EE 569: Homework #1

Issued: 1/13/2020 Due: 11:59PM, 1/28/2020

Problem 2: Image Denoising

(a)Basic denoising methods (10%)

(1) The embedded noise in Corn_noisy.raw image is Additive White Gaussian Noise (AWGN).

BOX FILTER

1. Approach and Procedure

Box filter is a form of low pass filter. The convolution kernel has the average values of the box filter. It smoothens the image.

2. Experimental results:

Figure 1a: Noisy Input Image

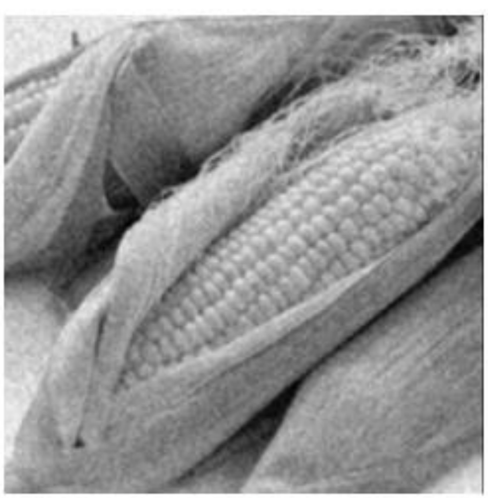


Figure 1b: Box filter size 3x3

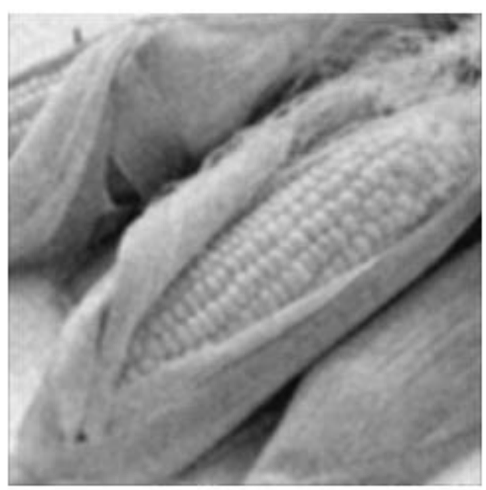


Figure 1c: Box filter size 5x5

3. Discussion

Figure 1a shows the input noisy image Corn_noisy.raw. This noisy image consists of Adaptive White Gaussian Noise (AWGN). The PSNR of the noisy image to the original image is 17.76db.

Figure 1b shows the denoised image after applying a **box filter of size 3x3. The PSNR value of this denoised image is 19.62db.** It is evident that this image has a better PSNR value compared to the noisy image. The image is smoother and reduces the intensity variation between nearby pixels.

Figure 1c shows the denoised image after applying box filter of size 5x5. The PSNR value of this denoised image is 19.44db. The PSNR value has reduced compared to the 3x3 box filter. The PSNR value of this image is lower than the PSNR value of the denoised image with 3x3 box filter. We can see that the edge details of the image are reduced.

From Figure 1b and 1c it can be observed that as the kernel size of the box filter increases the smoothing effect of the kernel increases, thus reducing the fine image details and the performance of the filter.

GAUSSIAN FILTER

A gaussian filter is a low pass filter. Gaussian filtering is done by convolving an image with kernel of Gaussian values. Gaussian smoothing is generally used in edge detection. By applying gaussian blur before edge detection improves the performance of edge detection algorithms.

$$Y(i,j) = \frac{\sum_{k,l} I(k,l)w(i,j,k,l)}{\sum_{k,l} w(i,j,k,l)}$$

$$w(i,j,k,l) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(k-i)^2+(l-j)^2}{2\sigma^2}\right)$$

σ is the standard deviation of Gaussian distribution.

Figure 2a: Gaussian Filter Sigma:0.7

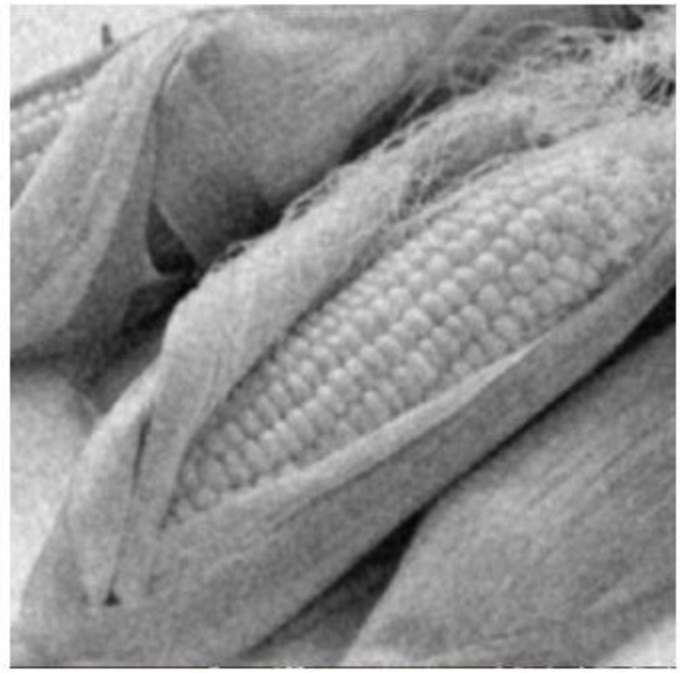


Figure 2b: Gaussian Filter Sigma:1.0

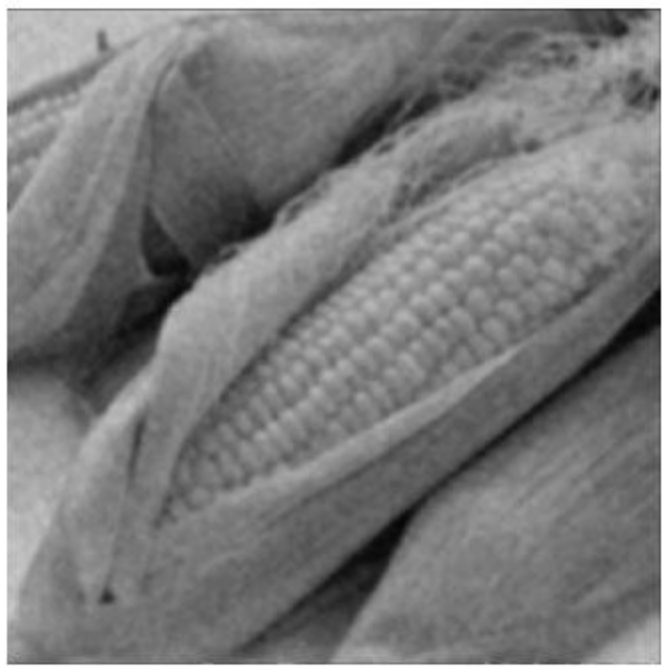


Figure 2c: Gaussian Filter Sigma:1.5

Discussion

Figure 2a shows the denoised image after applying a gaussian filter of **sigma 0.7. The PSNR value of this denoised image is 19.7857db**. The PSNR value is higher compared to the box filter.

Figure 2b shows the denoised image after applying a gaussian filter of sigma 1.0. The PSNR value of this denoised image is 19.0321db. The PSNR value reduced as sigma value increased.

Figure 2c shows the denoised image after applying a gaussian filter of sigma 1.5. The PSNR value of this denoised image is 14.1459db.

Sigma controls the variation of the gaussian filter around the mean. From the above figure, it can be observed that as sigma increases, we are losing the details of the image and the PSNR value reduces. The performance of the gaussian weight function is better than the uniform weight function because higher weights are assigned to pixels that closer to the center pixel in the sliding window and lower weights are assigned to pixels that are far away from the center pixel. This functionality works better because, nearby pixels in a small neighborhood are more correlated than the pixels that are present at the borders of the sliding window.

(b) Bilateral filtering

Bilateral filtering preserves edges and reduces the noise in the image. The weight in Bilateral filtering is gaussian and it depends on the distance between the pixels and the difference in their intensity values.

Experimental Results

Figure: Bilateral Filtering using Sigmac=0.7 and Sigmas=10

Figure: Bilateral Filtering using Sigmac=0.9 and Sigmas=200

Figure: Bilateral Filtering using Sigmac=0.9 and Sigmas=100

Discussion

(b) The PSNR value when sigmac=0.7 and sigmas=0.9 is 17.9661db. **The PSNR value when sigmac=0.9 and sigmas=100 is 19.8291db.** The PSNR value when sigmac=0.9 and sigmas=200 is 19.7797db. The PSNR value when sigmac=5 and sigmas=100 is 19.821db.

Sigmac is the space parameter and sigmas is the range parameter.

From these observations we can conclude that:

- The PSNR is high when the sigmac value is around 0.9 and sigmas value is around 100. We can see that as the sigmas value increases it approaches the gaussian convolution. For sigmas=200 the PSNR value is 19.77db which is the same as the value that we obtained for gaussain filtering in the previous question.
- When sigmas is 100*sigmac, the PSNR value is high.
- As the sigmac value increases it smoothens the features and therefore the PSNR value reduces.

(c) Bilateral filtering is better than the mean and gaussian filters, because the weights of the kernel depends on the distance as well as the difference in the intensity values of the pixels. Thus, Bilateral filtering considers those neighboring pixels that have similar values and therefore preserves the edges and denoises the image. The PSNR value is 19.82db for Bilateral filtering and its performance is better than box and gaussian filter.

(c)Non-Local Means (NLM) Filtering

Window Size- 7x7

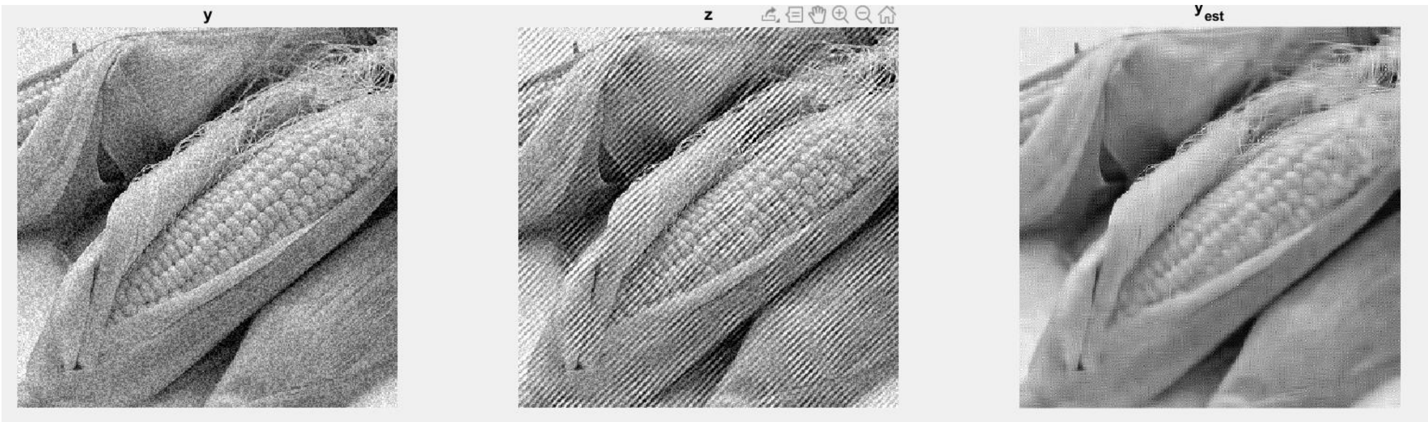
h=15 (degree of smoothing)

n1xn2= 21x21 (size for the gaussian)

(d)Block matching and 3-D (BM3D) transform filter

(1)

Experimental Result



The psnr value is 23.3366db. Thus, BM3D is the best denoising algorithm among the above algorithms. The edge details and the image information are preserved.

(e) Mixed noises in color image

(1) The noises present in the pepper image are Additive White Gaussian Noise (AWGN) and Impulse noise (salt and pepper). 'Additive' in AWGN refers to the noise being added to the signal, 'White' because the noise is uncorrelated in different spatial locations and 'Gaussian' because the probability density function is Gaussian. Gaussian noise is generally spread across the entire image, but in impulse noise only few pixels are white or black and the rest of the pixels are noise free. Impulse noise present in the pepper image is also called as the salt and pepper noise. In the image it appears as white and black dots. In salt and pepper noise, the noisy pixels either take salt value (intensity is 255) or pepper value (intensity is 0). This image as a combination of AWGN and Salt and pepper noise.

(2) In order to denoise the image consisting impulse noise, Median filter can be applied. While applying median filter on a color image, either the median filter can be applied separately on each R,G,B channel or the image can be converted to HSV color space and the median filter can be applied on Hue, Saturation and Value channels separately. Instead of applying the filter separately for R, G and B channels, a better approach would be to apply median filter on H, S, V channels individually and then convert it back to RGB. To denoise an image having gaussian noise the mean filtering, gaussian filtering, bilateral filtering, non-local means filtering or

BM3D filtering can be used. The convolution kernel generated by any of these filters should be applied separately on the R, G, B channels.

(3) Median filter can be used to remove impulse noise. Median filter is better than gaussian filter in removing impulse noise because it removes noise while preserving the edges. To denoise an image having gaussian noise the mean filtering, gaussian filtering, bilateral filtering, non-local means filtering or BM3D filtering can be used, which acts like a smoothing filter. These filters should be cascaded in a specific order. Initially apply the median filter to remove salt and pepper noise followed by any gaussian smoothing filter to remove AWGN noise.

Problem 1: Image Demosaicing and Histogram Manipulation

(a)Bilinear Demosaicing

(1) Approach and Procedure:

- Read a raw file named ‘Dog.raw’.
- The raw file consists of all the sensor values at each location of the image. It is grayscale file that has sequential bytes that correspond to each pixel in the image. The sensors are arranged in the form a Bayer array of colors. Converted the data received into a two dimensional array named ‘Imagedata[532][600][0]
- Created four masks for convolution:
  mask1[3][3] ={{0.0,1/2.0,0.0},{0.0,0.0,0.0},{0.0,1/2.0,0.0}};
  mask2[3][3] ={{0.0,0.0,0.0},{1/2.0,0.0,1/2.0},{0.0,0.0,0.0}};
  mask3[3][3] ={{0.0,1/4.0,0.0},{1/4.0,0.0,1/4.0},{0.0,1/4.0,0.0}};
  mask4[3][3]={{1/4.0,0.0,1/4.0},{0.0,0.0,0.0},{1/4.0,0.0,1/4.0}};
  By using bilinear interpolation, it averages the neighboring pixels of the same color.
- The values that exceeded 255(MAX) are assigned to 255. The pixels with intensities below 0(MIN) are made zero.

Figure: Bilinear Demosaicing

Discussion

(2) While comparing the demosaiced image with the original image Toy.ori some artifacts were identified. At the borders of the dog, the edges are distorted with red and yellow color

Bilinear demosaicing uses bilinear interpolation which finds linear interpolation in one direction followed by another direction. Thus, this might lead to wrong estimation of the direction of the edges. The green color is sampled twice the rate of blue and red pixels. The red and blue pixels are functions of the green pixels thus increasing the error rate with errors in the green pixel. The weak correlation between the three channels might also lead to artifacts. Sudden intensity changes at the edges could lead to the artifacts.

Figure: Artifacts in Bilinear

(b) Malvar-He-Cutler (MHC) Demosaicing

(1)

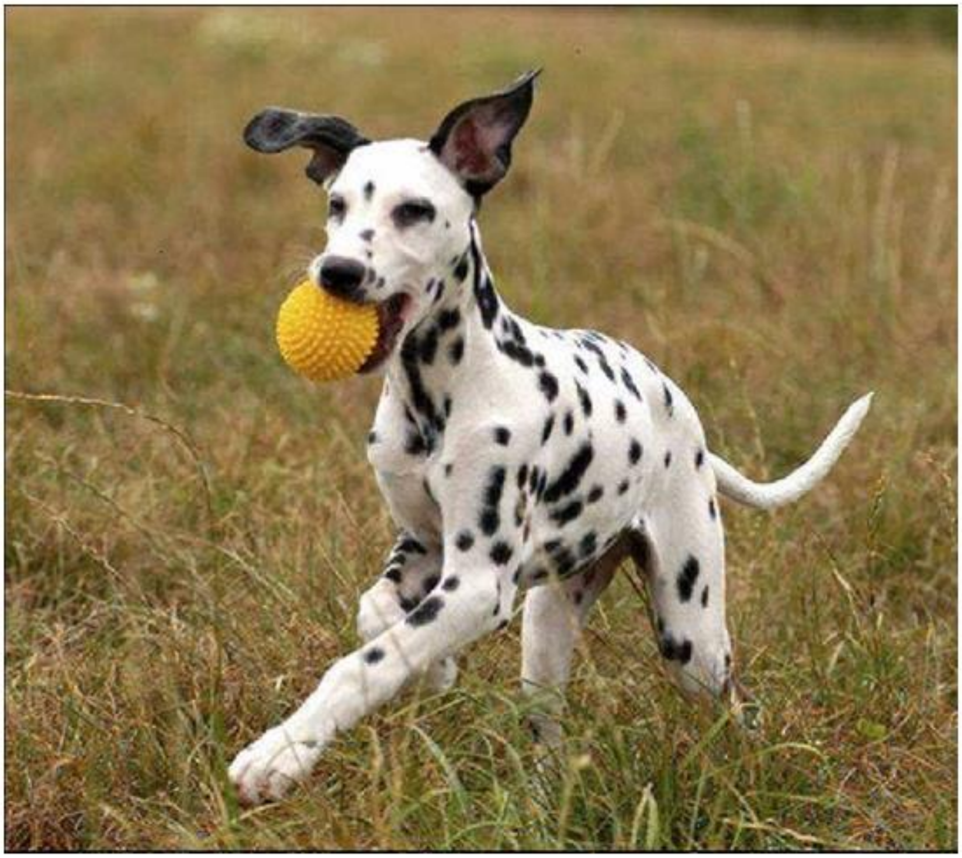
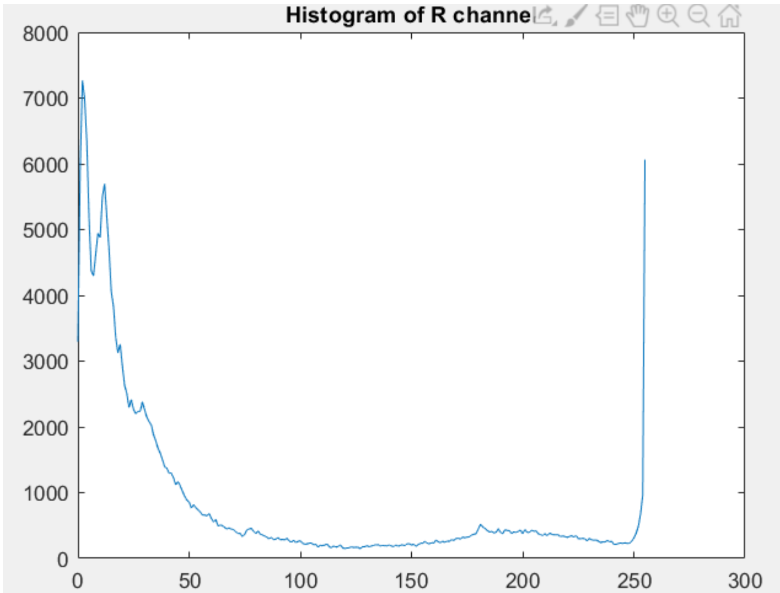


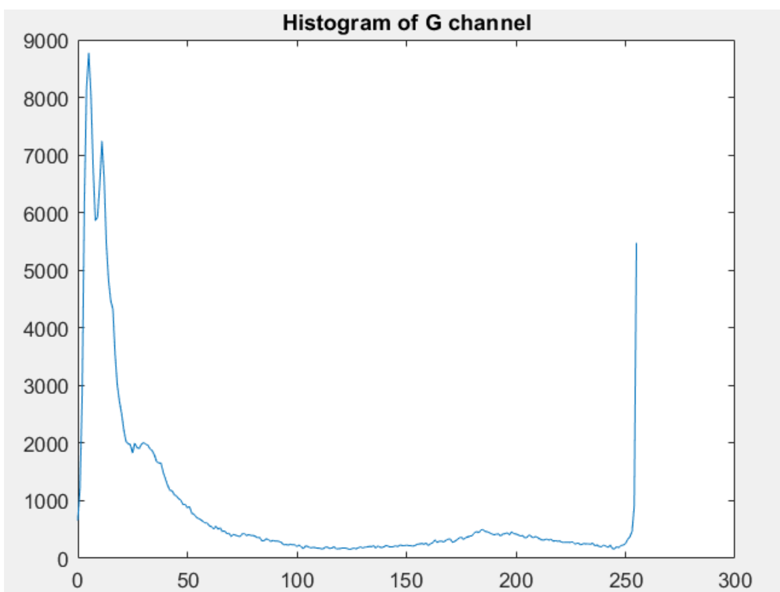
Figure: MHC Demosaicing

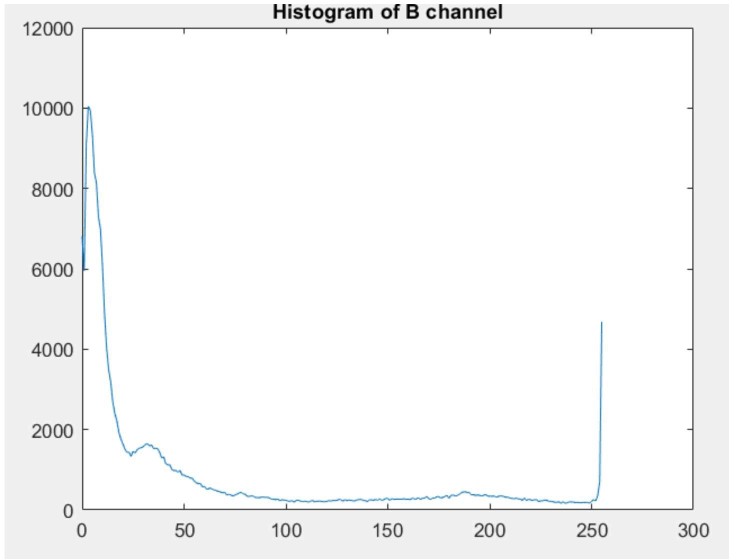
(2) The MHC result is much better than the Bilinear demosaicing result. The artifacts observed in the Bilinear are reduced in MHC. Since MHC is an improved linear interpolation demosaicing algorithm, it yields a better result. Since MHC has this additional component of second order correction term for cross channel, the results are better.

(c) Histogram Manipulation

(1)







(2)Method A

Figure: Original Image

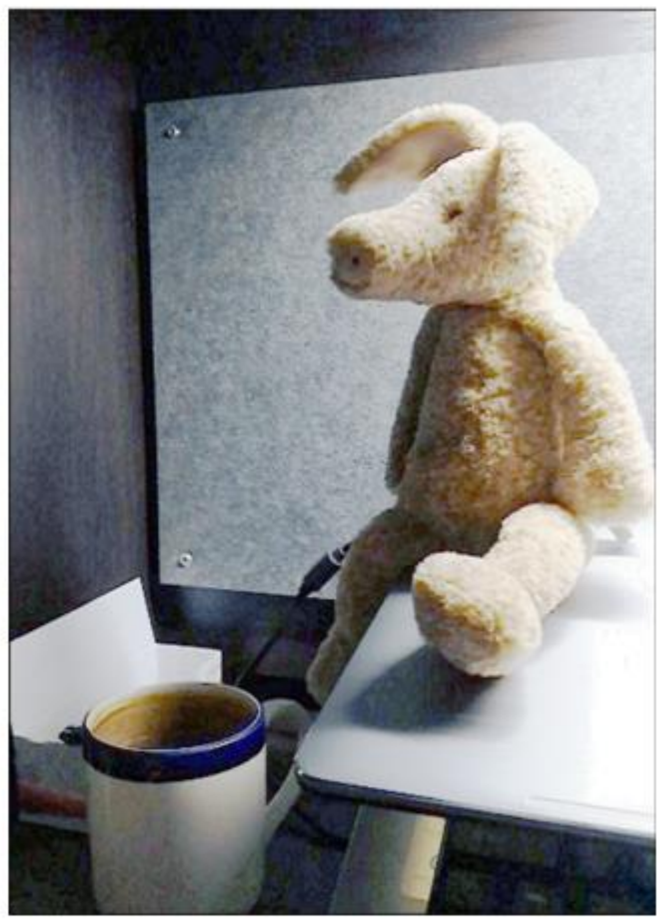


Figure: Output Image using Method A