HOMEWORK#1 Report

Issued: 1/20/2021 **Due: 11:59PM, 2/7/2021**
Name: Siyu Li **ID:2455870216**
E-mail: lisiyu@usc.edu

Problem 1: Image Demosaicing and Histogram Manipulation (30%)

(a) Bilinear Demosaicing (10%)

I. Abstract and Motivation

Due to the property of CMOS sensors, we can only acquire a Bayer-Pattern mosaic-like image when taking a photo. This kind of picture only have the intensity values of pixels in only one channel. To reconstruct RGB color image form the Bayer-Pattern image, the R, G and B channel need to be calculated separately from the Bayer-Pattern image.

In a Bayer-Pattern image, pixels representing red, green and blue distribute in a mosaic pattern on the image. Bilinear demosaicing is a linear reconstruction method, which means that a certain missing point on each channel can be calculated by linear combination of the surround points.

Malvar-He-Cutler (MHC) demosaicing is another method to perform demosaicing, it is an improved version of Bilinear demosaicing. MHC method assume that the intensity values of pixels in different channels are correlated so the values of other channel neighbors can be used to calculate the intensity value of pixel which need to be reconstructed. Because of this improvement, the reconstructed image is expected to be sharper and have higher Peak Signal to Noise Ratio (PSNR) than the bilinear method.

II. Approach and Procedures

To perform the demosaicing operation, we need to use a mask, panel or convolution core with squared shape to calculate the output value pixel by pixel. The size of the mask is $L \times L$ where $L=2 \times r+1$ and the center of the mask is $r+1$. When dealing with points on the edge of the image, there will be lack of points around the center pixel to calculate with some certain points on the mask. Thus, image padding which means enlarging all four corners of an image is applied to perform all calculation on the edge points. For example, the size of convolution core for Bilinear demosaicing L is 3 which means core radius r or in other words width needed to be enlarged on each edge of image is 1.

To address the issue that different method with different convolution core needs different padding width, my approach is to make a function named universalPadding which accept the input of an image and the mask radius r. With this function, we can meet any requirement of padding when facing different size of mask for demosaicing operation.

Then, by calculating the missing values on the R, G and B channels respectively with the corresponding convolution cores we can get the R, G, B channel images. And after arranging three R, G, B images in correct order, the color image will be reconstructed. For the MHC demosaicing, the process of coding is nearly the same except different convolution core and different input parameter r for padding. To show the comparison between the Bilinear and MHC demosaicing method, the PSNR and PSNR for all R, G, B channels separately are applied and calculated.

III. Experimental Results

Fig 1.1 RGB image of house reconstructed by bilinear demosaicing method

Fig 1.2 RGB image of house reconstructed by MHC demosaicing method

Method	Red	Green	Blue	Average
Bilinear	23.1484	34.0179	31.074	29.4134
MHC	35.2616	41.7893	35.2562	37.4357

Table 1.1 PSNR (db) of Bilinear and MHC method for different channels

IV. Discussion

In general, the performance of MHC method is better than that of Bilinear method because the PSNR of MHC result in every color channel and on average is superior to that of the Bilinear result, which means the result of MHC method is more similar to the original image. And as we can observe from the image, when compared to result of MHC method, the color image reconstructed by Bilinear method have more blurred edges where the color and intensity change very sharply. This is because the image is not uniformly distributed and it is always the case that there is some convex or concave between pixels. MHC performs better overall because it takes this property into consideration when designing the mask or convolution core.

(b) Histogram Manipulation (20%)

I. Abstract and Motivation

Due to the reason that for some pictures most of the pixels distribute in a very narrow range, there is little contrast between bright color and relatively dark color, which makes it difficult to recognize some objects in the image. For example, in a certain image, the intensity values of over 75% pixels locating in the range of 0-64, which makes the image a left skewed distribution and looks very dark overall. On the contrary, a picture with over 75% pixels locating in the range of 192-255 will exhibit a right skewed distribution and looks very bright. Both of the cases above need image intensity equalization to get a uniform intensity distribution and balanced brightness which will make it easier to recognize objects. The equalization means the original pixel intensity can be mapped to the new one by some transform so that the intensity distribution is more uniform.

There are two approaches to achieve the goal. One is transfer-function-based method (**Method A**) and the other is cumulative-probability-based method (**Method B**). In the transfer-function-based method, a transfer function is calculated first, and then each pixel value in the original picture is mapped to the new value according to the transfer function. After the procedure above, we will obtain a more uniform image intensity distribution. While in the cumulative-probability-based method, all the pixels are first rearranged based on intensity values in an ascending order. Then do random permutation among the pixels with the same intensity value. And regroup all pixels into 256 groups by the rearranged and randomized order before with the same number of pixels each group. Finally, assign value 0-255 to each group by their order and an equalized image with uniform intensity value distribution will be acquired.

II. Approach and Procedures

In the transfer-function-based method (**Method A**), it can be proved that the transfer function in this form: $Transfer(x) = L \times F_X(x)$ can transfer the pixel intensity distribution from the skewed one to the uniformed one with probability mass function (PMF):1/L. Here x is the old intensity value of each pixel and $F_X(x)$ is the cumulative distribution function (CDF) of x. To acquire the transfer function, we need to get the CDF by operating discrete integral to the PMF of x which is just the frequency histogram of pixel intensity values. For the frequency histogram of pixel intensity values, we can realize it just by traverse all the pixels in the image and count the numbers of pixels with the same intensity value. After the transfer function is calculate, we can just traverse all pixel values to calculate every pixel of equalized image by the transfer function. At this time by calculate the frequency histogram of pixel intensity values, we can see that it is a uniform distribution with the mean of 1/L that is 1/256 because the intensity level here is 256.
Meanwhile in the cumulative-probability-based method (**Method B**), all the pixels are first rearranged based on intensity values in an ascending order. Then do random permutation among the pixels in one group where all the pixels have the same intensity value. The reason for this step is that when pixels are being rearranged they are actually carrying spatial information because when traversing pixels the order is usually from above to below, left to right. For example, let us assume 10000 pixels have the value 1. And after all the pixels in the image are regrouped into 256 group with equal number of pixels, each group will have only 2000 pixels. So the pixel from the 10000 pixels will be assigned intensity from 1 to 5 based on the order of traversing when doing rearrangement. In this case, the picture will exhibit noticeably brightness difference between each region. The above and left part of the picture will be brighter than the below and right part because they are traversed first in rearrangement process. The next step is to regroup all pixels into 256 groups by the rearranged and randomized order with the same number of pixels each group. Finally, assign value 0-255 to each group by their order and an equalized image with uniform intensity value distribution will be acquired. And the output equalized image will be nearly the same as the transfer-function-based method.

III. Experimental Results

Fig 1.3 Original Toy Image

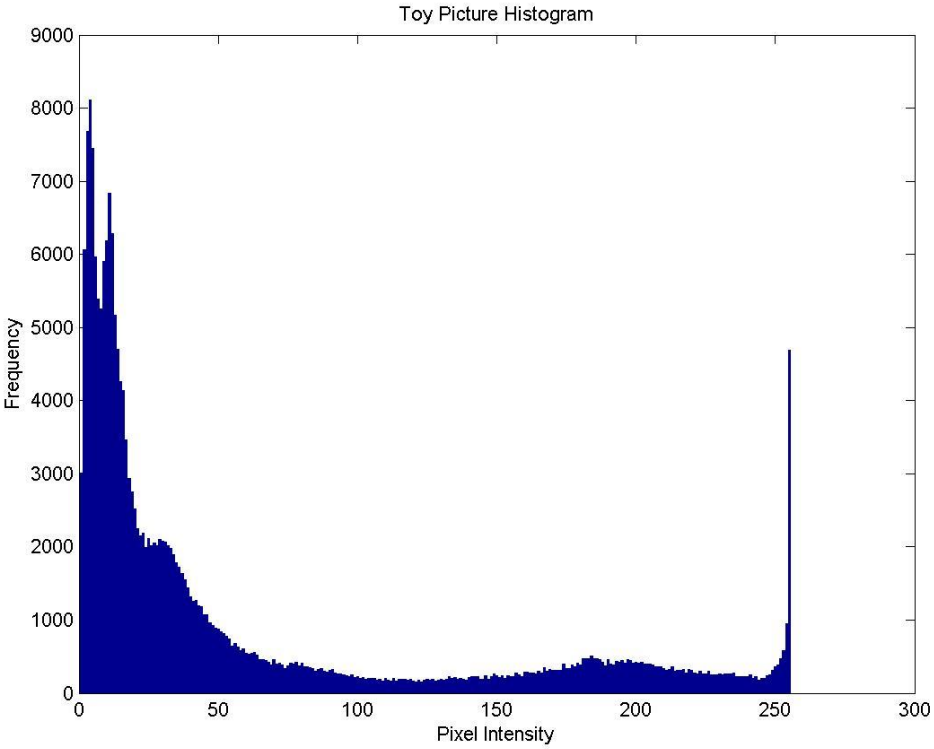


Fig 1.4 Histogram of Original Toy Image

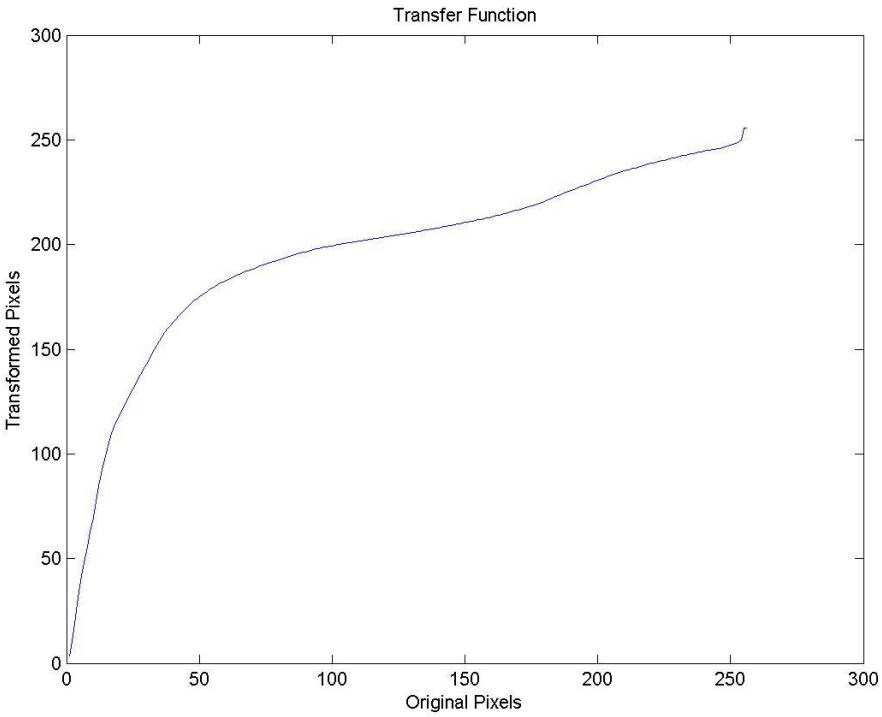


Fig 1.5 Transfer Function Applied to do the Equalization

Fig 1.6 Equalized Toy Image by Transfer Function Method

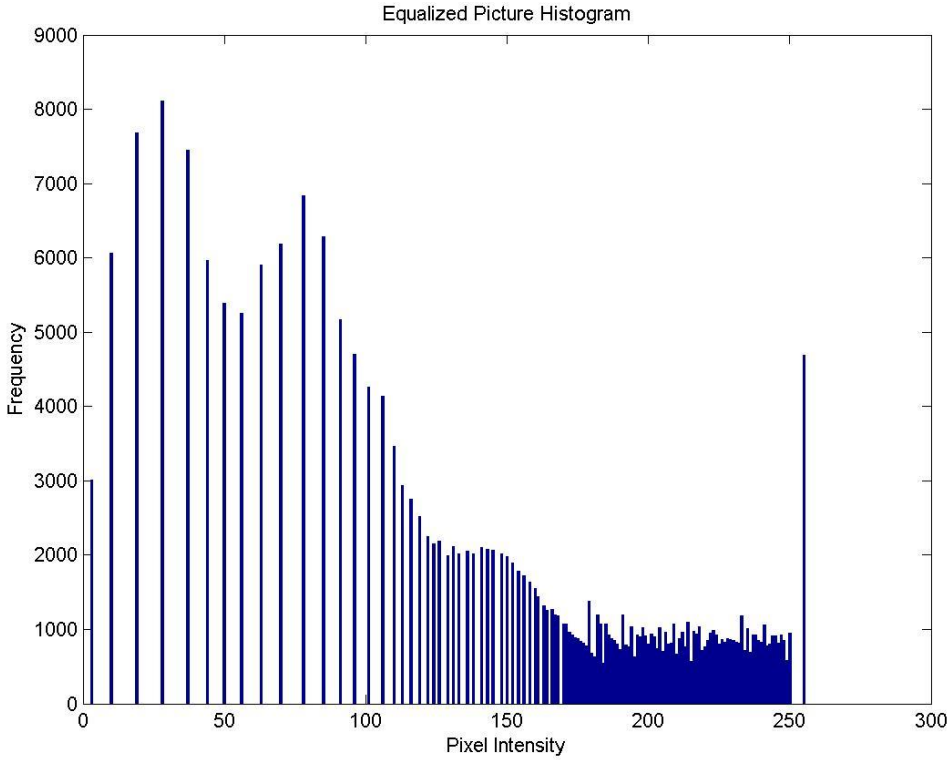


Fig 1.7 Equalized Toy Image Histogram

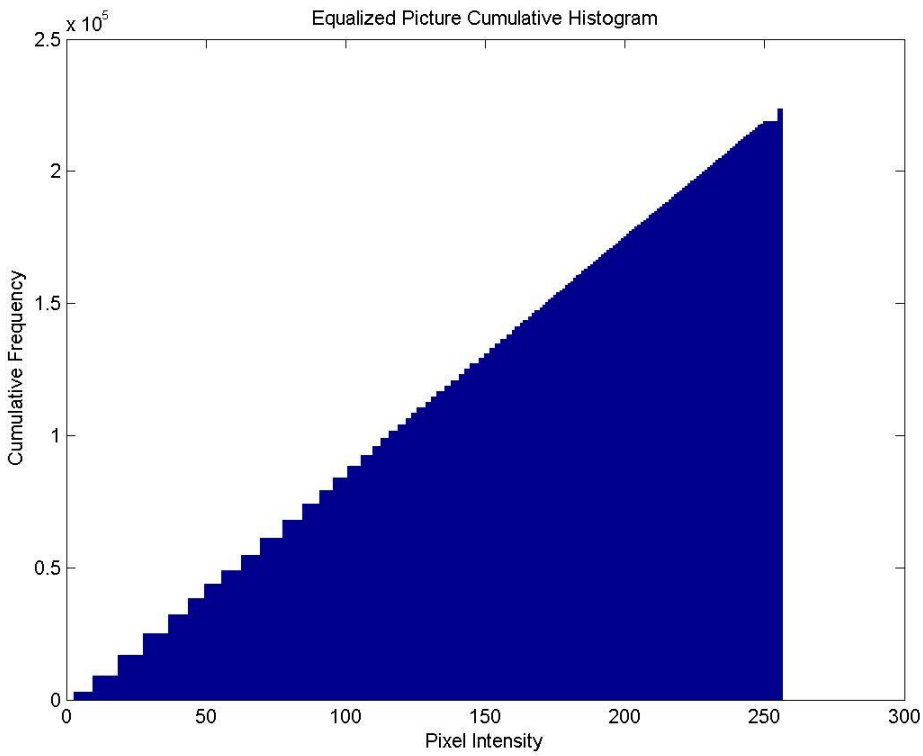


Fig 1.8 Equalized Toy Image Cumulative Histogram

Fig 1.9 Cumulative Probability Based Method Equalized Toy Image

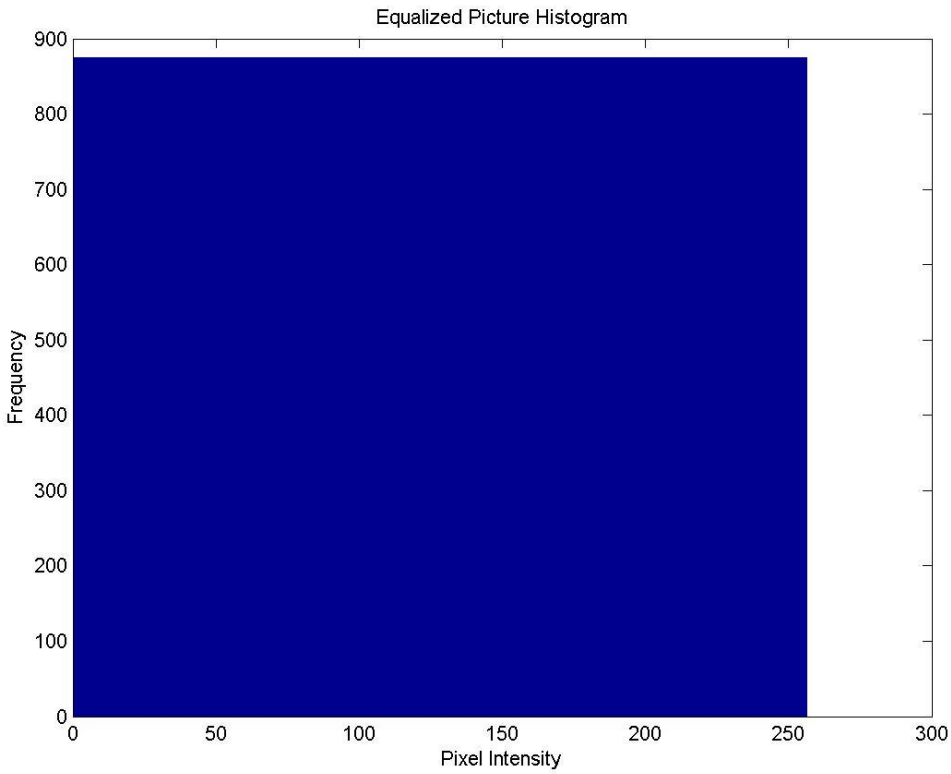


Fig 1.10 Equalized Toy Image Histogram

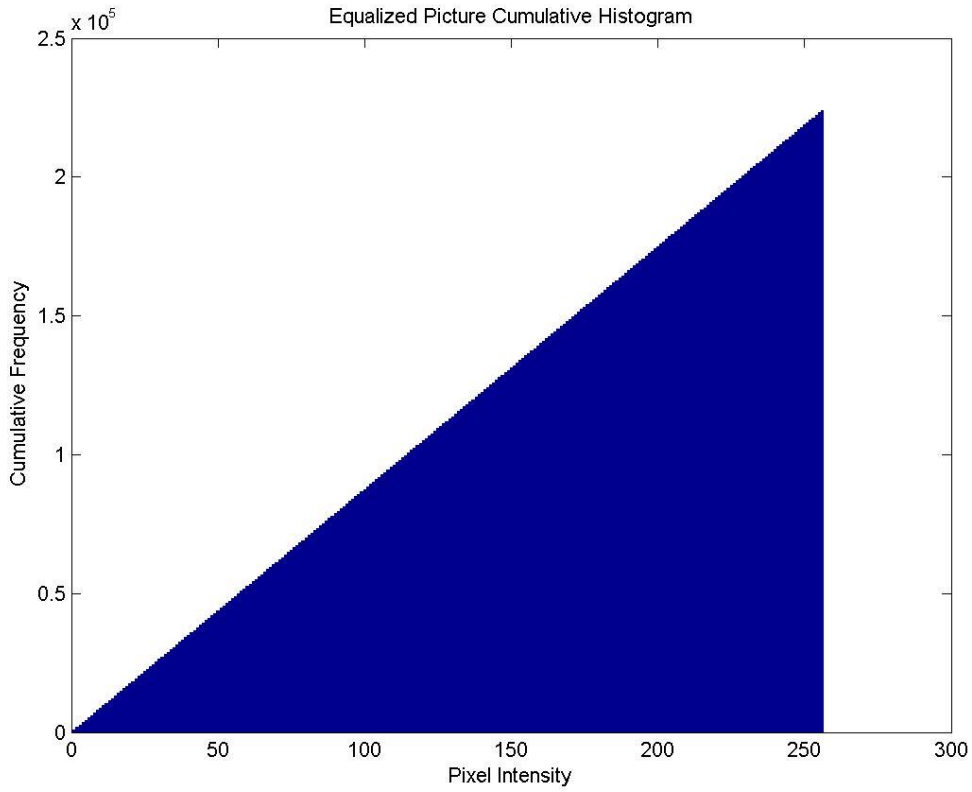


Fig 1.11 Equalized Toy Image Cumulative Histogram

IV. Discussion

From Fig 1.6 and Fig 1.9 above, we can observe that the performance of the two equalization methods is nearly the same. But there is still some nuance worth analyzing. The method A is a function-based equalization method which indicates that the mapping between old and equalized pixels is one to one mapping rather than one to multiple mapping. Meanwhile method B works on rearrangement and regrouping of pixels which may provide one to multiple mapping in some certain cases. For an intensity value group which contains many pixels, the pixels can only get one new value after equalization in method A while in method B they can get more than one value.

This property is reflected in Fig 1.7, Fig 1.8, Fig 1.10 and Fig 1.11. As is shown in Fig 1.7 and Fig 1.10, every frequency value at every pixel intensity is the same in Fig 1.10 but on the contrary very different in Fig 1.7. The frequency of pixel intensity under 150 in Fig 1.7 is very high because due to one-to-one mapping restriction in method A we cannot redistribute the data points on the intensity point with high frequency to its neighbor intensity point to make the distribution more even. This feature is also reflected in the jagged shape in cumulative histogram in Fig 1.8. In conclusion,

method A only makes the intensity distribution closer to uniform distribution while method B makes it exactly uniform distribution so in Fig 1.11 method B cumulative histogram we can hardly observe jagged shape.

Overall, both methods can do the equalization very well and there isn't much very obvious difference. Due to the one-to-one intensity mapping, method A creates more gaps between pixels than the original image, which will generate more unpredictable contours, typically where the pixel intensity values change abruptly. Method B can make the pixel values exactly uniform distribution which will make the equalized image smoother than method A. However, due to the random rearrangement procedure before regrouping pixels to 256 new groups, the pixel intensity spatial distribution may vary from time to time, which means the performance is unpredictable.

Problem 2: Image Denoising (40 %)

I. Abstract and Motivation

It is always the case that an image has some noise and the most common one is Additive White Gaussian Noise (AWGN), which has a zero mean and a certain standard deviation. There are various methods to eliminate or alleviate the noise and the simplest one is to replace the center pixel value with its weighted neighbor values. However, this method will more or less result in edge and texture information loss, which makes the image look blurry. This phenomenon is especially serious when the size of filter mask or convolution core has not been set the proper value.

To address this issue, bilateral filtering is proposed. It not only considers the relative location to the center pixel but also focus on the intensity similarity between the neighbor pixel and center one, which means neighbor pixel with big intensity difference to the center pixel will be given minor weight. This property will result in less loss in edge and texture information and exhibit the noise more effectively but the loss mentioned above still exist. What the two kind of filters mentioned above have in common is that they are both local filter which do not take global information into account.

Non-local Mean (NLM) filter is proposed to solve the theoretical limitation above by taking the global information into account. Rather than just taking neighbor value to calculate filtered center pixel value, it traverses the entire image to look for similar patterns and overlay similar patterns with weights. The noise will be effectively suppressed because the signal to noise ratio will increase sharply after similar pattern overlaying owning to the zero-mean property of AWGN. However, its relatively high computational complexity put a constraint on its application in some certain situations.

Another kind of noise is impulse or pepper noise which is generated by failed censors or oversaturation of a certain pixel. Its statistical property and denoising methodology

are completely different from AWGN. Usually, outlier removal filter such as median filter is applied to deal with this kind of noise. And pepper noise must be eliminated before dealing with AWGN because it will influence the statistical property of AWGN.

(a) Basic denoising methods (10%)

I. Approach and Procedures

In this part, a mask or convolution core is used to calculate weighted neighbor values around the center pixel to get the filtered value. One convolution core is uniformly distributed while the other one obeys Gaussian distribution. Due to the property that the filter core has its own size and radius, which result in lack of values in the original image to do calculation with the core when dealing with pixels on the edge of the image. Thus, image padding is needed to create extra rows and columns by mirror reflection to deal with pixels on the edge. Here image padding function is realized by the function universalPadding which means it can add any number of extra rows and columns on all edges. To search the optimal parameter, different radius of filter and different standard deviation value are traversed and their results are compared in this section.

II. Experimental Results

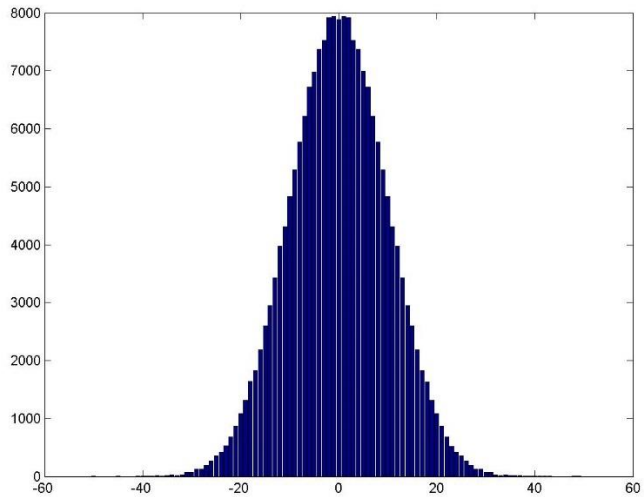


Fig 2.1 Noise Distribution of the Image

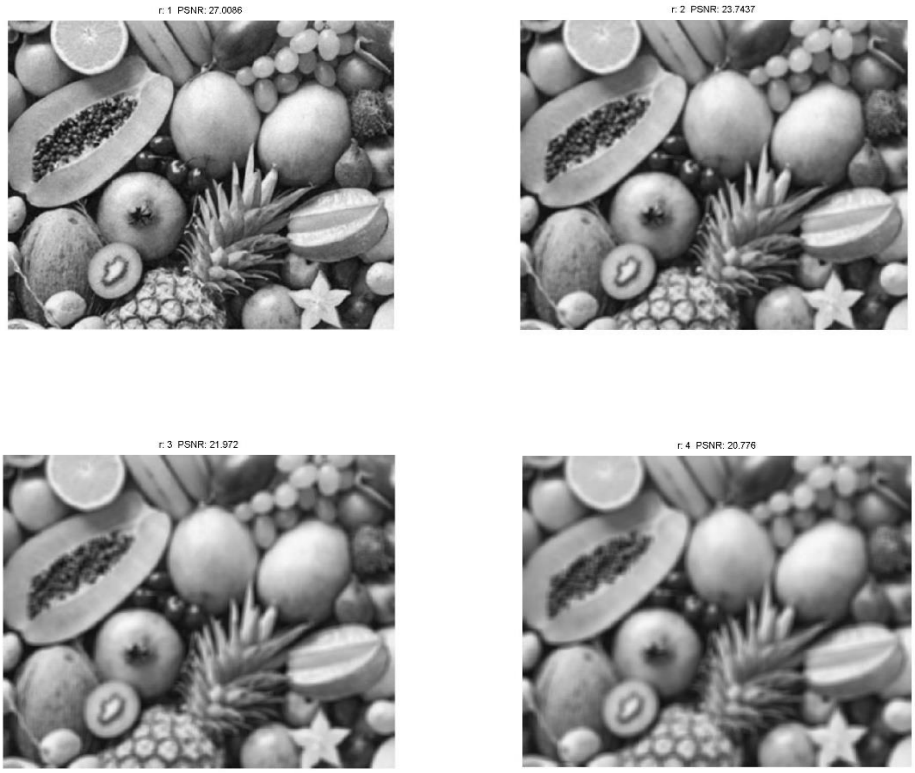
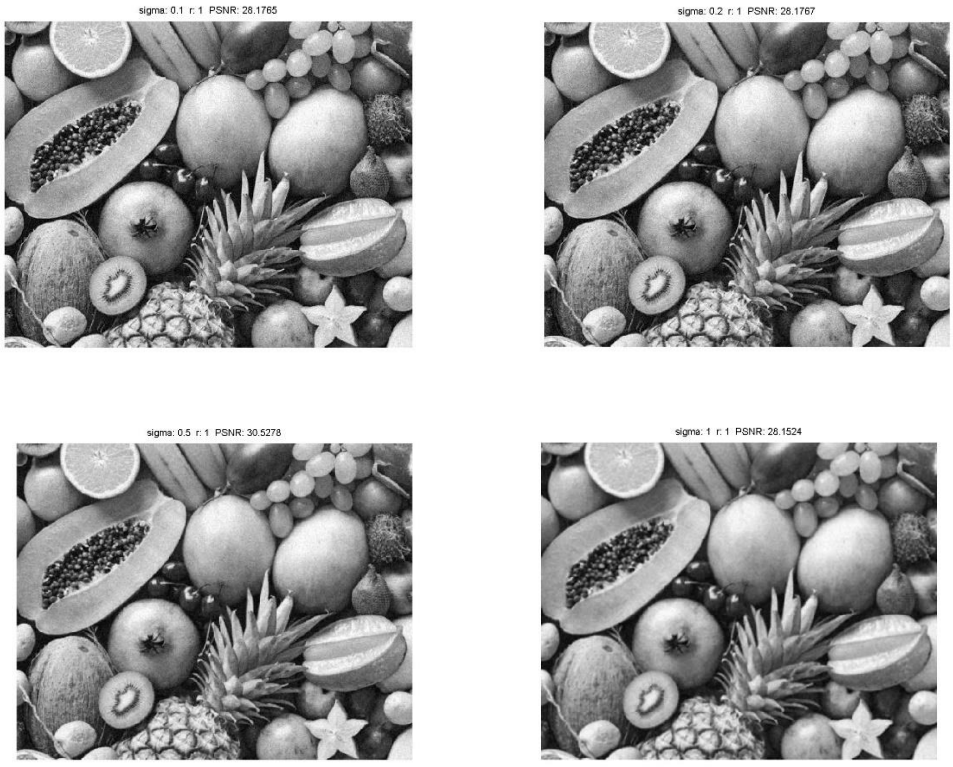
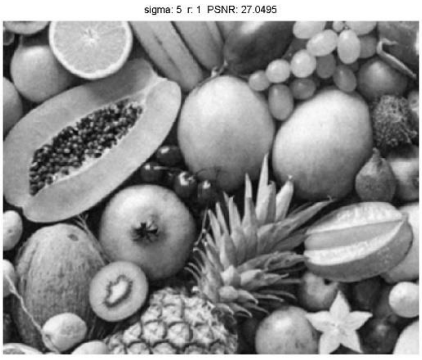


Fig 2.2 Filtered Image by Basic Denoising Method with different filer radius (Uniform Weight Function Core)





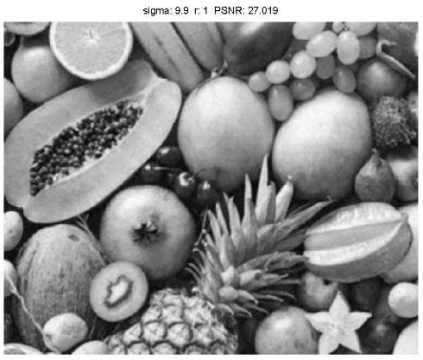
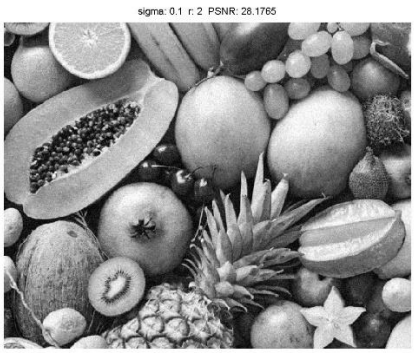
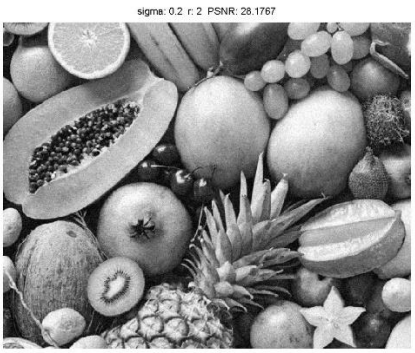
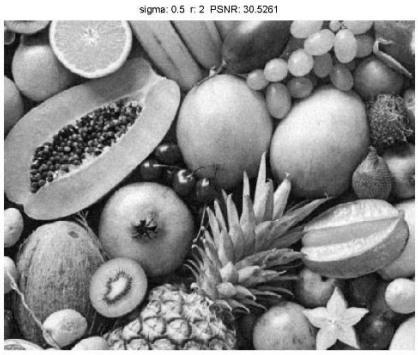


Fig 2.3 Filtered Image by Gaussian Weighted Function Core (r=1)







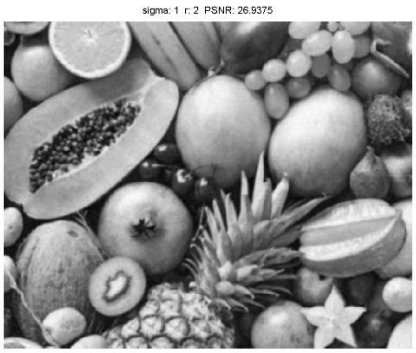
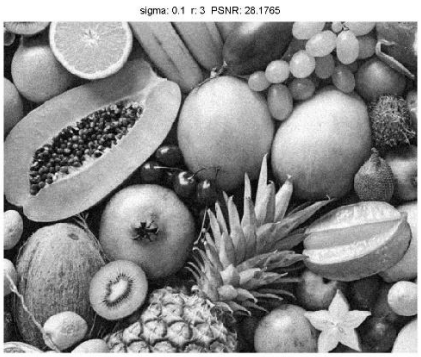
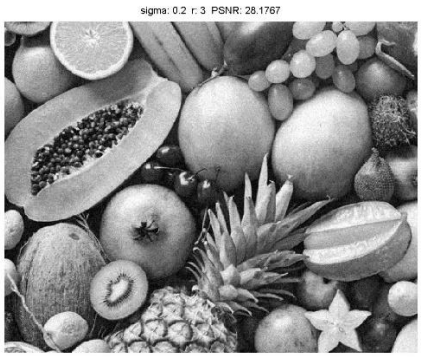


Fig 2.4 Filtered Image by Gaussian Weighted Function Core (r=2)





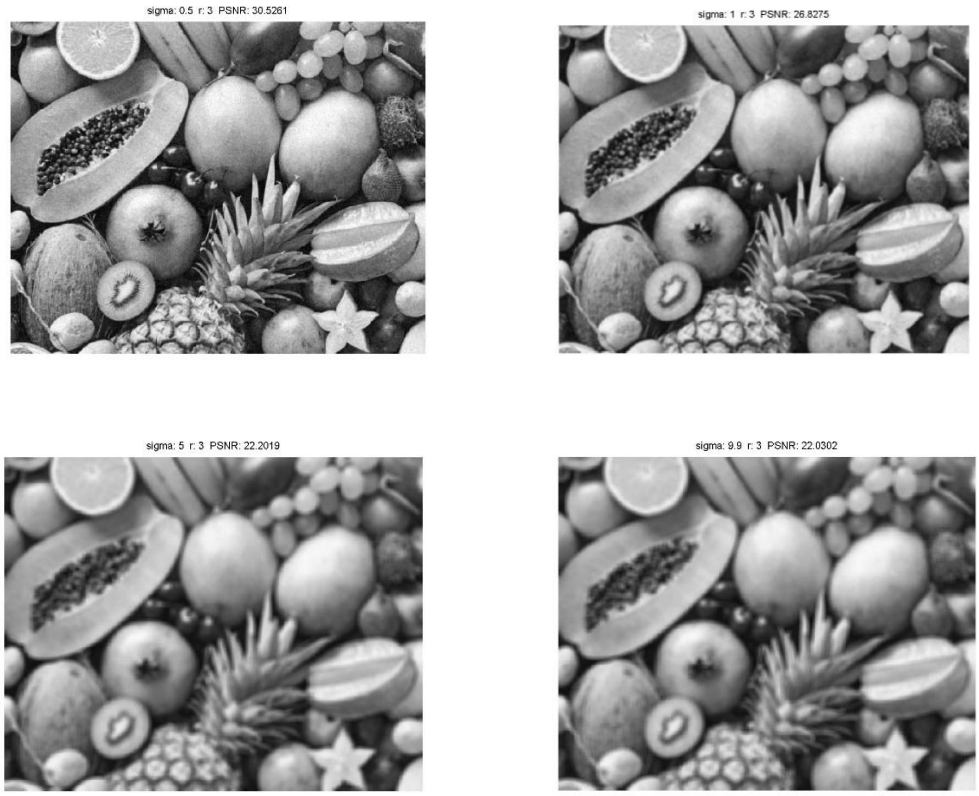


Fig 2.5 Filtered Image by Gaussian Weighted Function Core (r=3)

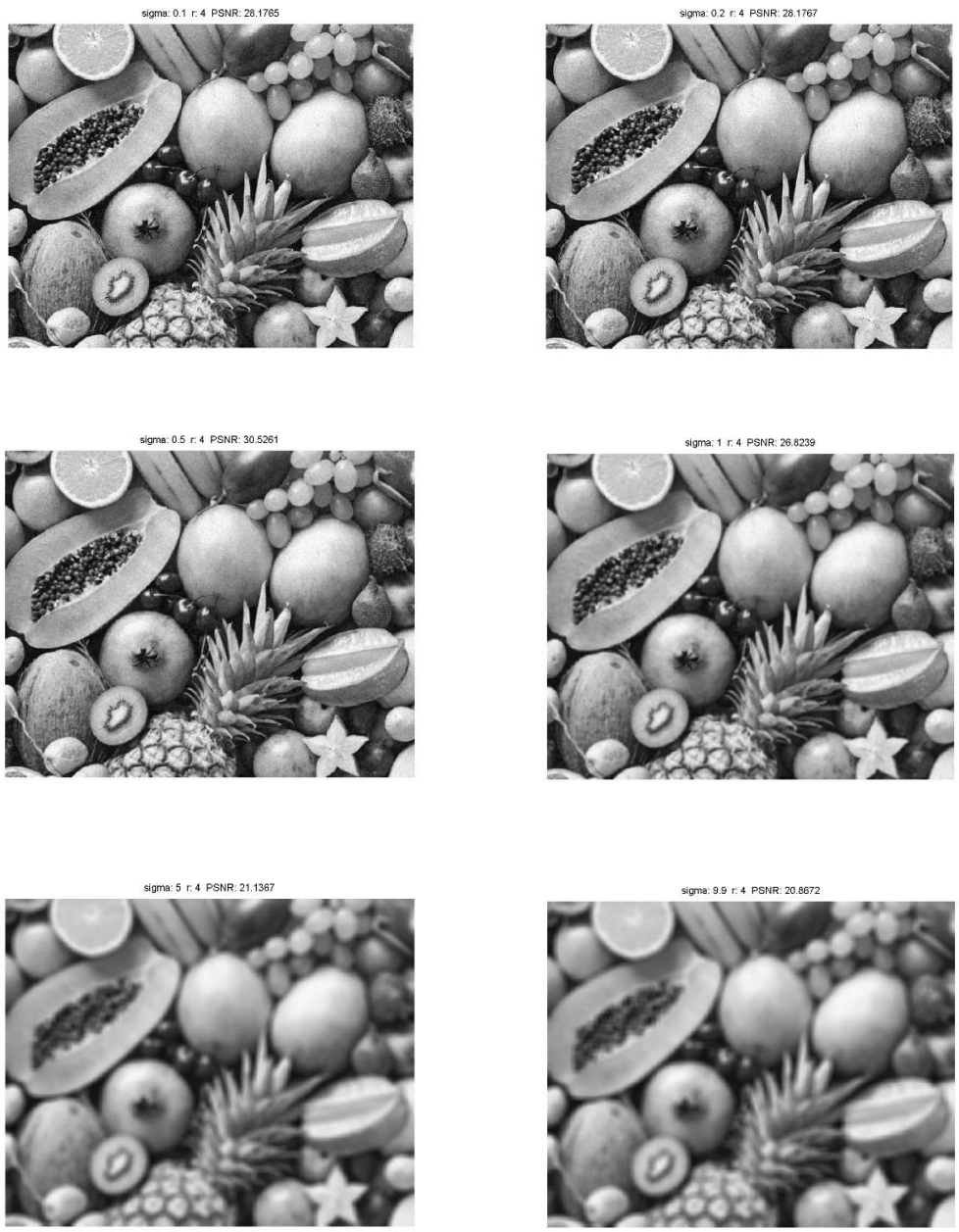


Fig 2.6 Filtered Image by Gaussian Weighted Function Core (r=4)

III. Discussion

(1) The noise obeys Gaussian distribution with mean -0.03787 and standard deviation of 9.9474.

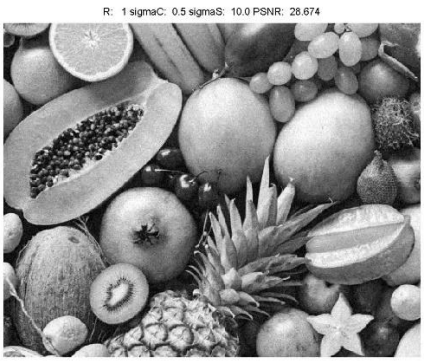
(2) The result of images filtered by core weighted by Uniform distribution does not look very well because it is very blurry and there is huge loss of edge and texture information. The PSNR peaked **27.0086** when filter radius r=1 in the case of **uniform distribution function**. And with the core weighted by **Gaussian distribution function**, the result is better as loss of edge information and blurry feeling become noticeably less. The PSNR peaked **30.5278** when sigma=0.5 and r=1.

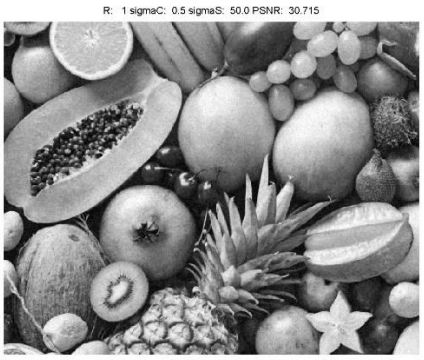
(b) Bilateral Filtering (10%)

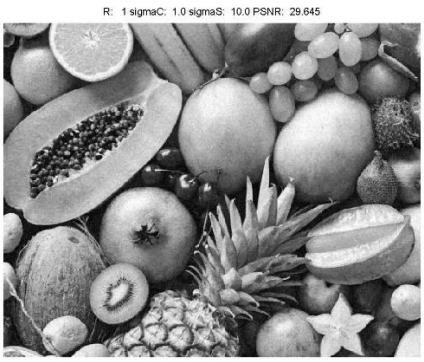
I. Approach and Procedures

The way of programming is nearly the same except that the weighted function of filter core has been changed. The weighted function of Bilateral Filtering has three input parameters which are filter radius, sigmaC and sigmaS respectively. It takes pixel location to the center pixel as well as intensity similarity into account, which makes it perform better.

II. Experimental Results







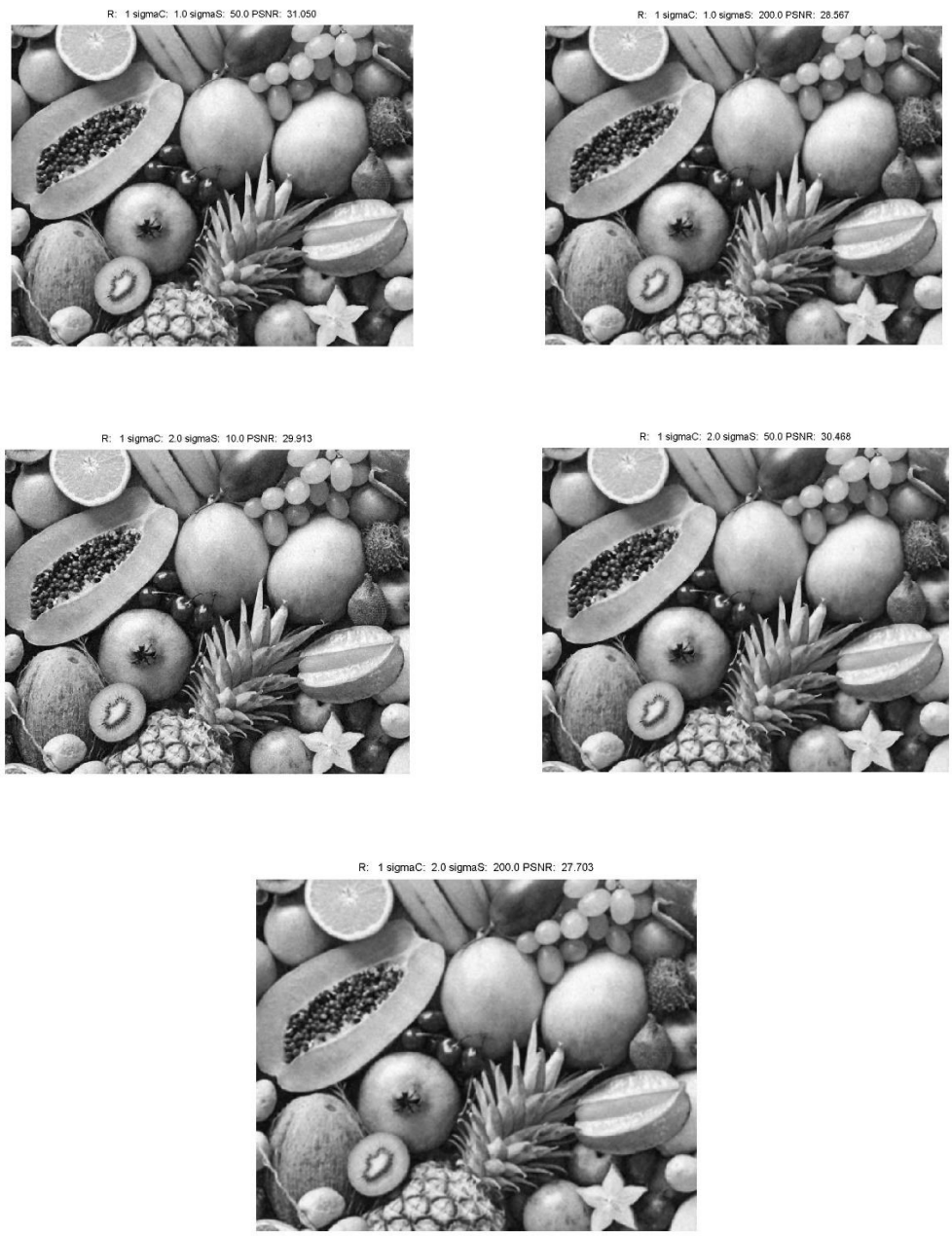
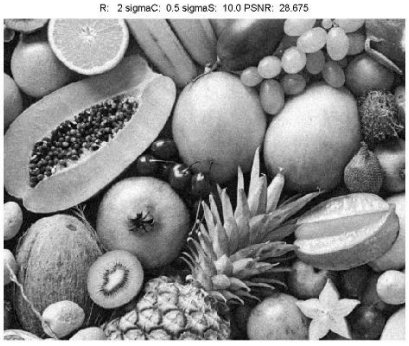
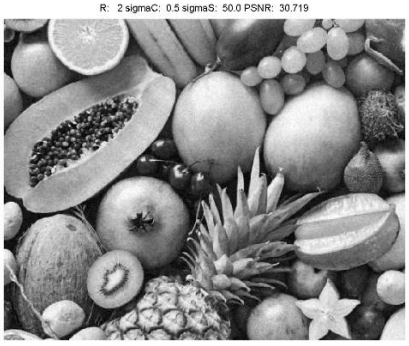


Fig 2.7 Image Output by Bilateral Filter with R=1

R: 2 sigmaC: 0.5 sigmaS: 10.0 PSNR: 28.675



R: 2 sigmaC: 0.5 sigmaS: 50.0 PSNR: 30.719



R: 2 sigmaC: 0.5 sigmaS: 200.0 PSNR: 30.626

R: 2 sigmaC: 1.0 sigmaS: 10.0 PSNR: 29.786

R: 2 sigmaC: 1.0 sigmaS: 50.0 PSNR: 30.725

R: 2 sigmaC: 1.0 sigmaS: 200.0 PSNR: 27.474

R: 2 sigmaC: 2.0 sigmaS: 10.0 PSNR: 30.167

R: 2 sigmaC: 2.0 sigmaS: 50.0 PSNR: 29.298

R: 2 sigmaC: 2.0 sigmaS: 200.0 PSNR: 25.178

Fig 2.8 Image Output by Bilateral Filter with R=2

R: 3 sigmaC: 0.5 sigmaS: 10.0 PSNR: 28.675

R: 3 sigmaC: 0.5 sigmaS: 50.0 PSNR: 30.719

R: 3 sigmaC: 0.5 sigmaS: 200.0 PSNR: 30.626

R: 3 sigmaC: 1.0 sigmaS: 10.0 PSNR: 29.793

R: 3 sigmaC: 1.0 sigmaS: 50.0 PSNR: 30.699

R: 3 sigmaC: 1.0 sigmaS: 200.0 PSNR: 27.376

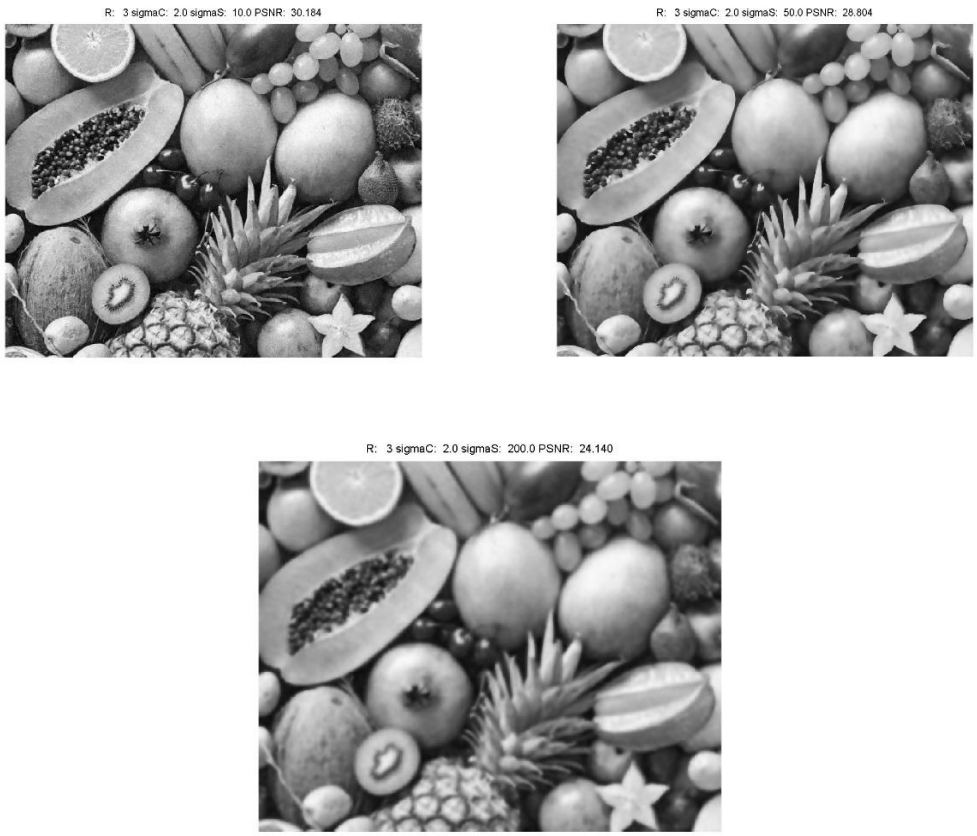


Fig 2.8 Image Output by Bilateral Filter with R=3

III. Discussion

(1) As is observed from the experimental result, when the value of R and sigmaS is fixed, the PSNR increases, and then decreases as the parameter sigmaC ascends. And when the value of R, and sigmaC is fixed, the PSNR increases, and then decreases as the parameter sigmaC increases, which means there is an optimal value for both of the two parameters when the radius of filter R is fixed. The highest PSNR among all the result images is **30.725**.

(2) For the parameter filter radius R, when R=1 and 2, though there are some minor but noticeable difference in each picture, the result based on visual effect and PSNR is nearly the same. And R=3 obviously has relatively low PSNR overall.

(3) Bilateral Filter does perform better than basic filter because bilateral filter can get a higher PSNR than basic filter in our experiment for bilateral filter takes the similarity of pixel intensity into consideration, which can preserve the edge information better. Bilateral filter can be regarded as a refined version of Gaussian weighted basic filter but it is still a local-information based filter.

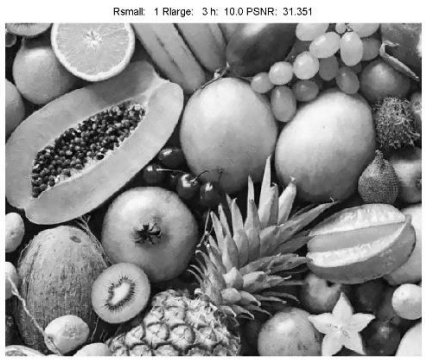
(c) Non-Local Means (NLM) Filtering (10%)

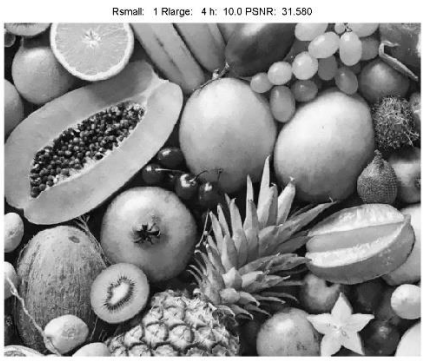
I. Approach and Procedures

In the previous methods, the results are getting better with more information such as intensity similarity taken into account, but the constraint that it can only use local information still exist. Due to the computational complexity limitation, Non-Local Means method are applied with one large window which is the searching range rather than the entire image. And two equal size small windows here are for comparing the relevance between each non-central pixel in the large window and the center pixel. After weight each small window with Gaussian function, we calculated the relevance of center pixel and non-central pixel in the large window by sum up each pixel intensity difference in the small window and normalizing it. Then traverse all the non-central pixels to get all the relevance factor or weight. Finally use all the weight above and normalize them to calculate the filtered image pixel value. In the realization process, we have to pad the image based on filter core radius and decide the large window shape on four edges with taking the minimum between center pixel to unpadded image edge distance and big window radius. Also turning parameter is very important to get improvement over previous method. Improper parameter setting will make the performance even worse than previous simple ones.

II. Experimental Results

To avoid putting too many images here, it has been tested that when h is somewhere near image noise standard deviation and Gaussian weighted function's sigma=25 can perform the best result. Therefore, we only put pictures with different big window size Rlarge and small window size Rsmall here.





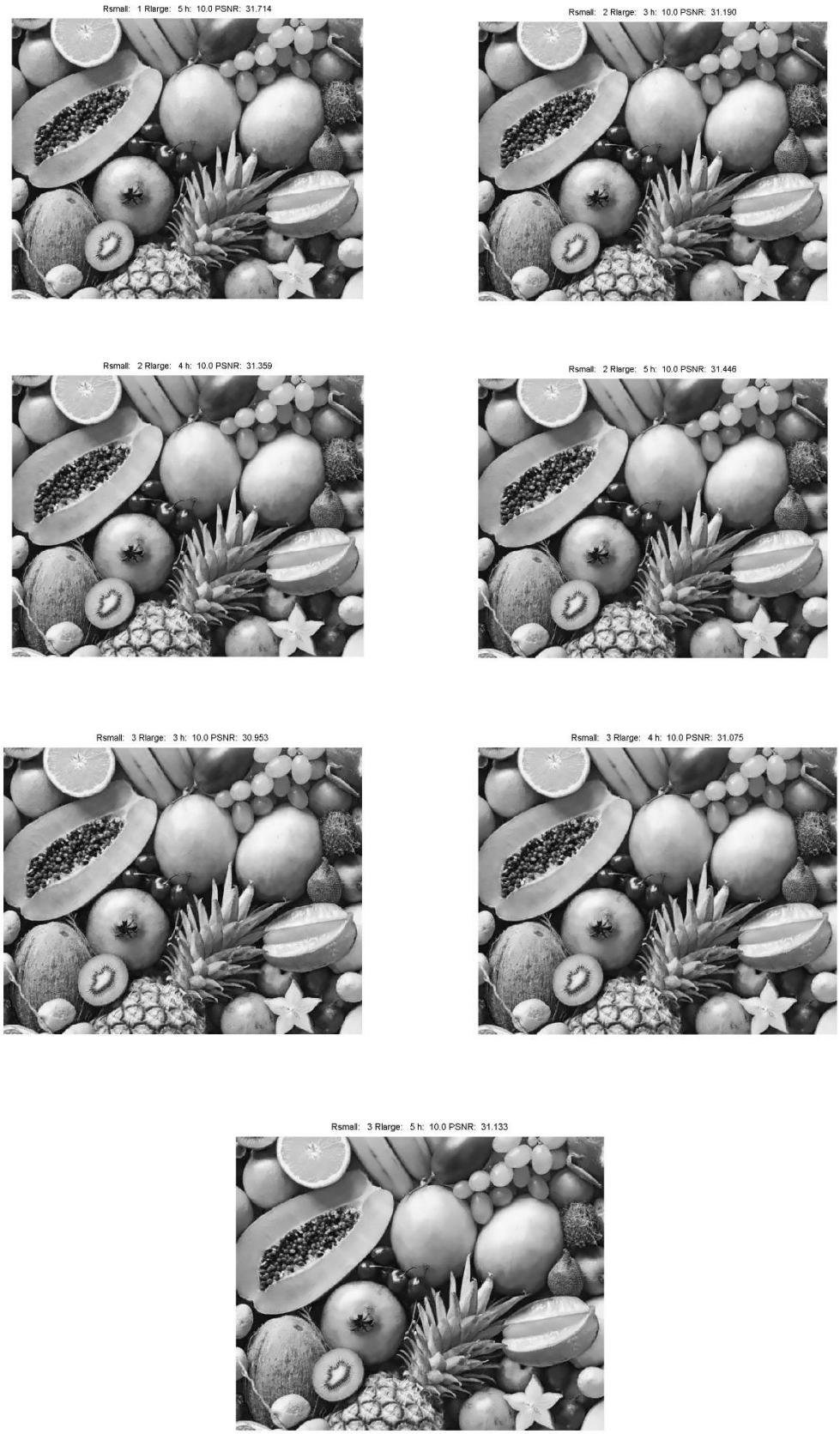


Fig 2.9 NL Means Result with different Rlarge and Rsmall (h=10, sigma=25)

III. Discussion

(1): In this problem I used the NL Means code programmed by myself which can adjust all four parameters so we can test the performance regarding every parameter. Here to avoid too much image we directly conclude that the optimal h can be any value near the noise standard deviation of the image and the optimal sigma is 25 by my experiment result. And improper h and sigma here will make the filtered image noisy and blurry.
It is also worth noticing that h value varies from image to image because the noise standard deviation of each image is not necessarily the same. Therefore, when faced with a new image, its noise standard deviation must be calculated first for later use or terrible result with low PSNR may occur.
And when we fix Rsmall and Rlarge and compare the impact of the other parameter to the result, we can conclude that Rsmall=1 and Rlarge=5 is optimal here with corresponding PSNR 31.714. It is still not sure whether this parameter setting is the optimal one for other images so we also test other Rsmall and Rlarge in Problem 2(d).

(2): The performance of NL Means method result here is much better than all methods analyzed previously owning to its non-local nature. The image with highest PSNR here is **31.714** which is much higher than that of Bilateral method which is only **30.725** and that of basic filter with Gaussian weighted core which is **30.5278**. And the overall performance with all output results and subjective feeling is also better than all the previous methods.

(d) Mixed noises in color image (10%)

I. Approach and Procedures

It is very important to know the noise type by observing its statistical distribution before making any denoising operation because every denoising method has their own limitation and concentration. To analyze the noise type, we create noise histograms of the RGB channels respectively by subtracting the original image from the noisy image and then calculating histogram. It is observed that for R and G channel there is still reasonable amount of values lies above 100 and below -100, this is just impulse or pepper noise which is caused by failed pixels or oversaturation.
To remove this, outlier removal filter is applied. It compares the difference between the average of pixels in the filter core except center one and center pixel value with a threshold to decide whether the center pixel is an outlier. After the outlier removal step, bilateral filtering and NL Means filtering are used to do the rest filtering work. The results and parameter turning details will be discussed later.

II. Experimental Results

Here for briefness, we only post part of the filtered images by Bilateral and NL Means method including the one with best PSNR and optimal parameters. The standard deviation after outlier removal is 47.2661 but by traversing h=30, 40, 47 we find h=30 performs better so we only post NL Means with h=30.

Outlier Removed Picture

Fig 2.10 Original Noisy Picture (above) and Outlier Removed Picture (below)

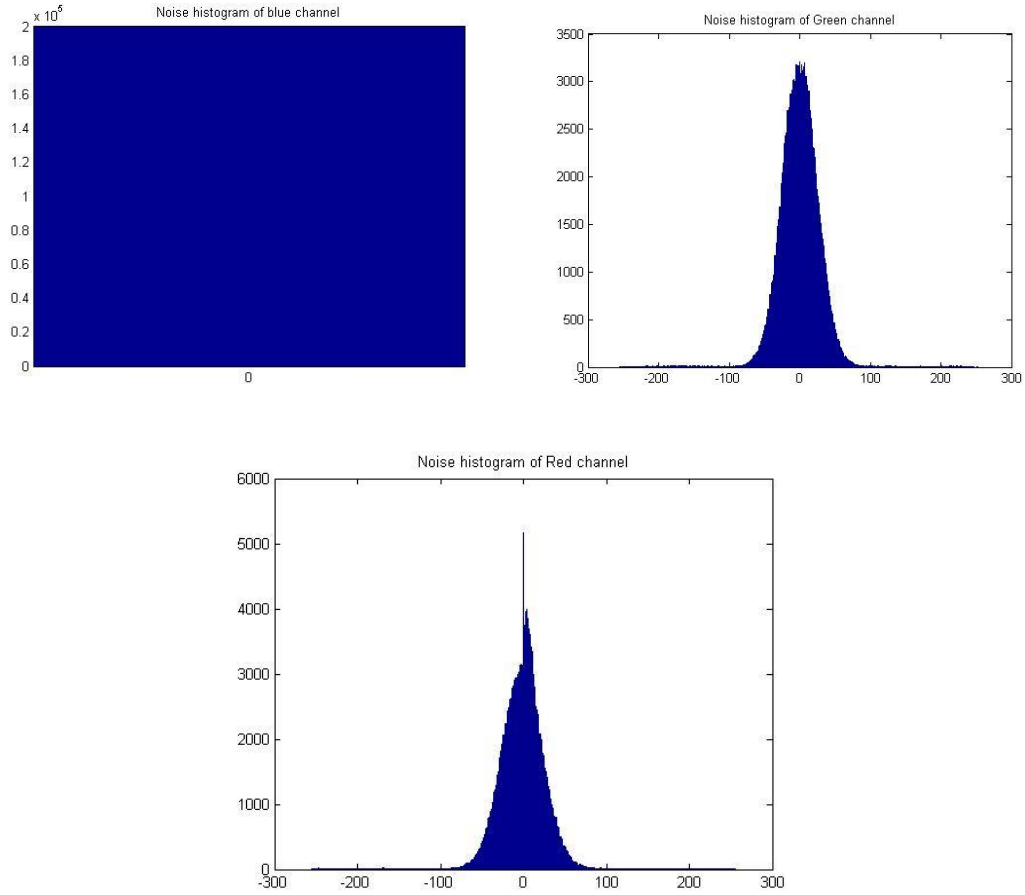


Fig 2.11 Noise Histogram of the RGB channels

Rsmall: 2 Rlarge: 5 PSNR: 22.994

Rsmall: 2 Rlarge: 6 PSNR: 22.872

Rsmall: 2 Rlarge: 7 PSNR: 22.764

Rsmall: 3 Rlarge: 5 PSNR: 22.745

Rsmall: 3 Rlarge: 6 PSNR: 22.587

Rsmall: 3 Rlarge: 7 PSNR: 22.447

Rsmall: 4 Rlarge: 5 PSNR: 22.576

Rsmall: 4 Rlarge: 6 PSNR: 22.398

Fig 2.12 Filtered Image by NL Means with different Rsmall and Rlarge (h=30, sigma=25)

Fig 2.13 Final Result Comparison

(First Row Left(a): Original Noisy Image, First Row Right(b): Outlier Removed Picture, Second Row Left(c): Image Filtered by Bilateral Means with optimal parameters, Second Row Right(d): Image Filtered by NL Means Filter with optimal parameters)

III. Discussion

(1): As is observed from Fig 2.11, the noise in R and G channels are mixture of impulse or pepper noise and AWGN. AWGN property is very easy to observe. And for impulse noise, we can observe that there are a lot of pixel position where the intensity difference between noise image and clean image is greater than 100 or less than -100. This huge difference is cause by pixel in noisy image with intensity 0 or 255. Therefore, it is not difficult to prove that impulse noise exists. And for blue channel there is no noise at all from Fig 2.11.

(2): First, I would like to use Outlier Removal Filter which compares the difference between the average of pixels in the filter core except center one and center pixel value with a threshold to decide whether the center pixel is an outlier. The Outlier Removal Filter must be applied ahead of other filters which rely on the statistical property of image or noise such as bilateral or NL Means filter. If not, impulse noise will be mixed with AWGN which will undesirably change the weights in filters resulting in incomplete elimination of noise.

(3): 1. From the Fig 2.13(b), we can observe that impulse noise is eliminated very well after outlier removal. For the NL Means filter after the outlier removal process described above, after traverse h=30, 40, 50 based on the assumption that h should be somewhere around the noise standard deviation value which is 47.2661 here, it is proved that h=30 outputs the optimal results. We also tried Rlarge=6, 7 as a replenishment for Problem 2.3. But it still turned out that Rlarge=5 is optimal. And best Rsmall and sigma here is 1 and 25. The NL Mean output the best PSNR 23.385 with parameters Rlarge=5, Rsmall=1, h=30 and sigma=25 while Bilateral filter with optimal parameters r=3, sigmaC=1, sigmaS=50 only get PSNR 23.141. Also from the subject observation feel by compare Fig 2.13(c) and Fig 2.13(d), we can feel Fig 2,13(d) is obviously smoother and less noisy. And the output for NL Means is very stable, even without the optimal parameter, we can still get PSNR greater than 22.2 at least.

2. The shortcoming is mainly in the high computational complexity of NL Means method, it takes real long time to search optimal parameters. Another disadvantage needed to be mentioned is that these kinds of filters can more or less rely on the statistical property of input image whose can vary significantly. For example, in the blue channel there is no noise while in other two channels there is a lot.

3. When it comes to improvement, due to the inherent variability of image statistics, we can customize parameters in filters by following the statistical change of each channel when getting a new image rather than using the same parameter for all channels all the time. And to shrink the tremendous time in computing, we can give effective assumption and reasoning to narrow the range of parameters. We can also apply parallel computing method to make the most potential of computer to save time.

Problem 3: Special Effect Image Filters: Creating Oil Painting Effect (30%)

Abstract and Motivation

Quantizing continuous analog signal to discrete digital signal is a very important step in getting a digital image. With more quantization level, an image will have higher intensity resolution which is desirable but it also requires better hardware performance. But it is observed that sometimes with less quantization level we can get blurry visual effect just like the style of old painting. And from the basic filter experiment result above, we can conclude that when the filter core size or radius is too big, the image will also lose edge information and look blurry. With the two characteristics above, we can transform a clear color image to an oil painting effect image by applying less quantization level and mode filter to the RGB channels respectively.

I. Approach and Procedures

My quantization method is to partition 256-pixel intensity levels to 4 equal length intensity interval group such as 0-63, 64-127, 128-191 and 192-255. And assign the original pixel to their corresponding group based on intensity value. To make the transformed image with nearly the same brightness, for each intensity interval group all the pixels take the floored integer value of the average of interval minimum and maximum such as 31 for 0-63. This is the 64-color case where each pixel in a certain color channel has 4 values. When realizing this process in coding, the universalColorReduction function which can deal image with different color quantity by different input is created. This means that the 512-color case can also be dealt with by this function with input r=8. After intensity level reduction to the RGB channels respectively, we apply the mode filter with core radius from N=3 to N=11 to replace the center pixel with most frequent value in the filter core to get optimal result. And then the oil painting effect creation process is finished.

II. Experimental Results

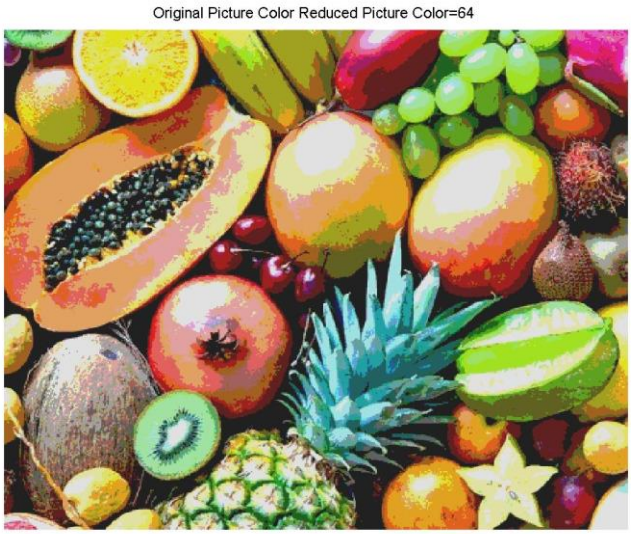


Fig 3.1 64-color Image After Color Reduction

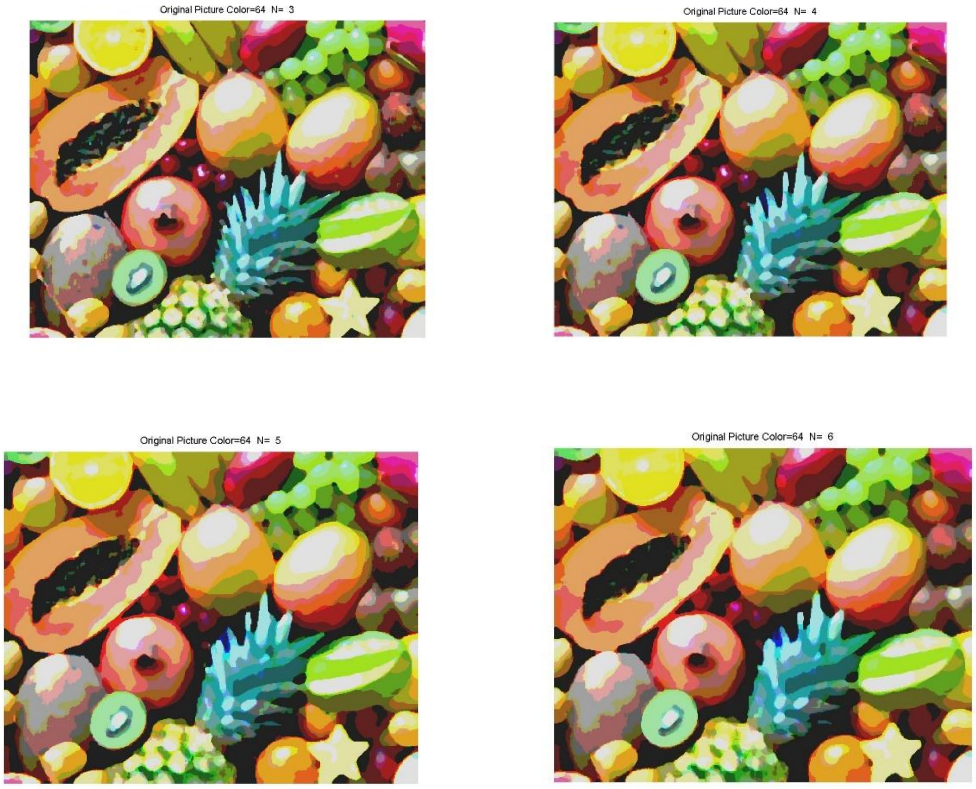


Fig 3.2 64-color Image with N From 3 to 11

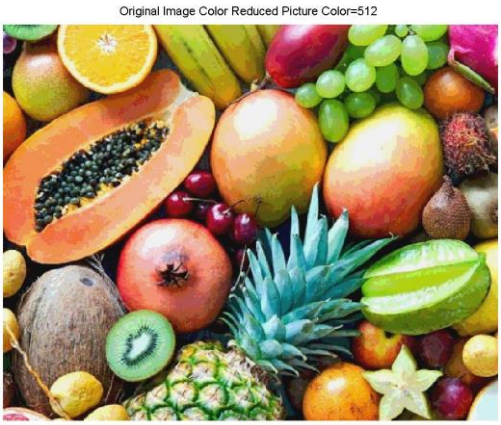
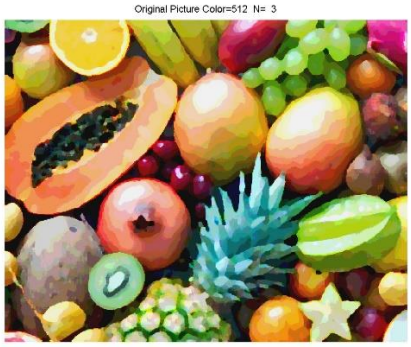
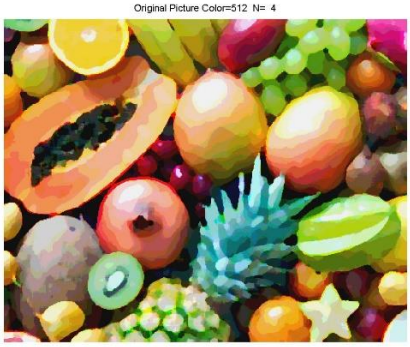
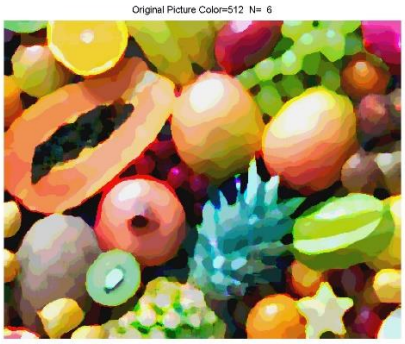


Fig 3.3 512-color Image After Color Reduction







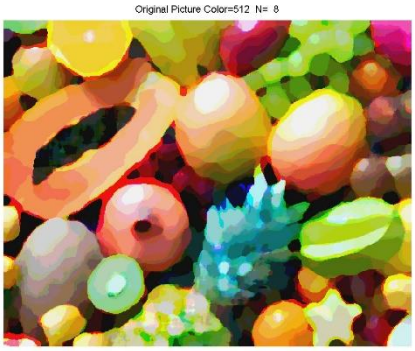
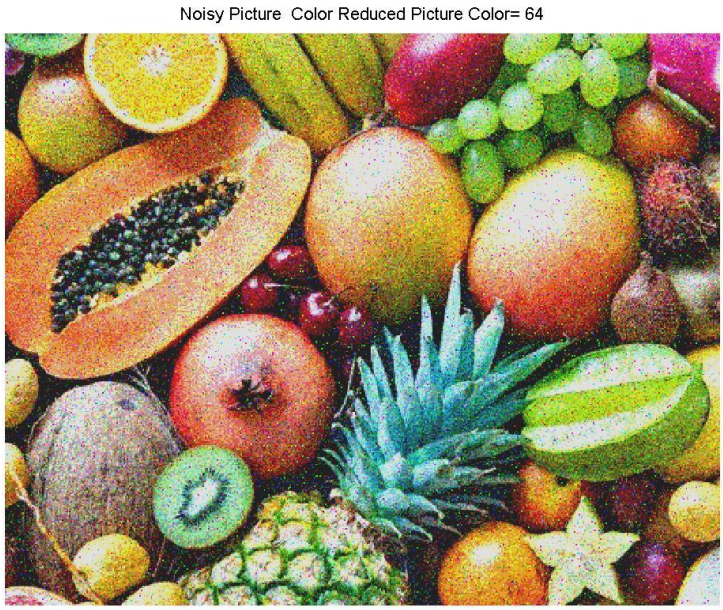


Fig 3.4 512-color Image with N From 3 to 11



Noisy Picture Color Reduced Picture Color= 512

Fig 3.5 64-color and 512-color Image After Color Reduction

Noisy Picture Color= 64 N= 3

Noisy Picture Color= 64 N= 4

Original Picture Color=64 N= 3

Original Picture Color=64 N= 4

Fig 3.6 64-color Noisy Image and Original Image with N=3, 4 and 5

Fig 3.7 512-color Noisy Image and Original Image with N=3, 4 and 5

III. Discussion

(1): The 64-color version of the Fruits image by following Step 1 is shown as Fig 3.1 above and my quantization process is described in detail in section Approach and Procedures above.

(2): Analyzing the performance of oil painting effect method is a very subjective matter. Here I think in the 64-color cases the performance is optimal when N=3 for it maintains reasonable amount of edge information and we can still recognize objects in it very easily. But at the same time, it also creates a blurry oil painting effect which distinguishes it from normal images. My conclusion can be proved easily by observing and analyzing Fig 3.2.

(3): The 512-color result is basically the same as that of 64-color image but there are also some noticeable differences. With more quantization levels, the image after intensity reduction looks less blurry and maintains more texture information. Therefore, the 512-color image filtered result looks less blurry when they are running the same N. In 64-color case, only N=3 result is good based on my standard while in 512-color case with the built in more intensity level both N=3 and 4 results are good. This conclusion can be testified by observing and comparing Fig 3.2 and Fig 3.4.

(4): When performing filtering on both the 64-color and 512-color noisy image, it can be found that the noise in the image has been alleviated obviously by comparing Fig 3.5, Fig 3.6 and Fig 3.7. But the noise can still make a difference between clean one and noisy one after filtering from observing Fig 3.6 and Fig 3.7. I picked N=3,4 and 5 case to show my finding though the optimal for 64-color is N=3 and for 512-color N=3 and 4. What I found is that with the N value going larger the difference caused by noise is shrinking continuously and the noise of 64-color case decreases faster for its blurrier nature than 512-color case.