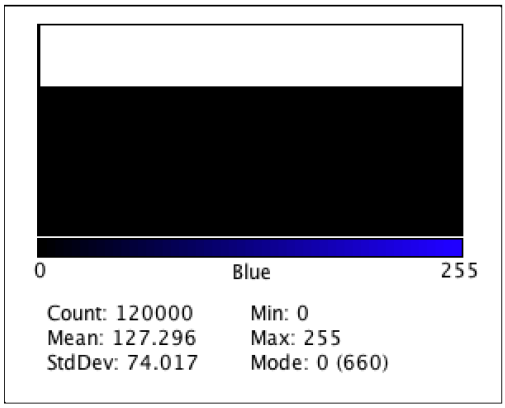


FIGURE: G and B channel PDFs

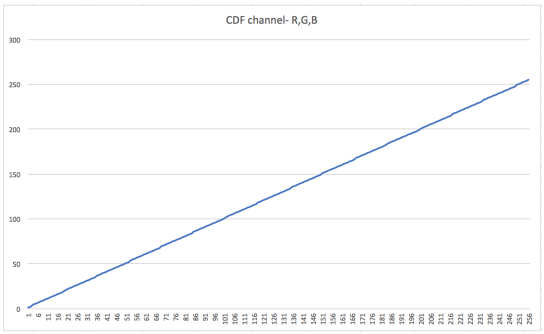


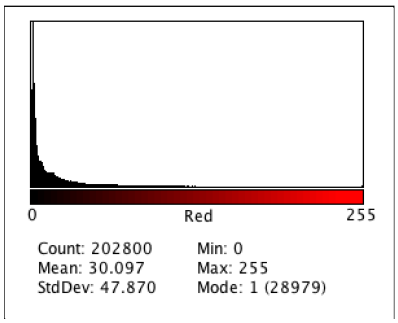
FIGURE: CDF of R channel (same as G and B channel)

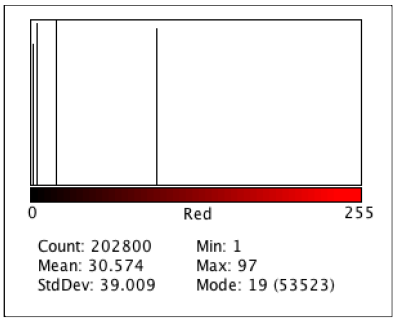
b) Image Filtering – Creating Oil painting effect:

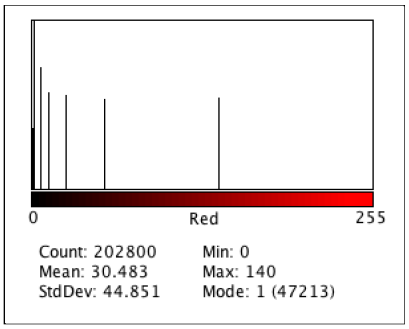
Figure: original

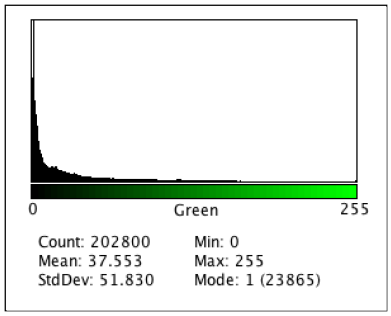
Figure: 64 colors

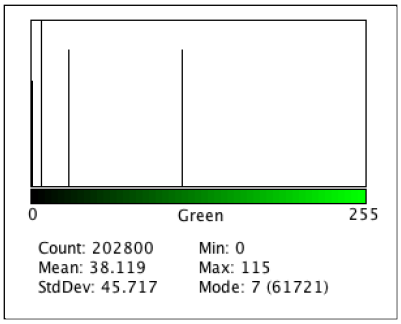
Figure: 512 colors

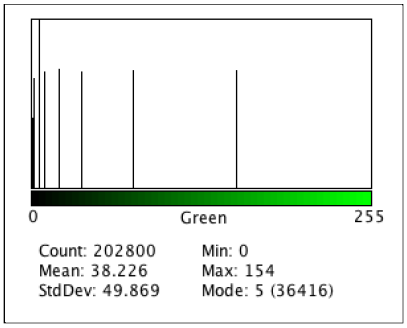

0
Red
255
Count: 202800
Mean: 30.097
StdDev: 47.870
Min: 0
Max: 255
Mode: 1 (28979)

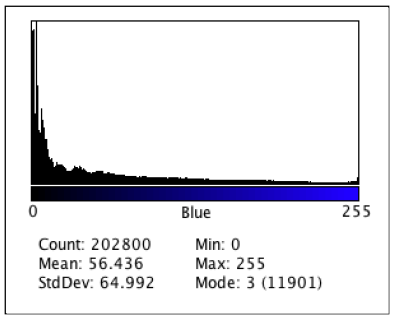

0
Red
255
Count: 202800
Mean: 30.574
StdDev: 39.009
Min: 1
Max: 97
Mode: 19 (53523)

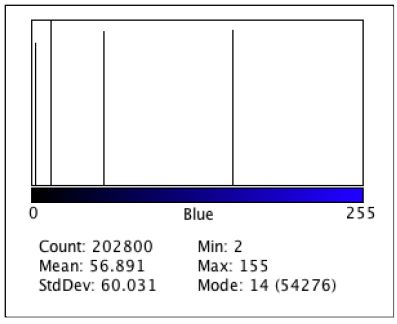

0
Red
255
Count: 202800
Mean: 30.483
StdDev: 44.851
Min: 0
Max: 140
Mode: 1 (47213)


0
Green
255
Count: 202800
Mean: 37.553
StdDev: 51.830
Min: 0
Max: 255
Mode: 1 (23865)


0
Green
255
Count: 202800
Mean: 38.119
StdDev: 45.717
Min: 0
Max: 115
Mode: 7 (61721)


0
Green
255
Count: 202800
Mean: 38.226
StdDev: 49.869
Min: 0
Max: 154
Mode: 5 (36416)


0
Blue
255
Count: 202800
Mean: 56.436
StdDev: 64.992
Min: 0
Max: 255
Mode: 3 (11901)


0
Blue
255
Count: 202800
Mean: 56.891
StdDev: 60.031
Min: 2
Max: 155
Mode: 14 (54276)

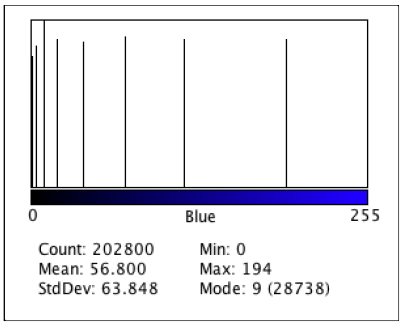

0
Blue
255
Count: 202800
Mean: 56.800
StdDev: 63.848
Min: 0
Max: 194
Mode: 9 (28738)

FIGURE: 3x3

FIGURE 5x5

FIGURE 7x7

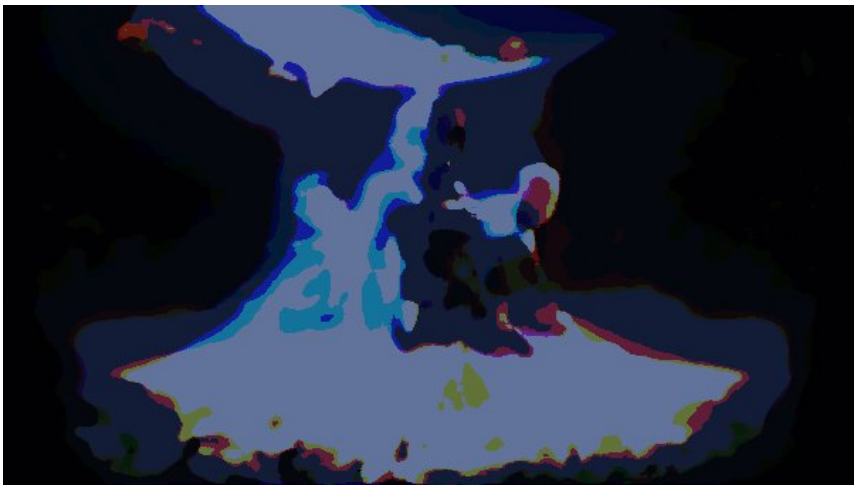


FIGURE 9x9

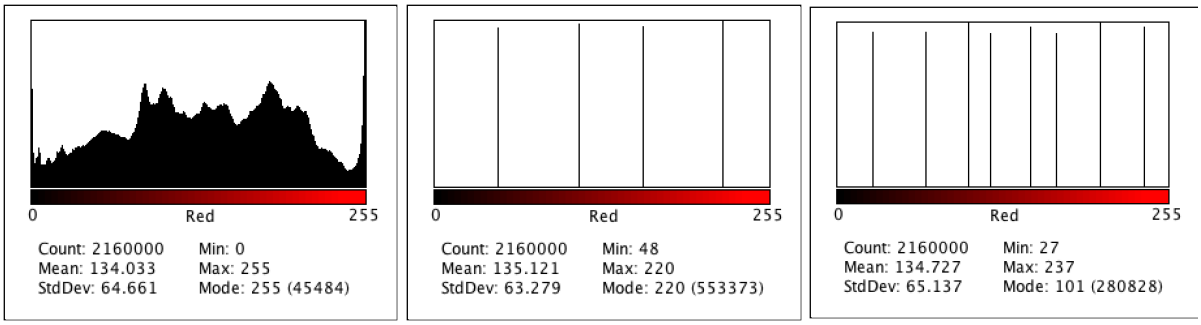
FIGURE 3x3 for 512 colors

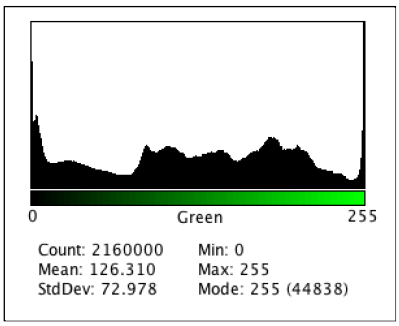
FIGURW 5x5 for 512 colors

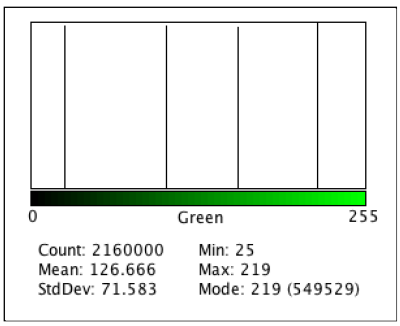
Figure: original

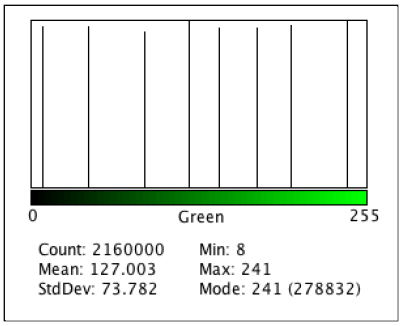
Figure 64

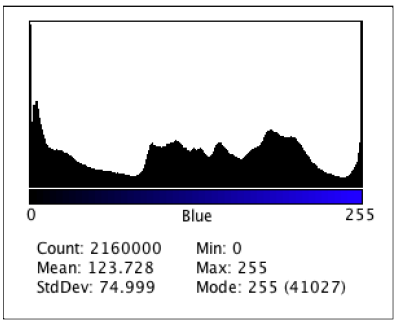
Figure 512

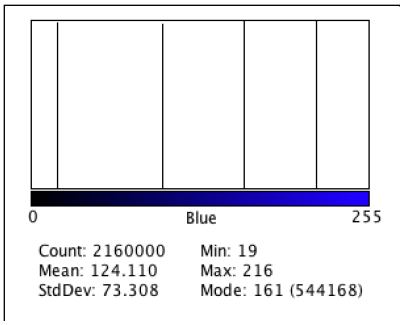












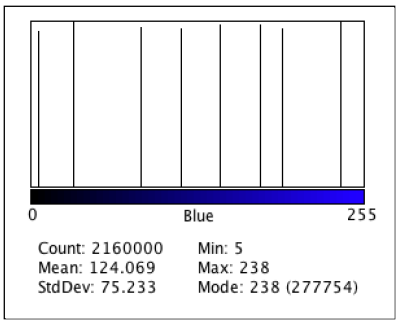


Figure: 3x3 64 colors

Figure: 5x5 64 colors

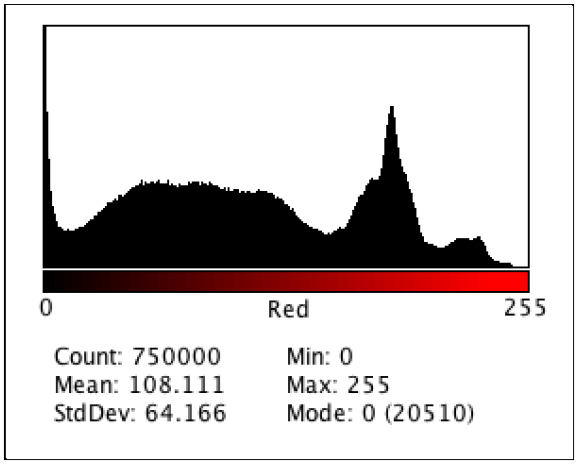
Figure: 7x7 64 colors

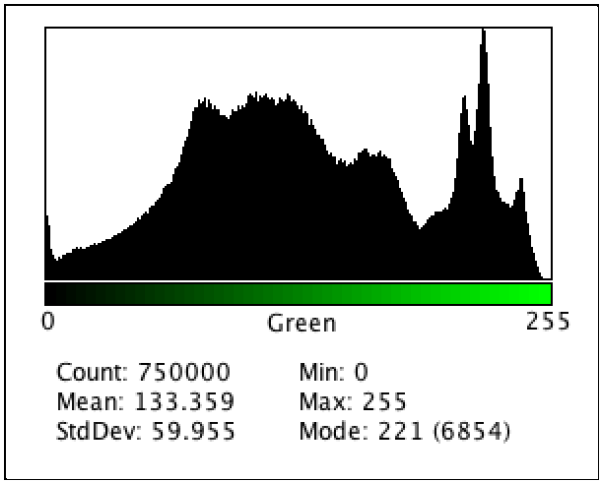
Figure: 9x9 64 colors

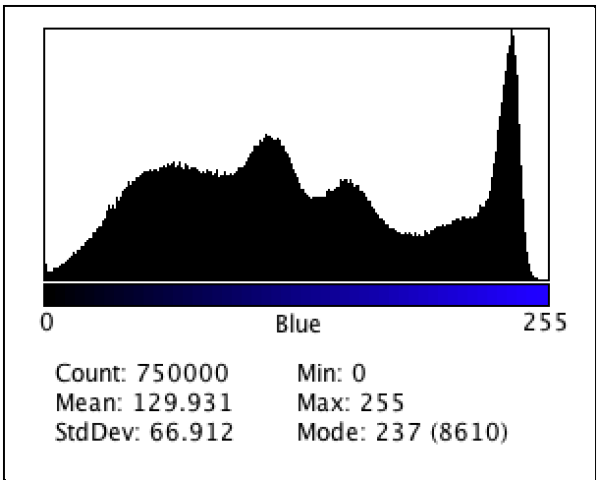

Figure: 3x3 512 colors

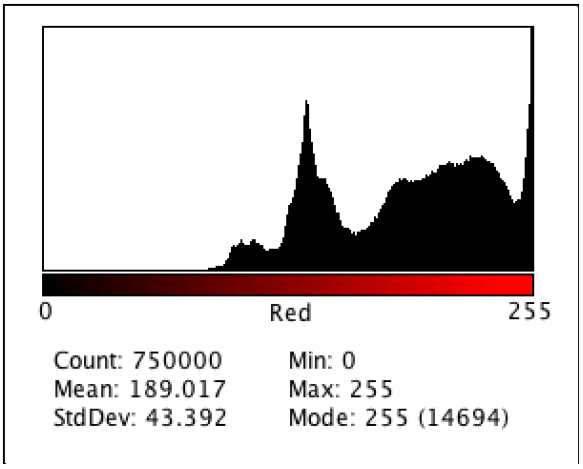

Figure: 5x5 512 colors

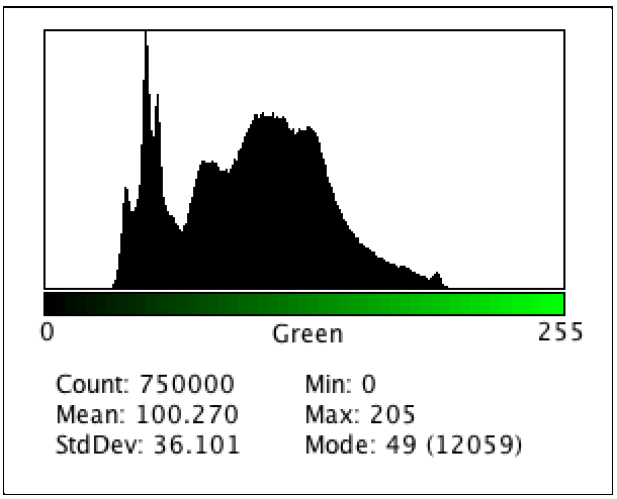
c) Image Filtering – Creating Film Special effect:

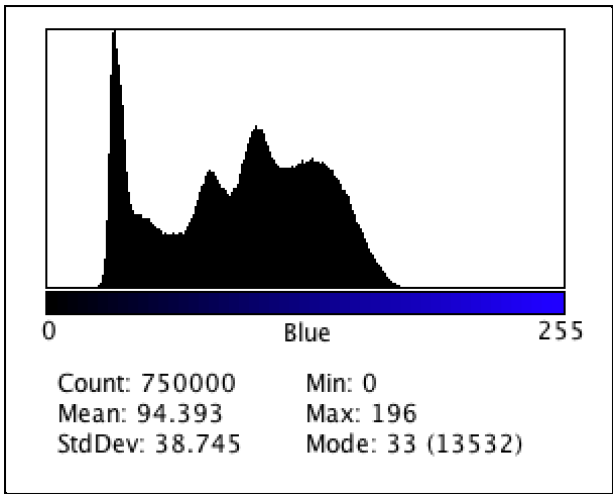


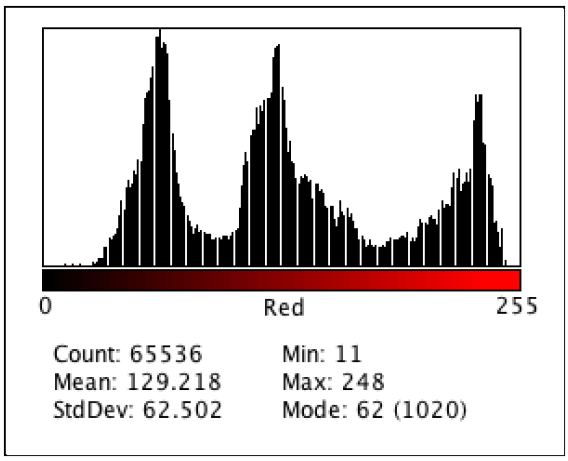

0
Green
255
Count: 750000
Mean: 133.359
StdDev: 59.955
Min: 0
Max: 255
Mode: 221 (6854)

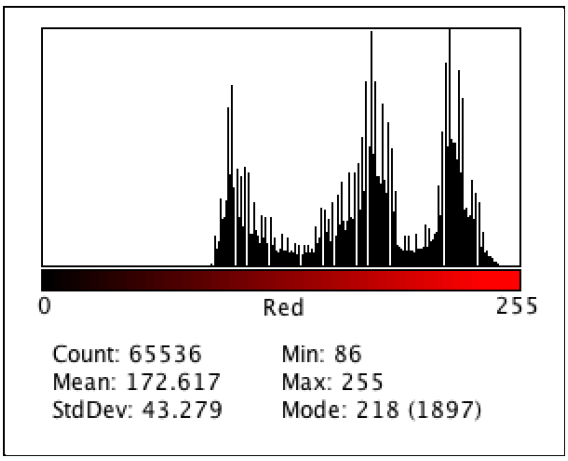

0
Blue
255
Count: 750000
Mean: 129.931
StdDev: 66.912
Min: 0
Max: 255
Mode: 237 (8610)

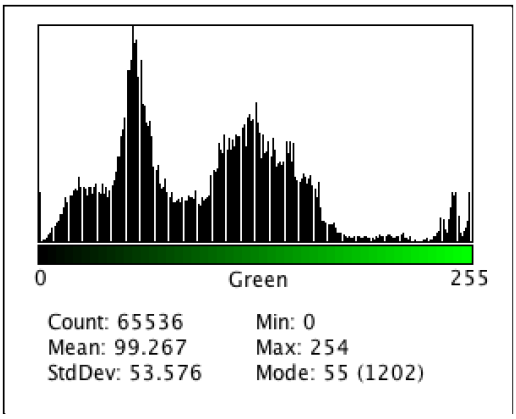

0
Red
255
Count: 750000
Mean: 189.017
StdDev: 43.392
Min: 0
Max: 255
Mode: 255 (14694)

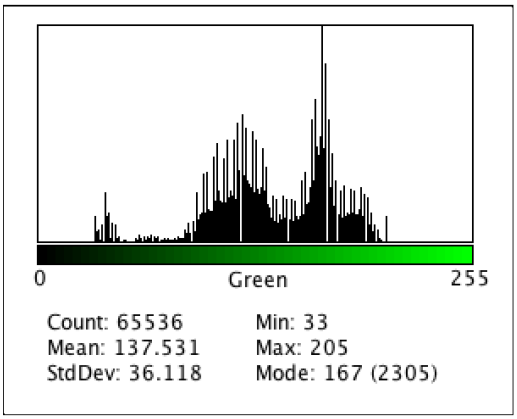

0
Green
255
Count: 750000
Mean: 100.270
StdDev: 36.101
Min: 0
Max: 205
Mode: 49 (12059)

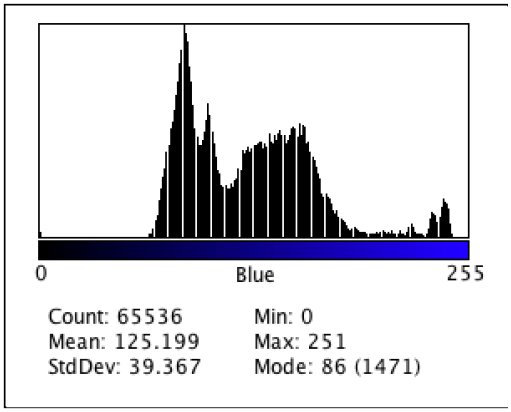

0
Blue
255
Count: 750000
Mean: 94.393
StdDev: 38.745
Min: 0
Max: 196
Mode: 33 (13532)

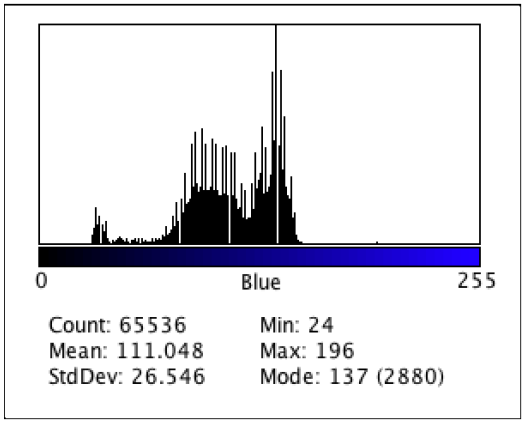

0
Red
255
Count: 65536
Mean: 129.218
StdDev: 62.502
Min: 11
Max: 248
Mode: 62 (1020)


0
Red
255
Count: 65536
Mean: 172.617
StdDev: 43.279
Min: 86
Max: 255
Mode: 218 (1897)









DISCUSSION:

a) Histogram equalization:

From observing the given image and its histograms, we see that this image needs to be enhanced and the histogram is concentratednb at the ends.

(i) Transfer function based Histogram equalization technique:

This method's objective is to make the cdf of the image linear. (As we know that, more linear the cdf is, more better distributed are the pixels and so is the image enhanced better). The output image's histogram is quite better than the input image's. The main thing to notice is the structure of the CDF. It takes a linear fashioned line. When we look at the curve structurally, it increases enormously at the beginning and somehow manages to have a constant slope. If it were to have

a constant slope throughout it's cdf then the image would have been enhanced better. This what we'll see in method (ii).

(ii) Cumulative Probability based Histogram equalization technique:

This method actually produces constant slope cdf, in other words, we can say that the image enhanced using this method is better than the previous method. By analyzing the histograms of the image produced using this enhancement technique we can see that the pixels are perfectly evenly distributed. The slope of the cdf is 1, i.e the perfect linear line.

b) Image Filtering – Creating Oil painting effect:

In this question, when we compare the original image with the 64 and 512 colors image, we can clearly make out the difference in the image fineness. And we studied the differences in them by analizing the histogram structure for each channel seperately. The original image has a range of pixel values where as the 64 and 512 color images only have quantized 4 and 8 pixel values for each channel. This study show the importance of pixel distribution or number of pixels present in the image. Eventhough human eye wouldn't notice difference in the tones of each color, this observation shows that greaterin the variance in the image's pixel value, richer is the picture with colors. The more down-quantize we make the picture, more distinct regions are formed but with loss in the color richness.

When it comes to windows, 3x3 window is the best window to use for filtering as it is handy, robust and effect addition is smooth. As the window size increases, the image becomes more pixelated after application of the effect. This is because as window size increases, the number of pixels that effect the MODE will also increase and greater variance in brought into action, as a result the present pixel maybe inappropriately weighted with the average.

c) Image Filtering – Creating Film Special effect:

As observed and studied using the original.raw and film.raw, the changes that we have made are, (i)flip, (ii)invert and (iii) histogram shrinkage. Similarly, we have done so with the girl.raw to get this film effect. The histogram of the effect added picture says it all. Another inference to be made is that, after adding the film effect we can observe the details in the picture in a different perspective. Many things that weren't noticeable in the color image was vividly detailed in the effect added picture. The shrining of the histogram towards the right-hand side of the plot clearly shows that the image is given a darker tone to all of its colors.

Problem 3: Noise Removal

In this problem, you should implement a number for image denoising algorithms to improve image quality. You can use Peak Signal to Noise Ratio (PSNR) quality metric to assess the denoising algorithm. Calculate/Find the PSNR values for each channel separately.

(a)Mix noise in color image:

Perform noise removal on the given input image shown below

Figure: Actual Image

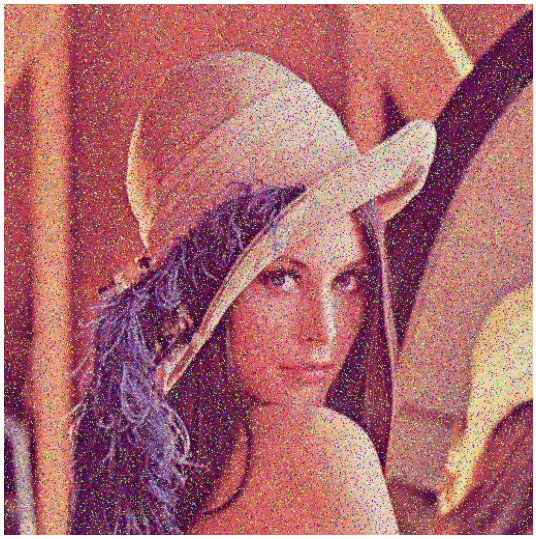


Figure: Image with noise added

1)

(1) Identify the type of noise and check if all the channels have same noise

(2) Should you perform filtering operation on each channel separately or for both noises together.

(3) What filter would you like to use to remove noise.

(4) Can you cascade these filters in any order? Justify it

(5) Describe the effect of different window size in filtering.

2) Try getting the best result in removing the noise

(1) Describe the method.

(2) Discuss its short comings

(3) Give suggestions to improve denoising.

(b)Principle Component Analysis(PCA):

PCA is used as one of the methods used in image denoising.

(1) Explain why PCA is used in filtering approach for noisy data. What do the components mean and how to choose the number of components in image denoising.

(2) Describe and implement patch based local PCA (PLPCA).

(3) Apply PLPCA to the house_noisy.png image with Gaussian noise, sigma=25, and optimize different parameters e.g patch size, # of components etc., and discuss their effect on denoising process. State the best PSNR value and the optimized parameters in your report.

(4) Compare the performance of PCA and that of the filters used in part (a). Explain why?

Figure: House.png

Figure: House_noisy.png

(c) Block Matching and 3-D transform filter:

(i)Explain BM3D algorithm in your own words and implement the BM3D filter to denoise the noisy image.

(ii) Why does the author utilize step 2. What will be the difference in denoising the image with or without using step 2.

(iii) How would you classify BM3D filter? Spatial domain filter or Frequency domain filter or both?

(iv) Compare PCA and BM3D.

ABSTRACT AND MOTIVATION:

Image noise can be defined as the random variation in the brightness or color of the pixel. It is mostly introduced due to electronic noise. Say, capacitive effect, thermal vibrations etc all add up to the noise in digital image.

There are many image denoising algorithms that are used to remove noise from the given image, but before applying denoising techniques one must know to identify the type of noise that is present in the image. A few of them are, (1) Salt and Pepper noise, (2) Gaussian noise, (3) Shot noise, (4) Quantization noise etc.

There is no ad-hoc or instant means to figure out the type of noise, but what we can do is that, we can analyze the histogram of the noisy image. Each type of noise has its characteristics embedded with the histogram structure. For impulse noises, we can find high-frequency spikes then and there on the histogram, while for Gaussian noise the histogram itself appears to be normally distributed. There are cases where we can spot the type of noise present by directly looking into the picture. For instance, in salt & pepper noise we can see dark pixels in regions of

lighter ones and lighter pixels in the regions of darker ones. There are various filters that help in process of noise removal. (1) Mean filter, (2) Median filter, (3) Weiner filter, (4) Linear filter etc. Here we'll deal a couple of them.

PCA or the Principle Component Analysis is a fast and flexible unsupervised for dimensionality reduction in data. It is a frequently used linear- statistical model based filter in the application of noise removal in image processing. PCA mathematically can be defined as an orthogonal linear transform that rotates the data to a new co-ordinate system such that the data with the most variance is present first and so on. Here we will be studying about the patch based PCA. Patch-PCA is a denoising algorithm that follows hard thresholding. The noise is spread all over the image and using PCA we arrange the patches in the decreasing variance order, and here we see that the noise patches are present in the lower corner of the variance matrix and it is truncated to reproduce the new denoised image.

BM3D is an image and video denoising by sparse 3D transform domain filter. It is the state of the art filter that is in application today.

APPROACH AND PROCEDURE:

(a)Mix noise in color image:

Read the given noisy image and segregate data based on for each channel. We know that both salt & pepper noise and Gaussian noise is present.

(i) To remove the salt & pepper noise, we use median filter. Here we use the window technique that we had discussed in the Oil-Painting question. For every pixel from the actual noisy image, we re-assign it with the median of the pixels that are present in the window. 'MEDIAN' the middle term in the ascending ordered pixel values.

(ii) To remove the Gaussian noise, first we have to apply median filter that is discussed above and then mean filter is applied. Here too with the help of windowing, but instead of finding the median of the pixels in the window, we find the mean. For a 3x3 window its summation of all the pixel values /9 and for 5x5 its summation/25.

Once applying filters is done, we need to calculate the PSNR value. PSNR can be calculated as follows,

$$\text{PSNR (dB)} = 10\log_{10}\left(\frac{\text{Max}^2}{\text{MSE}}\right)$$

$$\text{where MSE} = \frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M}\left(Y(i,j)-X(i,j)\right)^2$$

X : Original Noise-free Image of size $N \times M$

Y : Filterd Image of size $N \times M$

Max: Maximum possible pixel intensity = 255

PSNR value can be used as a metric to measure quality of noise removal. Higher is the PSNR value, greater is the denoising effect. But not necessarily in all cases.
Based on the PSNR values that we get, we re-apply the median and mean filters through iterative measures and check the PSNR values produced. With a couple of trial and errors we'll get to know the pattern of image denoising and we can edit the algorithm accordingly.

b) Principal Component analysis:

(1) Image denoising has always had to make a compromise between the reduction of noise and preserving significant amount of information about the image. In most of the filters that are used, only either of the two is achieved. PCA is a pattern identifying algorithm, where it expresses the information of the image by highlighting the differences and similarity in the image information. Data identification and patterning is very difficult to deal with high dimensional image. But with PCA we can analyze the pattern and compress the image or reduce its dimensions without much loss of information pertaining to the pixels. Mathematically, it rotates the data set in spatial dimension in such a way that it withholds most significant data and truncates or can neglect the less significant ones. i.e, the image details are stored leaving behind the noise. Here the number of components mean the number of axis present in the correlation matrix. This axis will give information regarding the total number of components present and the significant number of components to be considered while the rest is truncated during image reconstruction.
(2) & (3)
In PLPCA, the algorithm generates patches from the given input image. These patches are formed such that they form the basic building blocks of the entire image, i.e these patches can be used to reproduce the entire image. These patches include both image data and noise data. This patch based PCA (GLOBAL) compares each patch with every other patch and generates a corresponding data set with the most significant ones at the front. In local patch based PCA, instead of each patch being compared with all the other patches in the image, it is correlated with the patches present in its quadrant (the images are split into quadrants). Then the data set is produced and is projected in its orthogonal coordinate space in order to obtain the principal components with most variance (i.e. the image data). And then after projecting it to its orthogonal space, we can fixate on the number of components to be used in image reconstruction. Here for this Question, we assume the standard deviation to be 25. From the output image generated with the application of PLPCA noise removal technique, we calculate and determine the PSNR value. We can make modifications to the patch size, half size (for the quadrant window) and the number of components to be used to get better PSNR values.
(4) Use these two images (house.raw and house_noise.raw) in part (a) again and evaluate the PSNR value produced using those combination of filters.

(c)BM3D:

BM3D is a two-stage filter. In stage one it acts as a normal linear filter and stage two has a low pass weiner filter that enhances the denoising operation. Significant improvement in the denoising is brought about by the weiner filter that is brought above by the stage 2. The PSNR values obtained through BM3D denoising is found to be of very high value indicating it's richness in denoising.

EXPERIMENTAL RESULT:

(a)Mix noise in color image:

Figure: Lena original

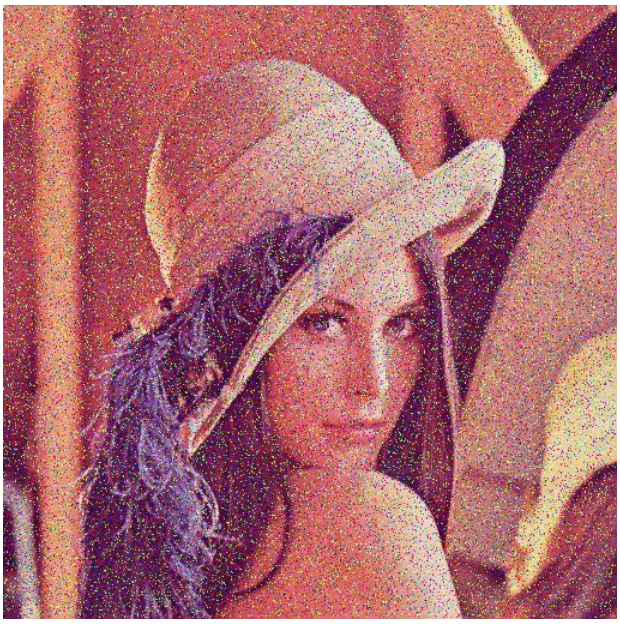
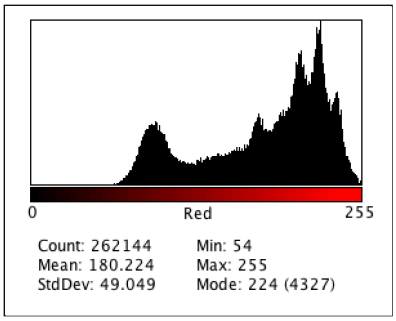
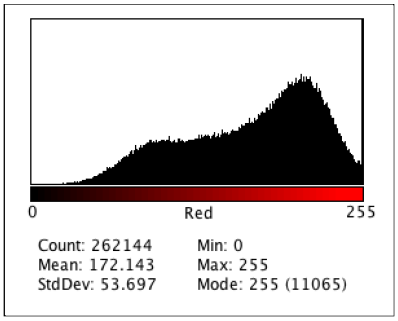
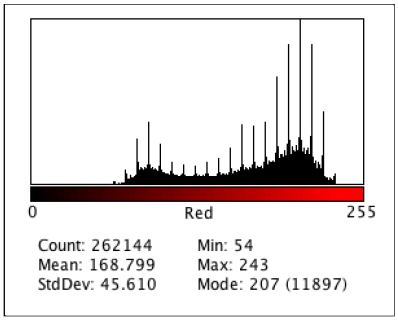


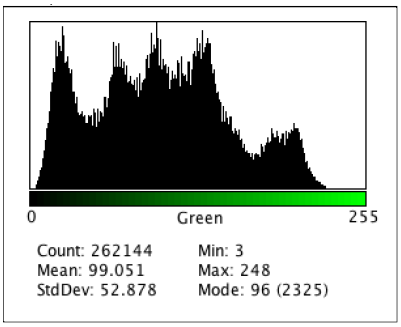
Figure: Lena Mix noise

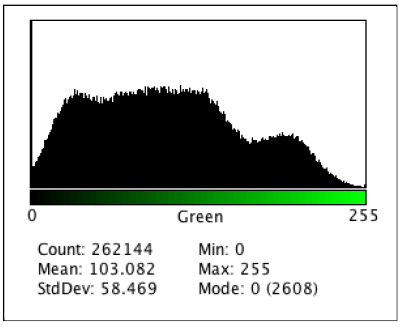
Figure: Lena denoised

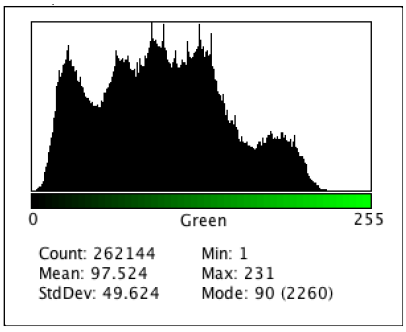


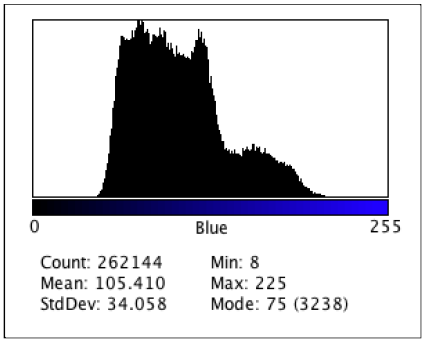


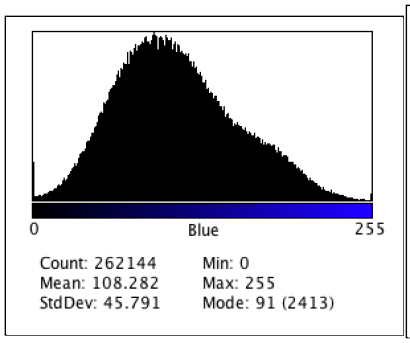


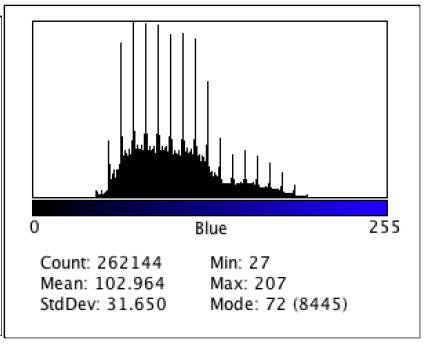












R	G	B	PSNR RED	PSNR GREEN	PSNR BLUE
m+M	m+M	m+M	23.0726	27.3346	25.4175
m+m+M	M+m	M+m+m	24.0726	27.2651	25.9804
m+m+m+M	m+m+m+M	m+m+m+M	24.1669	27.1235	26.1546
m+m+m+m+M	m+m+m+m+M	m+m+m+m+M	24.2006	27.0233	26.2178
m+m+m+m+m+M	m+M	m+m+m+m+m+M	24.2146	27.3346	26.2405
M+(10)m	M+m	M+(10)m	22.2789	26.06	25.4929

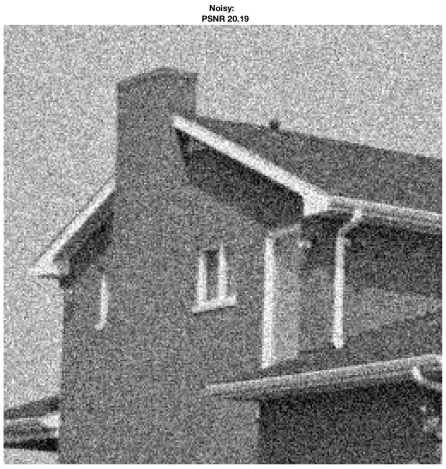
Median(m)

Mean(M)

(b) Principal Component Analysis:

Figure : House.raw original image

Figure: House_Noise.raw



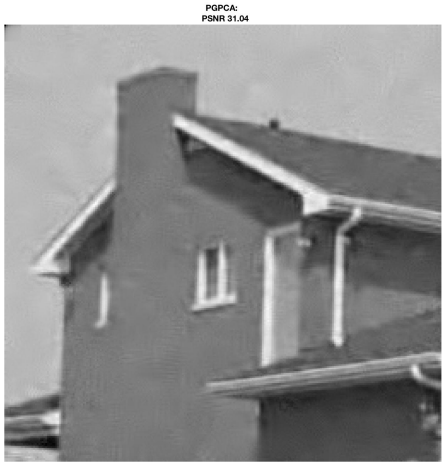


Figure: Noise image produced by adding noise Figure Denoising in PGPCA

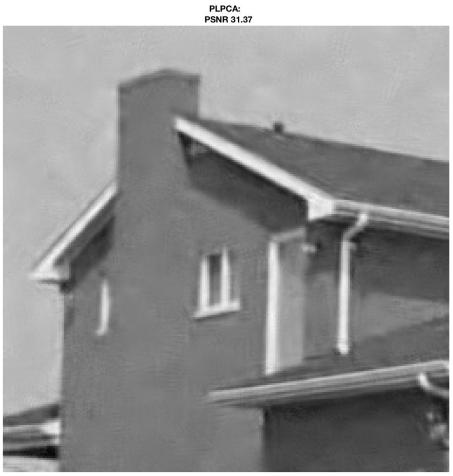


Figure Denoising in PLPCA

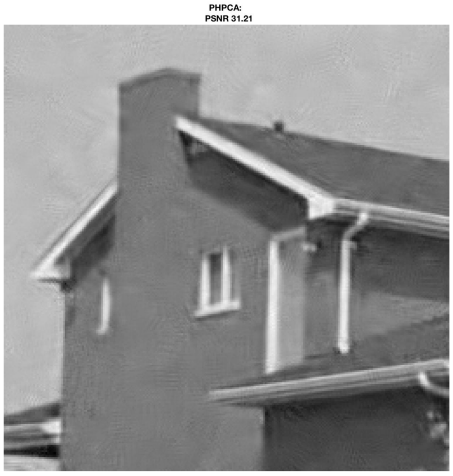


Figure Denoising in PHPCA

```
>> DEMO_GPPCA_online

ans =

    'PGPCA:
      PSNR: 31.04 dB
      Computing Time: 1.14 s.'


ans =

    'PLPCA:
      PSNR: 31.37 dB
      Computing Time: 1.73 s.'


ans =

    'PHPCA:
      PSNR: 31.21 dB
      Computing Time: 1.57 s.'

>>
```

Figure: PSNR values for optimized parameters.

TABLE: PSNR values while optimization(#of components is kept constant)

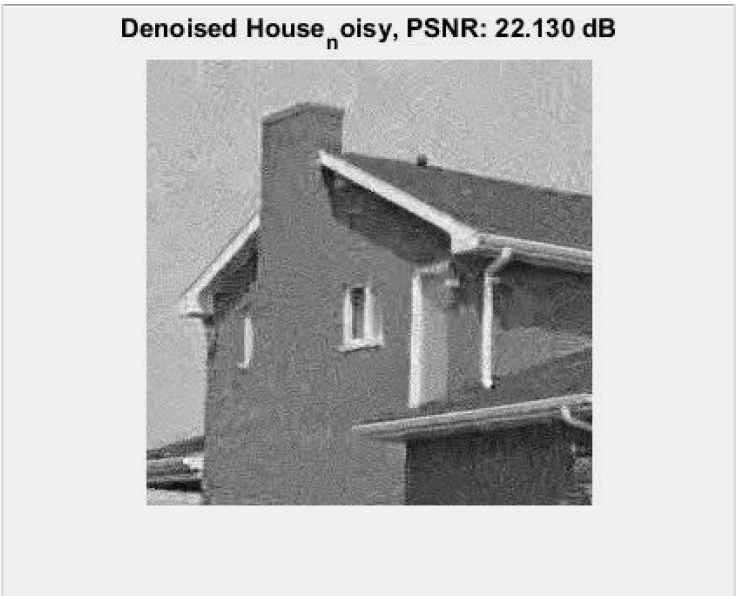
hW-half size	hP-patch size	PGPCA	PLPCA	PHPCA
25	10	31.04	31.35	31.21
25	5	30.53	30.95	30.95
25	7	30.80	31.28	31.18
10	7	30.80	31.10	31.18
15	7	30.80	31.24	31.18
25	7	30.80	31.27	31.21
35	10	31.04	31.04	31.21
35	15	31.04	31.04	
12	23	31.04	31.04	
11	24	31.02	31.32	31.16

(4) When the House_noisy.raw is used for denoising using part(a) program, we get the following result,

PSNR for R-Channel: 30.7318
PSNR for G- Channel: 30.7404
PSNR for B-Channel: 30.7511
Average PSNR: 30.7412

 It looks more out of shape than it should be.

(c) Block Matching and 3-D transform filter:



DISCUSSION:

(a)Mix noise in color image:

Firstly, we analyze the histogram of the given Lena image and Lena-Noise image. There we can see that, in comparison to the histograms of the noiseless image, the image with the noise was slightly Gaussian distributed. And we could notice small spikes all over the curve, and also spots in the image. This denotes the salt and pepper noise. Some cases have impulsive noises too. The absence of impulsive noise doesn't mean than white or Gaussian noise is widespread in the image (as it is fact that with the Gaussian noise, impulsive noise is also present).

Salt and Pepper noise can be removed using median filter, but median filter is not the best filter that can achieve denoising of salt and pepper noise. There are many other standard filters that are in use, still median is chosen because of the ease of complexity to bear with.

Gaussian noise is removed by cascading the median and the mean filter. Mean filter because, the Gaussian noise is signal independent and independent of the noise present in its neighborhood. This mean filter averages out the noise from that particular pixel at each iteration.

Apart from the filter that is used, window size and cascading of the filters also play an important role in image denoising. We see that when we use 5x5 window for median filtering and 3x3 window for mean filtering, we achieve better PSNR value. The table is shown above in the experimental section. But one has to take note that, as we keep increasing the window size, there is pixilation in the output image.

And we can notice that as we keep cascading the image starts to smoothen and ends up over-smoothened and dull.

Here based on the cascading result of filters we can see that; salt and pepper noise is more prominent in R and B channel. And the Gaussian noise is present I all the 3 channels.
(1) All channels have both the noise but the amount of noise present in each of the channels is not the same. Gaussian noise is present in each channel and Salt and pepper noise is more prevalent in R and B channels.
(2) For Gaussian noise, we can apply the mean filter to all the channel at once, but for the salt and pepper noise we'll have to apply for channels separately.
(3) Mean and Median Filters in cascade.
(4) No, we can't cascade the filters in any order. We'll have to apply the median filter first and then we'll apply the mean filter. This is because, to remove the Gaussian noise, we'll have to cascade median and mean in order.
Another filter that can be used for effective denoising is the wiener low pass filter. It prevents loss of information while denoising.

(b) PCA: Principal Component Analysis

(1) Image denoising has always had to make a compromise between the reduction of noise and preserving significant amount of information about the image. In most of the filters that are used, only either of the two is achieved. PCA is a pattern identifying algorithm, where it expresses the information of the image by highlighting the differences and similarity in the image information. Data identification and patterning is very difficult to deal with high dimensional image. But with PCA we can analyze the pattern and compress the image or reduce its dimensions without much loss of information pertaining to the pixels. Mathematically, it rotates the data set in spatial dimension in such a way that it withholds most significant data and truncates or can neglect the less significant ones. i.e, the image details are stored leaving behind the noise. Here the number of components mean the number of axis present in the correlation matrix. This axis will give information regarding the total number of components present and the significant number of components to be considered while the rest is truncated during image reconstruction.
(2) & (3)
From the output images, we can see that the image is reconstructed without much loss in the image information and reduction in the noise levels.
Here for this Question, we assume the standard deviation to be 25. From the output image generated with the application of patch-PCA noise removal technique, we calculate and determine the PSNR value. We can make modifications to the patch size, half size (for the quadrant window) and the number of components to be used to get better PSNR values. (as shown in TABLE: PSNR values while optimization).
(4)
When we use the same noisy image to be denoised with the filtering technique used in part (a) of this problem, we can be sure that PCA denoising is much lossless and efficient. Though in this

method we get good PSNR values i.e noise is significantly reduced, so is the image information. This part of the question proves true to the advantages of using PCA for image denoising.

(c) Block Matching and 3-D transform filter:

(i) Here in BM3D denoising filter, the input image is read and its grided into patches. Like in PCA each patch is compared with all the other patches in the image and is stacked in groups. That is similar patches are stacked together and dissimilar patches are stacked as a new pile of patches. These stacks are denoised with a normal linear filter in stage one and is reconstructed into the image. Here a part of the noise is removed. (When the patches are at stack, weighted average of each patch-stack is evaluated and re assigned to the patch. After this the image is reconstructed.) In stage 2, the most significant stage of the filter, the patches are transformed to frequency domain and noise is removed using low pass weiner filter. In this stage all the non-significant parts of the images are truncates. This then after passing through the weiner filter is re-transfrormed to the spatial domain and is reconstructed into an image.
(ii) In clustering methods such as k-means, if a patch is allocated to a stack it exists only in that vicinity (holds o relation with its neighbors), but in method such as BM3D, a patch in a stack can be attributed to another stack of similar properties. (there comes the weighted average of the patch).
(iii) BM3D filter is both spatial domain and frequency domain filter. The stacking of patches denotes the spatial domain and the transform used to filter the image with the weiner filter involves filtering in the frequency.

REFERENCES:

[1] Deledalle, Charles-Alban, Joseph Salmon, and Arnak S. Dalalyan. "Image denoising with patch based PCA: local versus global." BMVC. Vol. 81. 2011.

[2] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, “Image denoising by sparse 3-d transform-domain collaborative filtering,” Image Processing, IEEE Transactions on, vol. 16, no. 8, pp. 2080–2095, 2007.

[3] [Online]. Available: http://www.cs.tut.fi/~foi/GCF-BM3D/

[4] [Online] http://images.google.com/

[5] [Online] http://sipi.usc.edu/database/

[6] [Online] https://stackoverflow.com/questions/12752168/why-we-should-use-gray-scale-for-image-processing