CSE585/EE555: Digital Image Processing

Computer Project 3

Yifei Xiao, Ling Zhang, Jiaming Chai

March 26, 2020

A Objectives

We conduct the following topics in this project:

1. Learn different kinds of filters including: mean filter, median filter, alpha-trimmed mean, sigma filter and symmetric nearest-neighbor mean and implement it
2. Implement anisotropic diffusion algorithm to clean the noise for given images.

B Method

B.1 Nonlinear Filtering

We implement five different filters and apply them to the image 'disk.gif'. We analyze the gray-scale histogram, mean and standard deviation of the interior of the large disk region. The MATLAB file *Nonlinear_Filtering.m* contains all five filters. In the following sub-sections, we give more details about each filters.

B.1.1 5 × 5 mean. mean5×5.m

The mean filter replaces the central pixel in the window with the mean of its neighbours. For pixel (i, j) in image, we calculate the mean of $(i \pm 2, j \pm 2)$ and set the result to (i, j). After applying the mean filter 5 times to the image, we use `imhist` to get the gray-scale histogram. We also extract $im(100 : 150, 50 : 100)$ as our interior image. Then by applying `mean` and `std` to get the mean and standard deviation of the interior image.

B.1.2 5 × 5 median

The median filter works similar to the mean filter. It replaces the central pixel in the window with the median of its neighbours. MATLAB built in function `medfilt2` implements the median filter. We use `medfilt2(im, [5 5])` to make it a 5×5 median filter. After applying the median filter 5 times, we also use `imhist` and `mean` and `std` to get the histogram, mean and standard deviation, respectively.

B.1.3 5 × 5 alpha-trimmed mean. alpha_trim_filt5×5.m

The Alpha-Trimmed Mean Filter works as follows, we first sort each element in a window, and add all elements from $\lfloor \alpha n \rfloor + 1$ to $n - \lfloor \alpha n \rfloor$, and then multiply it with a constant coefficient. The math formula is as follows:

$$y_k = \left(\frac{1}{n - 2\lfloor \alpha n \rfloor} \right) \sum_{i=\lfloor \alpha n \rfloor + 1}^{n - \lfloor \alpha n \rfloor} x_{(i)}$$

where n is the window length, α is set to 0.25 here and $x_{(i)}$ means the i^{th} large element in the window. This filter removes highest and lowest values in a window, which are usually the noisiest pixels. MATLAB built in function `ordfilt2` implements the order statistic filter to help us find $x_{(i)}$. We use `ordfilt2(im, order(i), domain)` to find $x_{(i)}$ where `order(i)` is the i^{th} order and `domain` is a 5 $\times$ 5 array of all ones. After apply alpha-trimmed mean filter 5 times, we also use `imhist` and `mean` and `std` to get the histogram, mean and standard deviation, respectively.

B.1.4 5 $\times$ 5 sigma filter. sigma_filter5$\times$5.m

In alpha-trimmed mean filter we need to sort elements in window to implement the filter, which takes time and not very efficient. In sigma filter, we set a threshold σ so that we keep any element that the distance between it and the center is less than 2σ, otherwise we discard it. The math formula is as follows:

$$\hat{y}_k = \frac{1}{N_C} \sum_{i=-N}^{N} \delta_i x_{k-i}$$

where

$$\delta_i = \begin{cases} 1 & |x_{k-i} - x_k| \leq 2\sigma \\ 0 & \text{otherwise} \end{cases}$$

and N_C is the number of points x_{k-i} having $\delta = 1$. In MALTAB, for pixel (i,j), we check the distance between each element in $(i \pm 2, j \pm 2)$ with center. If it is less or equal to 2σ, we keep it, otherwise, discard it. Then we calculate the mean of selected elements and assign it to (i,j). After applying sigma filter 5 times, we also use `imhist` and `mean` and `std` to get the histogram, mean and standard deviation, respectively.

B.1.5 5 $\times$ 5 symmetric nearest-neighbor mean. near_neigh5$\times$5.m

The last filter we apply is the symmetric nearest-neighbor mean filter. For each element $x_{k,l}$, we choose a pair $\{x_{(k-i,l-j)}, x_{(k+i,l+j)}\}$ that elements in the pair is symmetrically opposite each other. For each element in the pair, we calculate the distance between the element and the center, then we choose the element that has the least distance. After that, we calculate the mean of selected pixels and assign it to $y_{k,l}$. In MATLAB, for pixel (k, l), we first create an empty array `min_vals`, then we create all pairs in $(k \pm 2, j \pm 2)$ and select the pixel from each pair that has the least distance to (k, l) and add that pixel to `min_vals`. After that, we calculate the mean of array `min_vals` and assign it to $y_{k,l}$. After applying symmetric nearest-neighbor mean filter 5 times, we also use `imhist` and `mean` and `std` to get the histogram, mean and standard deviation, respectively.

B.2 Anisotropic diffusion Algorithm. anisodiff.m

Anisotropic diffusion also named Perona-Malik diffusion. We implement the anisotropic diffusion equation which defined as:

$$\frac{\partial I}{\partial t} = div(c(x,y,t)\nabla I) = \nabla c \cdot \nabla I + c(x,y,t)\triangle I \tag{1}$$

Then we based on the formula which discussed in class to implement the equation (1):

$$I_{i,j}^{t+1} = I_{i,j}^{t} + \lambda[c_N^t \nabla_N I + c_S^t \nabla_S I + c_E^t \nabla_E I + c_W^t \nabla_W I]_{i,j}^t$$

Here N, S, E, W means north, south, east, west neighbors of pixel (i,j). For here:

$$\begin{aligned}
\nabla_N I_{i,j}^t &= I_{i,j+1}^t - I_{i,j}^t \\
\nabla_S I_{i,j}^t &= I_{i,j-1}^t - I_{i,j}^t \\
\nabla_E I_{i,j}^t &= I_{i+1,j}^t - I_{i,j}^t \\
\nabla_W I_{i,j}^t &= I_{i-1,j}^t - I_{i,j}^t
\end{aligned}$$

For $g(\cdot)$, we have two choices: high-contrast edges favored and wide regions favored:

$$\begin{aligned}
g(||\nabla I||) &= exp\{-||\nabla I||/k)^2\}, choice - 1 \\
g(||\nabla I||) &= \frac{1}{1 + (||\nabla I||/k)^2}, choice - 2
\end{aligned}$$

The two options in our code are all included in one function. To make easier switch we put these two options as number for a parameter so that we can simply change the parameters for either high-contrast edges of wide-regions without modifying the code.

C Result

C.1 Nonlinear Filtering

We show the results of all five filters that we implemented and the corresponding gray-scale histogram and the mean and standard deviation of interior image. We choose row 100 to 150 and col 50 to 100 (`im(100:150, 50:100)`) as interior image. Figure 1 and Figure 2 shows the original image and its gray-scale histogram. Table 1 shows the mean and standard deviation of interior image.

mean	197.51
std	31.34

Table 1: Mean and Std of interior of original image.

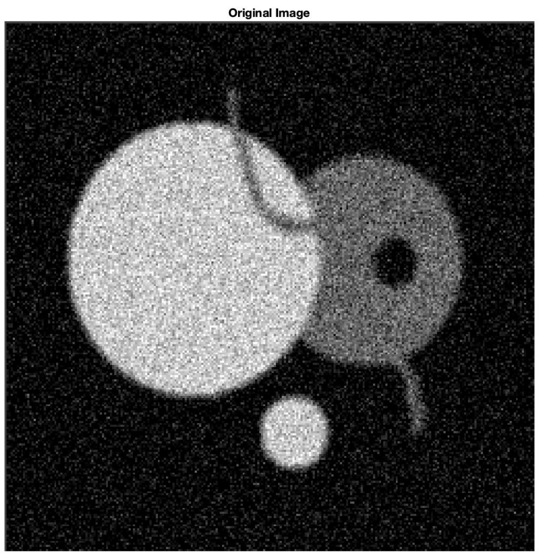

Figure 1: Original Image.

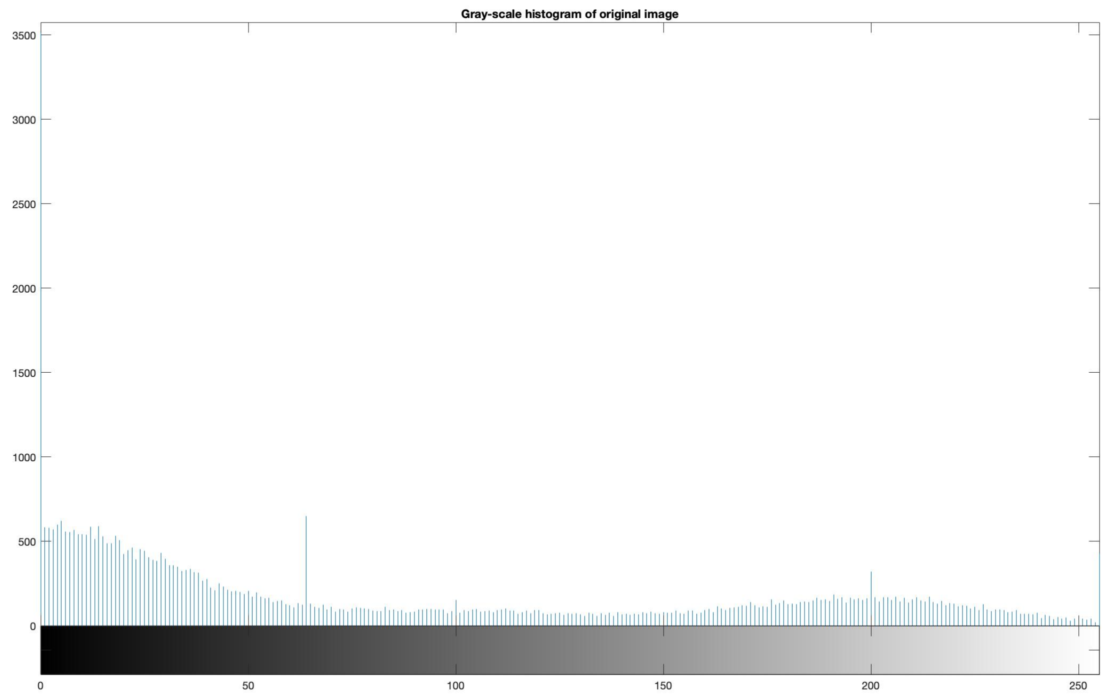

Figure 2: Gray-scale histogram of original image.

C.1.1 5 $\times$ 5 mean

Figure 3 shows the image after applying mean filter for once. Figure 4 and Figure 5 shows the result after applying five times. Table 2 shows the mean of standard deviation of interior image. As shown in figures and the table, the standard deviation decreases while the mean keeps the same. But edges are blurred after applying mean filter.

mean	200.08
std	2.48

Table 2: Mean and Std of interior after applying mean filter five times.

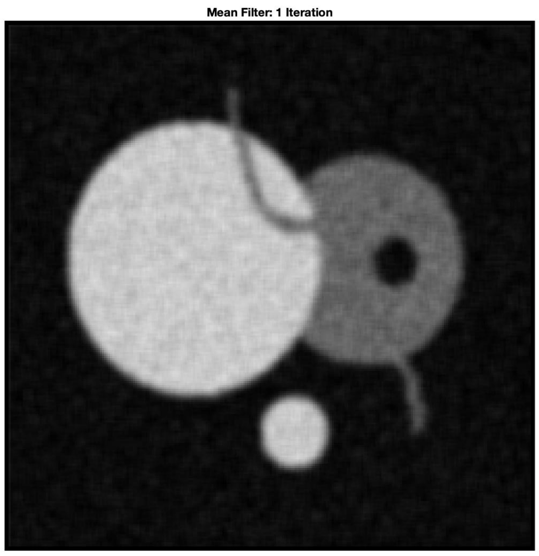

Figure 3: Image after applying mean filter once.

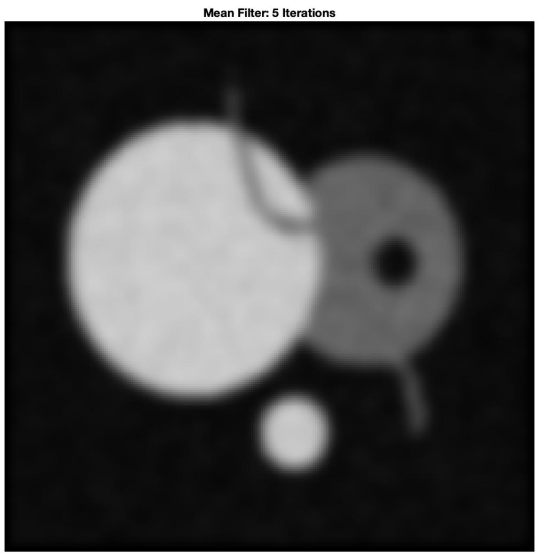

Figure 4: Image after applying mean filter five times.

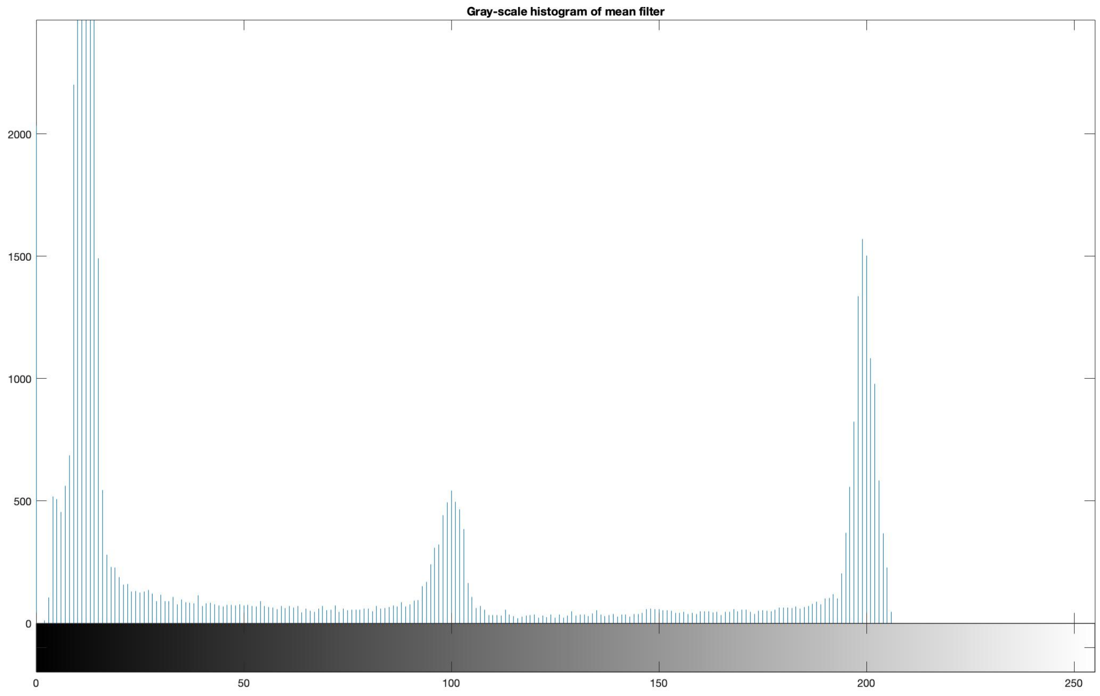

Figure 5: Gray-scale histogram after applying mean filter five times.

C.1.2 5 $\times$ 5 median

Figure 6 shows the image after applying median filter for once. Figure 7 and Figure 8 shows the result after applying five times. Table 3 shows the mean of standard deviation of interior image. As shown in figures and the table, the standard deviation decreases while the mean keeps the same. Edges are not blurred, it is because the outputs of median filter are values that are already in original image.

mean	200.39
std	3.45

Table 3: Mean and Std of interior after applying median filter five times.

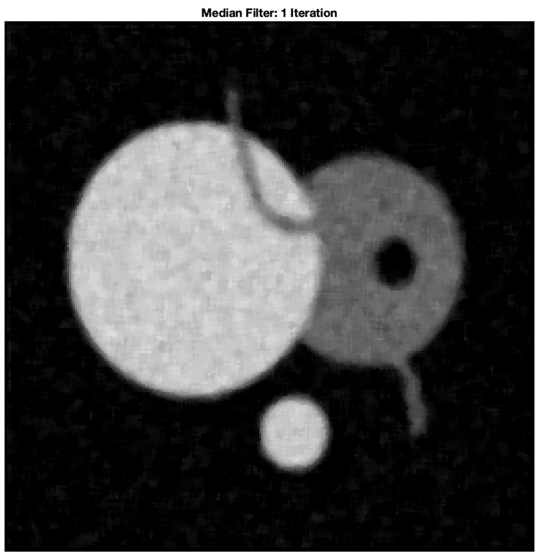

Figure 6: Image after applying median filter once.

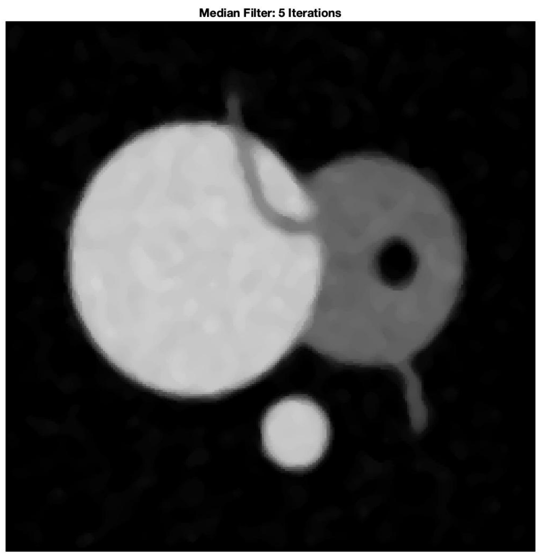

Figure 7: Image after applying median filter five times.

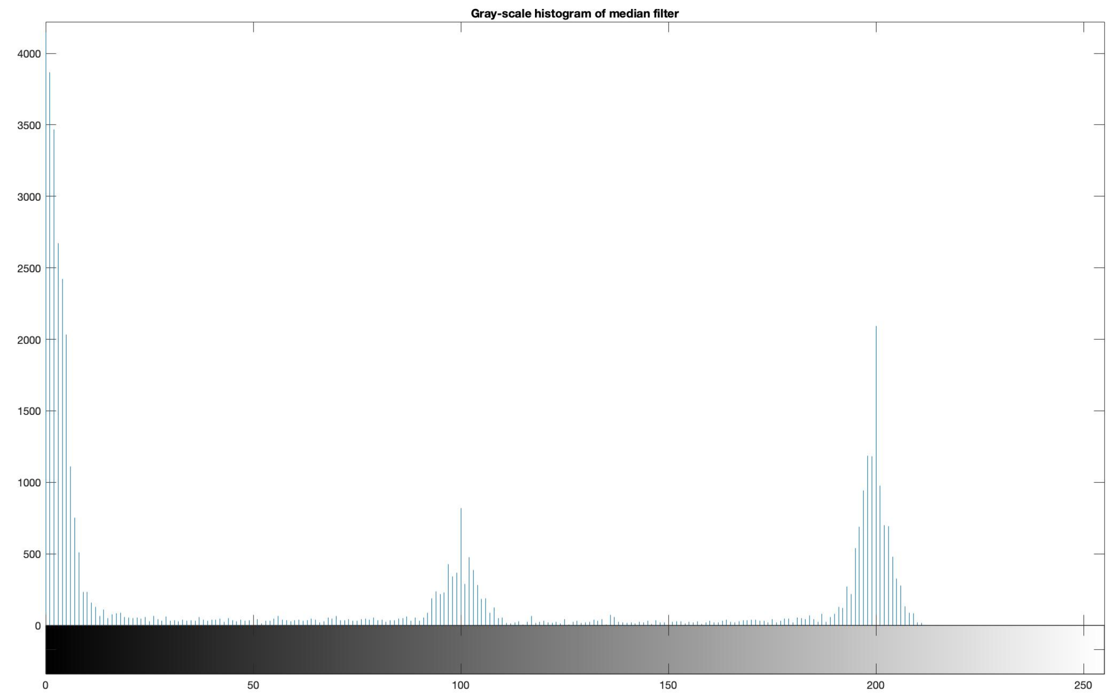

Figure 8: Gray-scale histogram after applying median filter five times.

C.1.3 5 $\times$ 5 alpha-trimmed mean

Figure 9 shows the image after applying alpha-trimmed mean filter for once. Figure 10 and Figure 11 shows the result after applying five times. Table 4 shows the mean of standard deviation of interior image. As shown in figures and the table, the standard deviation decreases while the mean keeps the same. Edges are blurred but not as the mean filter, this is because we discard highest and lowest values.

mean	200.30
std	3.03

Table 4: Mean and Std of interior after applying alpha-trimmed mean filter five times.

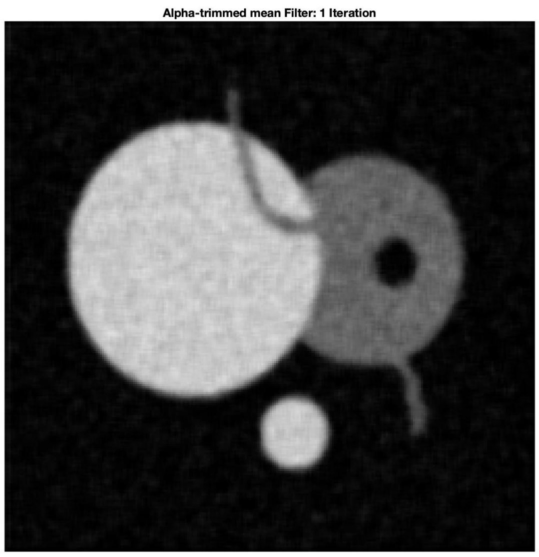

Figure 9: Image after applying alpha-trimmed mean filter once.

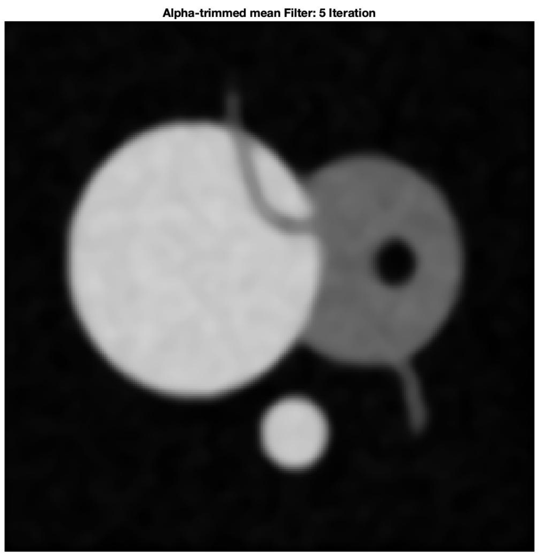

Figure 10: Image after applying alpha-trimmed mean filter five times.

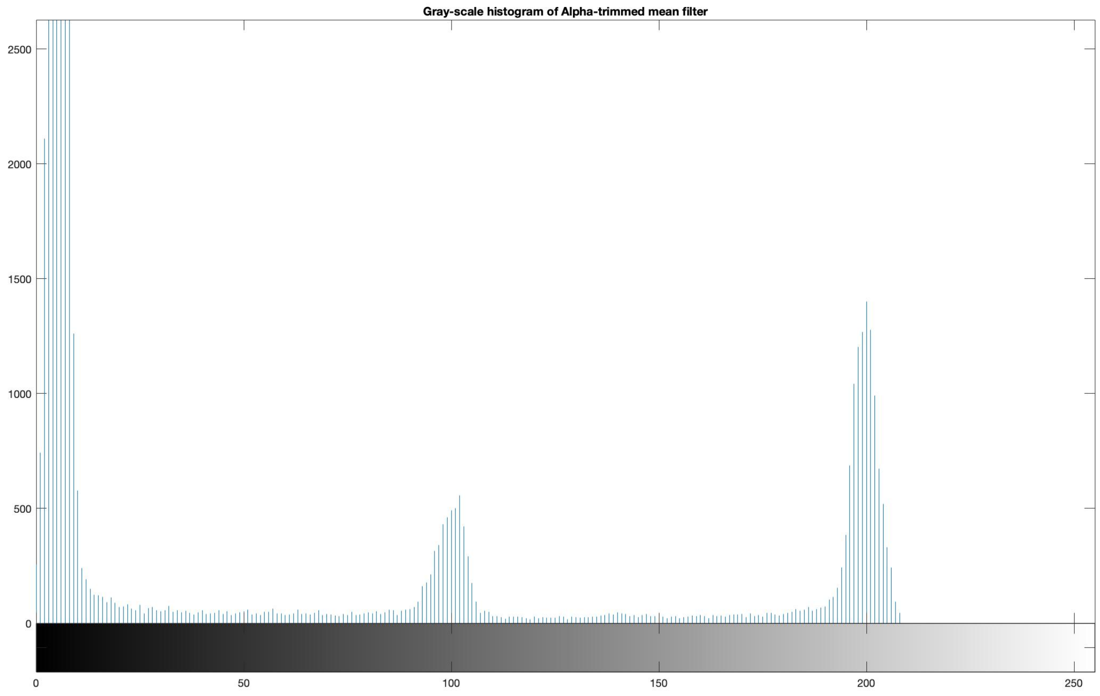

Figure 11: Gray-scale histogram after applying alpha-trimmed mean filter five times.

C.1.4 5 $\times$ 5 sigma filter

Figure 12 shows the image after applying sigma filter for once. Figure 13 and Figure 14 shows the result after applying five times. Table 5 shows the mean of standard deviation of interior image. As shown in figures and the table, the standard deviation decreases while the mean keeps the same. Edges are not blurred as alpha-trimmed mean filter.

mean	191.06
std	5.18

Table 5: Mean and Std of interior after applying sigma filter five times.

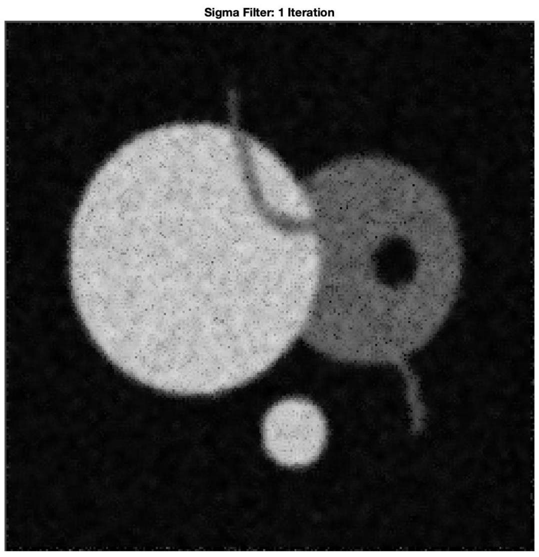

Figure 12: Image after applying sigma filter once.

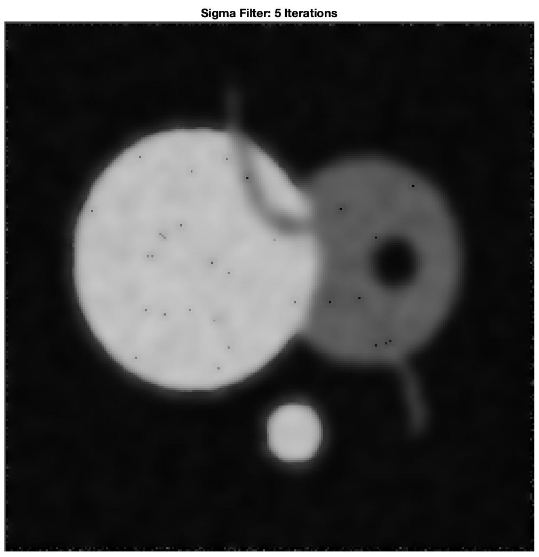

Figure 13: Image after applying sigma filter five times.

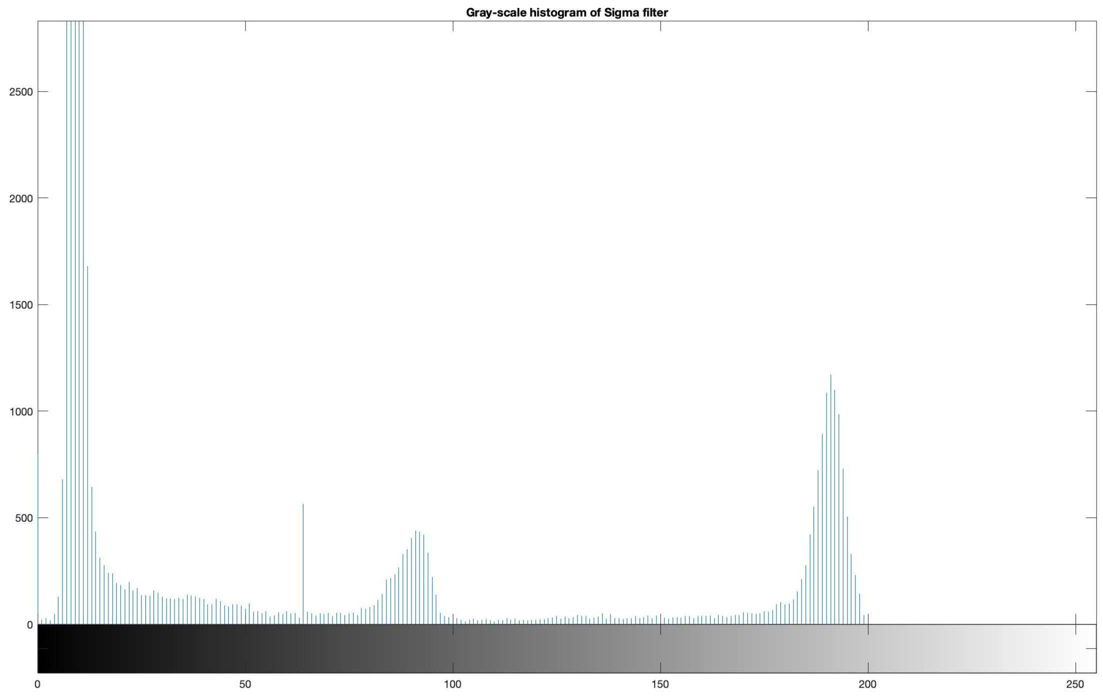

Figure 14: Gray-scale histogram after applying sigma filter five times.

C.1.5 5 $\times$ 5 symmetric nearest-neighbor mean

Figure 15 shows the image after applying symmetric nearest-neighbor mean filter for once. Figure 16 and Figure 17 shows the result after applying five times. Table 6 shows the mean of standard deviation of interior image. As shown in figures and the table, the standard deviation decreases while the mean keeps the same. The filtered image gets much sharper than the original image.

mean	181.32
std	4.33

Table 6: Mean and Std of interior after applying symmetric nearest-neighbor mean filter five times.

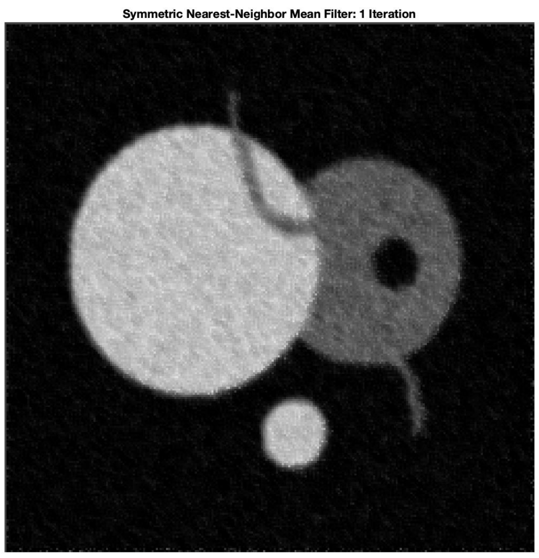

Figure 15: Image after applying symmetric nearest-neighbor mean filter once.

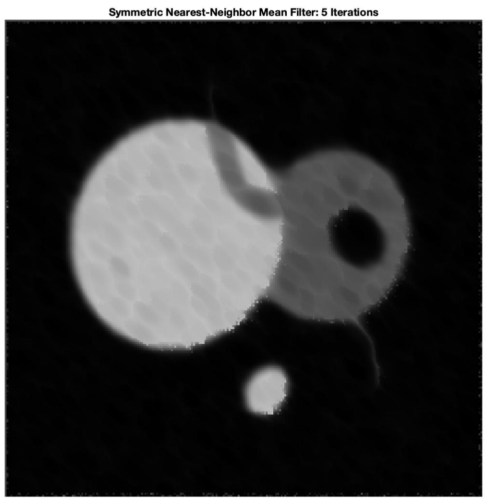

Figure 16: Image after applying symmetric nearest-neighbor mean filter five times.

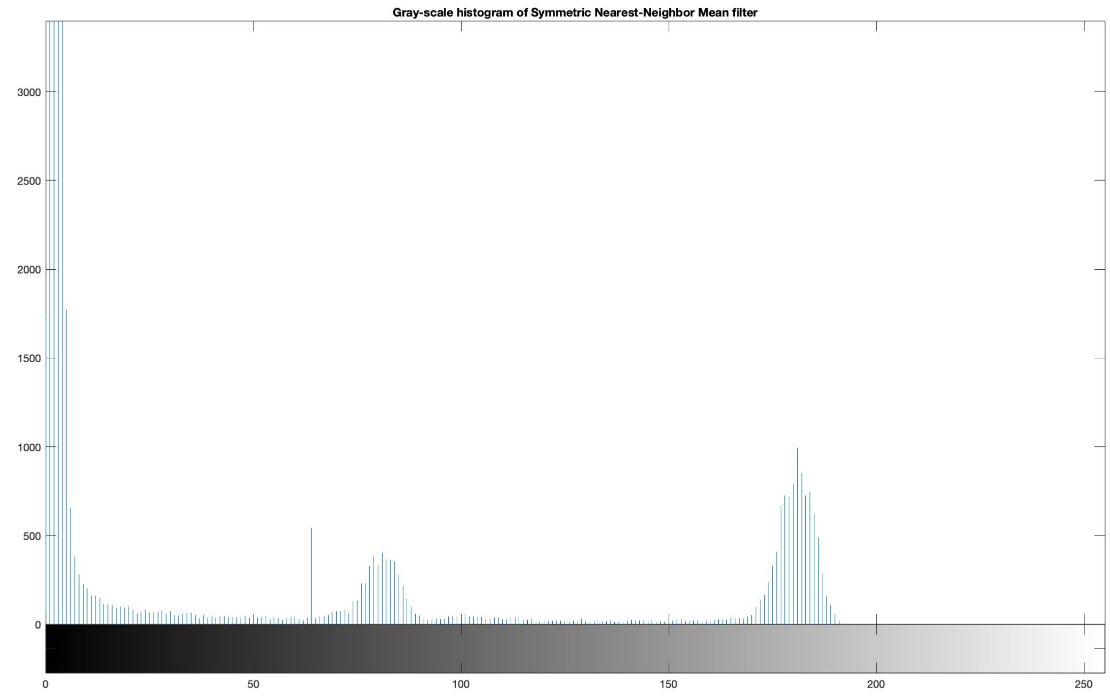

Figure 17: Gray-scale histogram after applying symmetric nearest-neighbor mean filter five times.

C.2 Anisotropic Diffusion Algorithm

We show that two images using our implemented algorithm based on the methods we illustrate in the previous section. As stated in the handout, $\lambda = 0.25$ and we try different K during our experiments. Two options are two forms of $g(\cdot)$ as stated in the previous section. During different testing of K, we finally choose $K = 40$ in the final decision. The results for each experiments we conducted as shown in the order of first option with iteration 0, 5, 20, 100.

C.2.1 Wheel Noise

Specifically, there are four parts for each run. We put the filtered images on top and the plots and histogram stacked below for better observation on the changes for latter discussion.

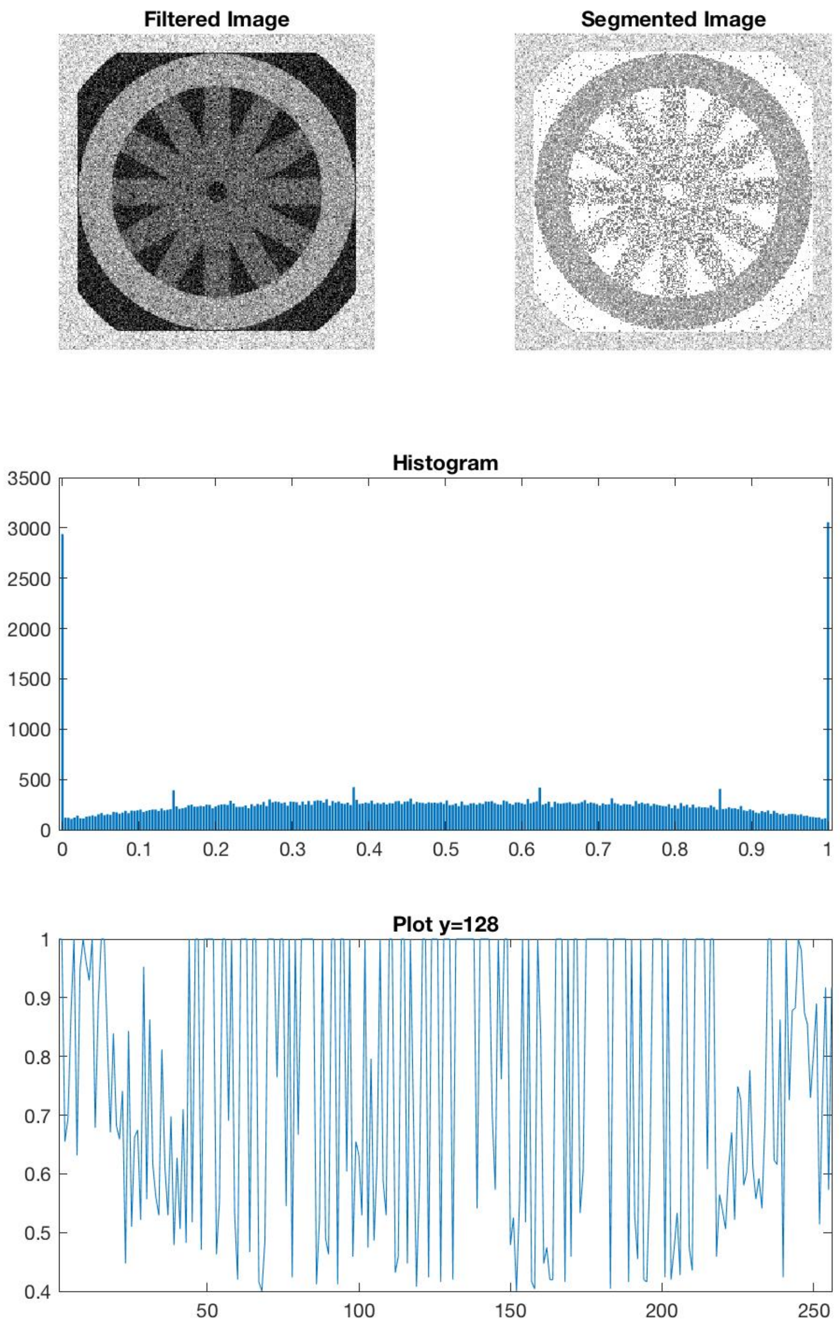

Figure 18: The filtered image at iteration 0 using first form of $g(\cdot)$, segmented image, gray-scale histogram, and the plot of the line with pixel value.

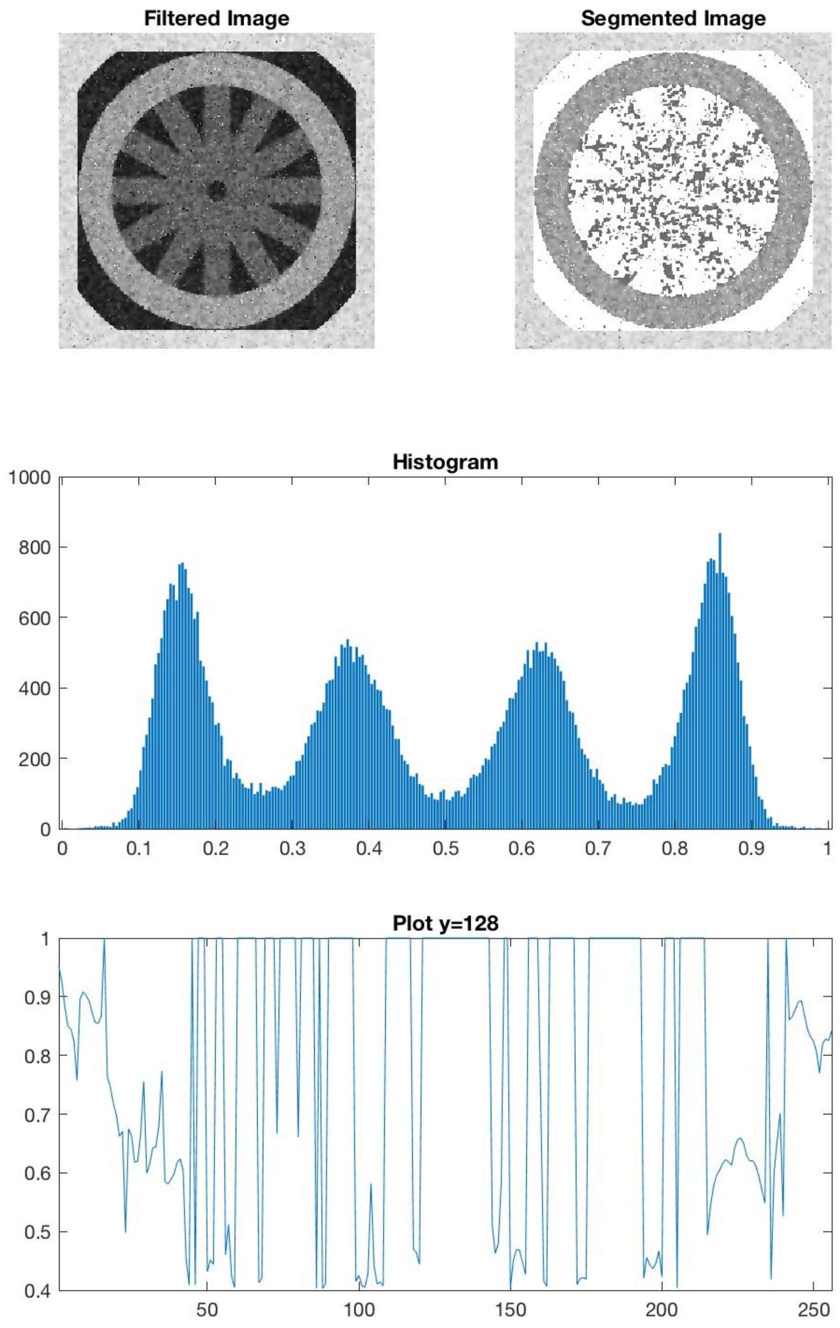

Figure 19: The filtered image at iteration 5 using first form of $g(\cdot)$, segmented image, gray-scale histogram, and the plot of the line with pixel value.

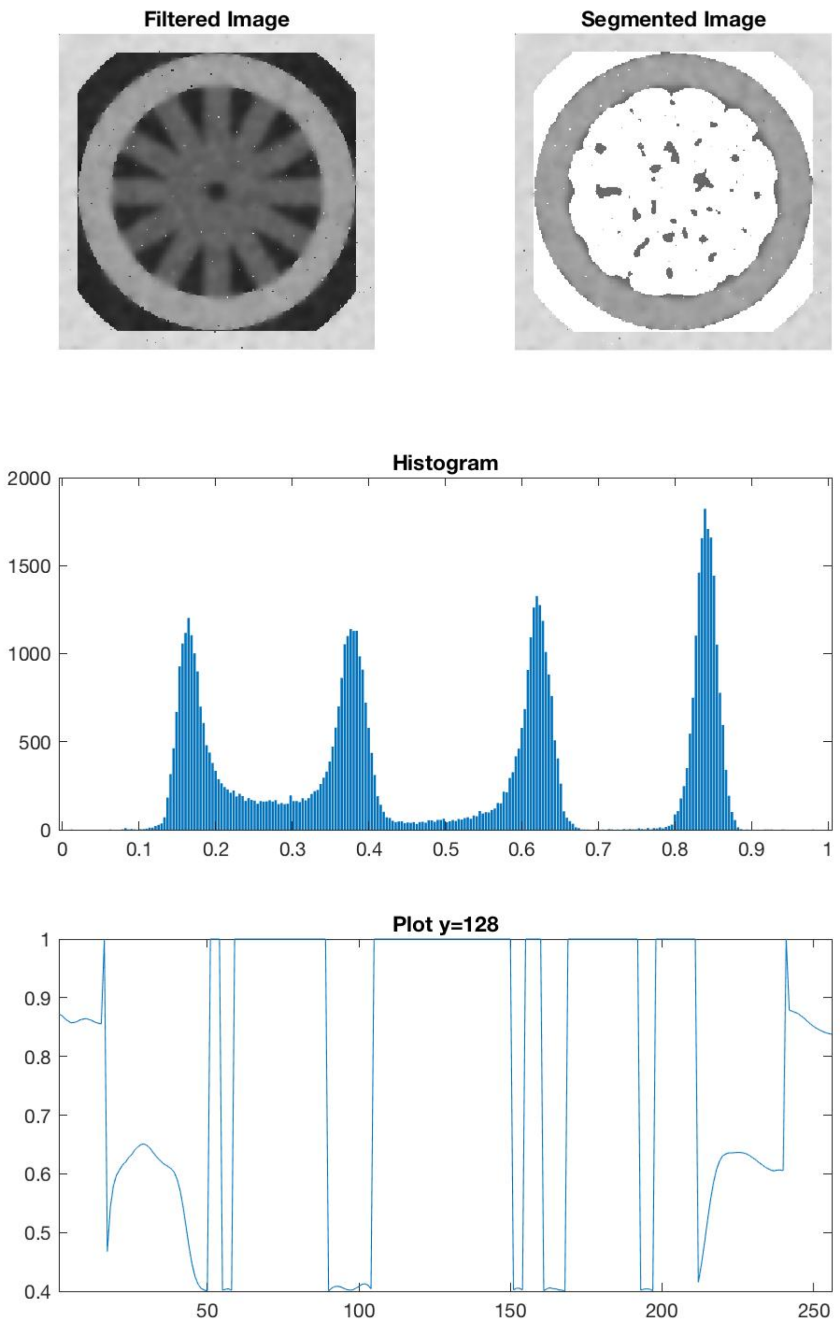

Figure 20: The filtered image at iteration 20 using first form of $g(\cdot)$, segmented image, gray-scale histogram, and the plot of the line with pixel value.

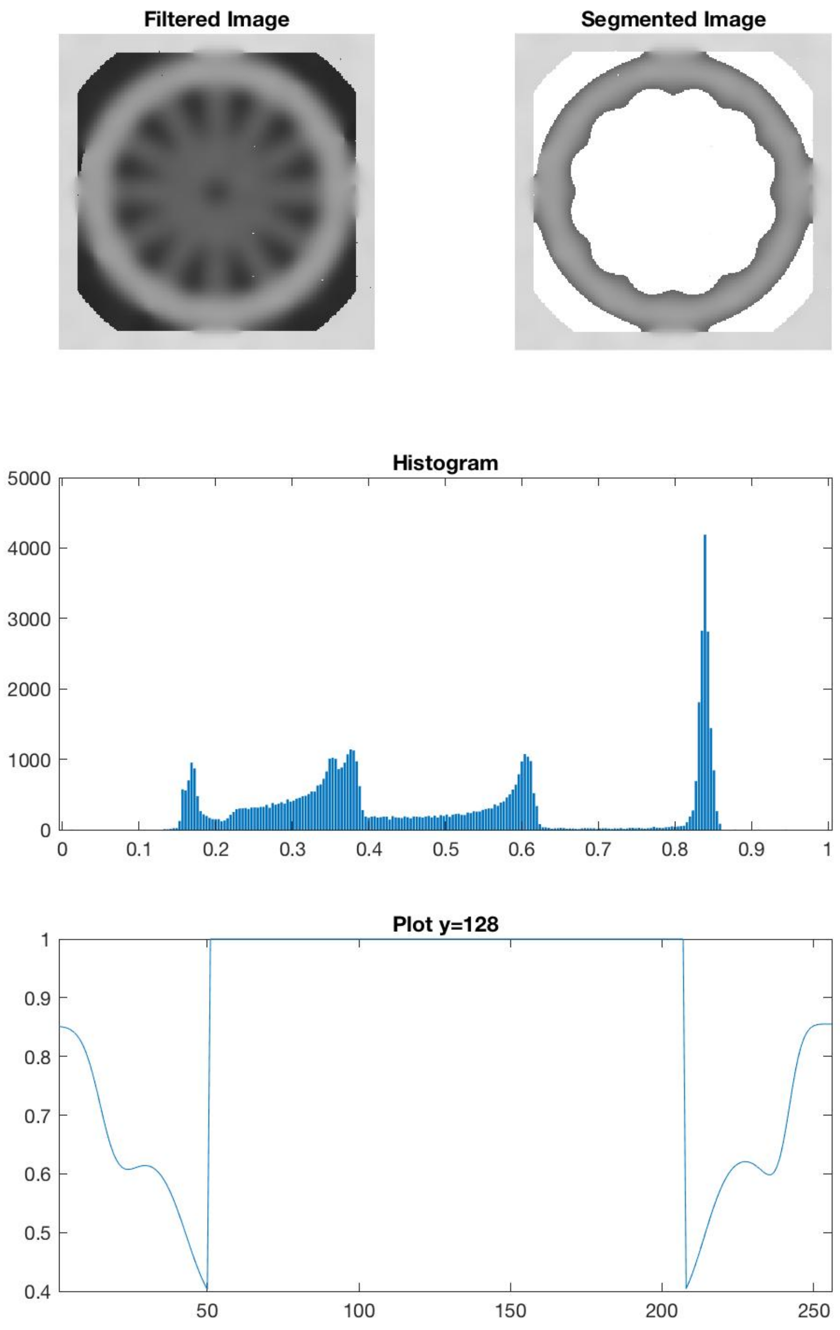

Figure 21: The filtered image at iteration 100 using first form of $g(\cdot)$, segmented image, gray-scale histogram, and the plot of the line with pixel value.

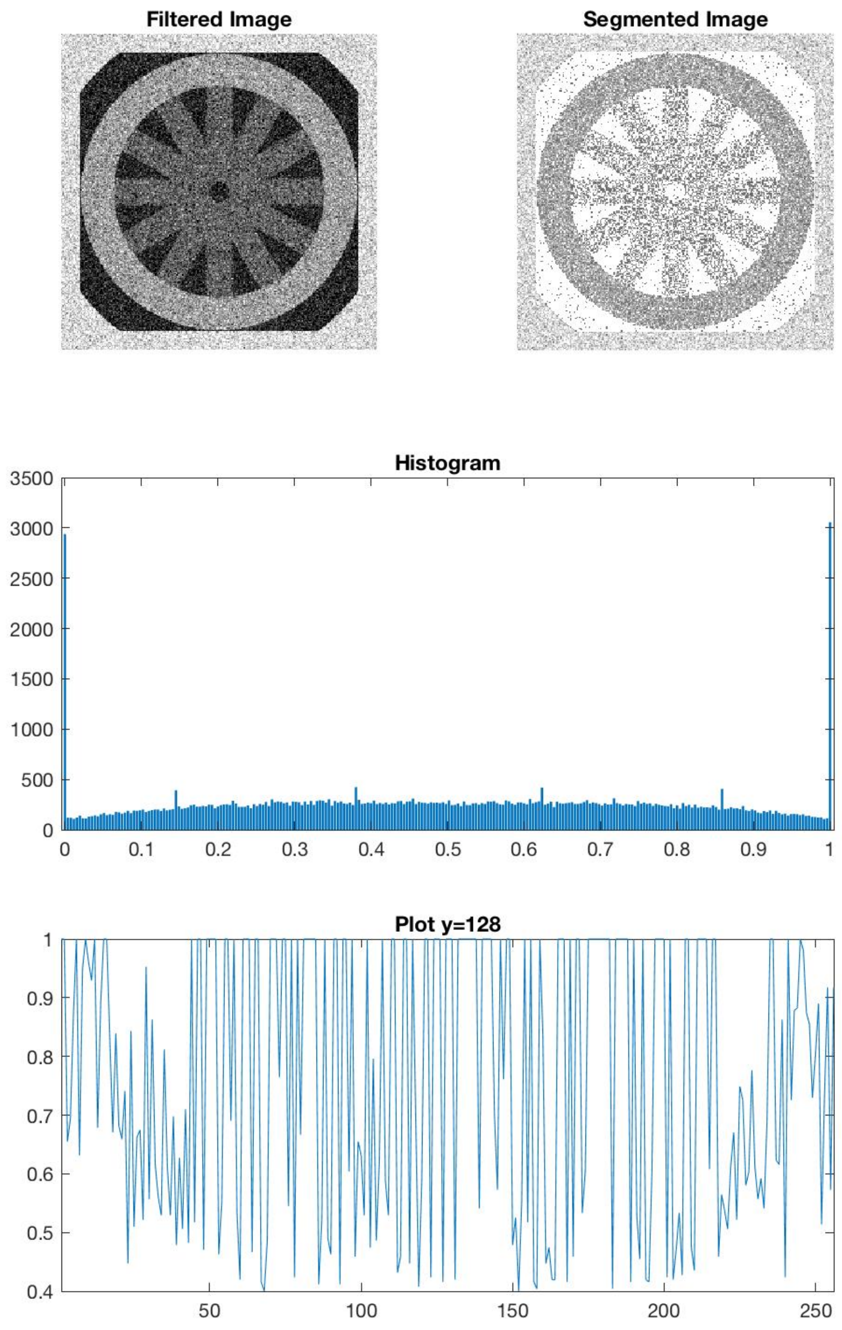

Figure 22: The filtered image at iteration 0 using second form of $g(\cdot)$, segmented image, gray-scale histogram, and the plot of the line with pixel value.

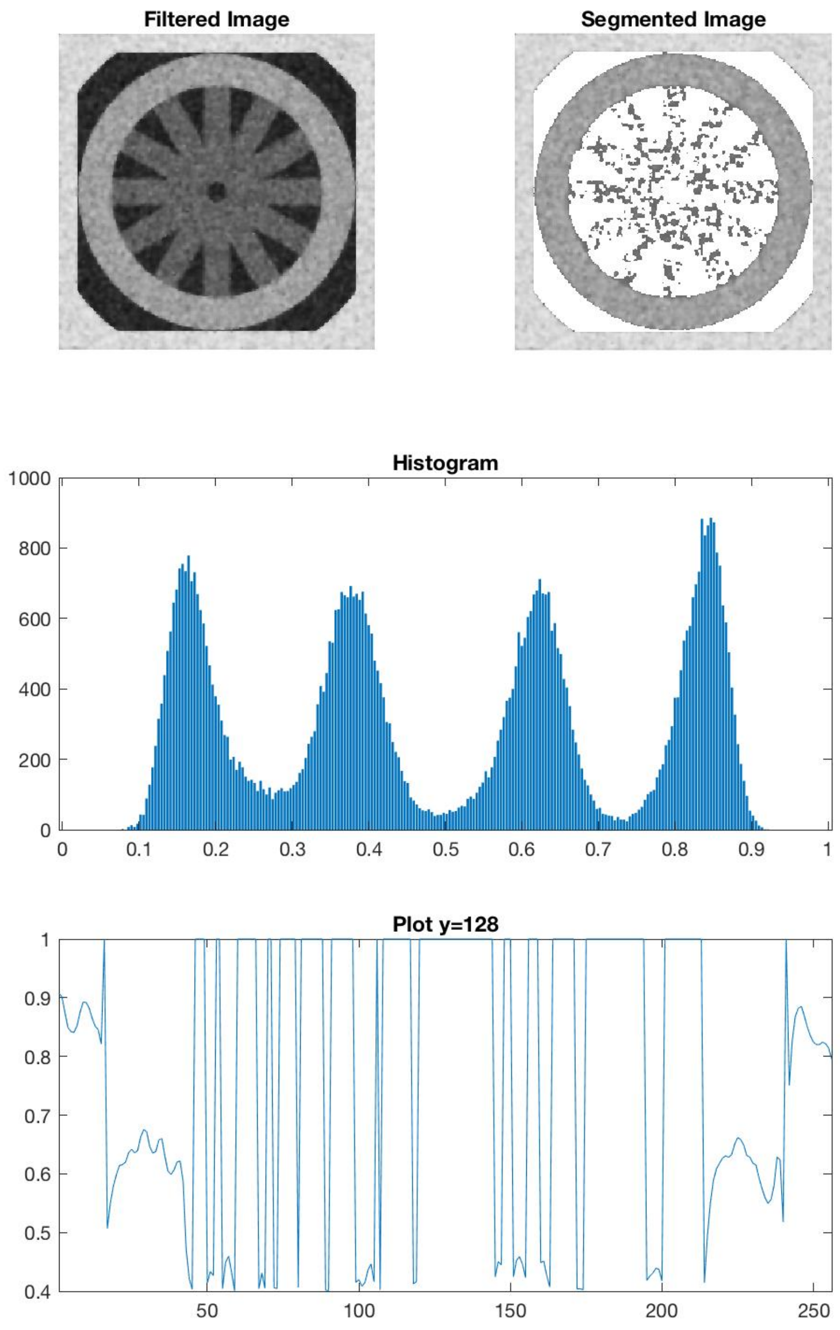

Figure 23: The filtered image at iteration 5 using second form of $g(\cdot)$, segmented image, gray-scale histogram, and the plot of the line with pixel value.

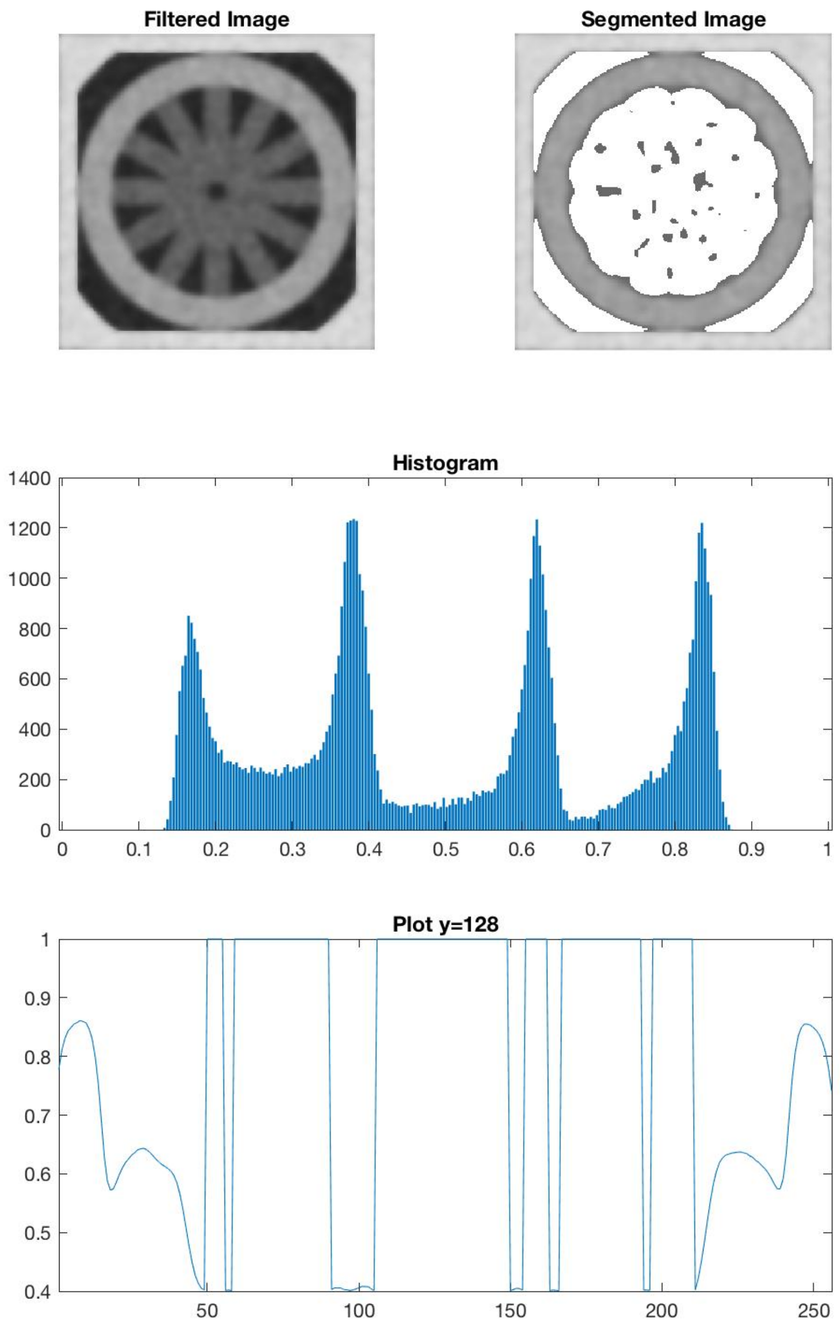

Figure 24: The filtered image at iteration 20 using second form of $g(\cdot)$, segmented image, gray-scale histogram, and the plot of the line with pixel value.

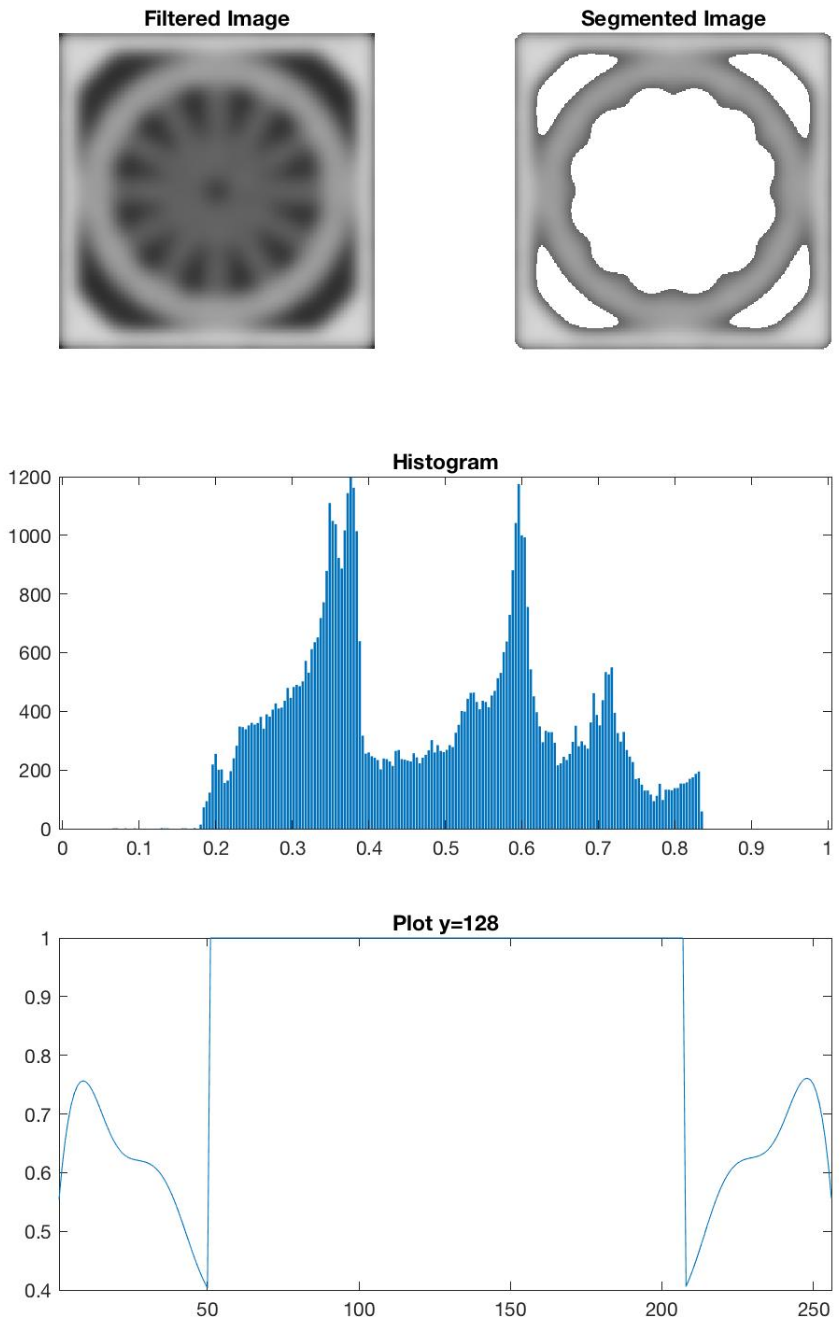

Figure 25: The filtered image at iteration 100 using second form of $g(\cdot)$, segmented image, gray-scale histogram, and the plot of the line with pixel value.

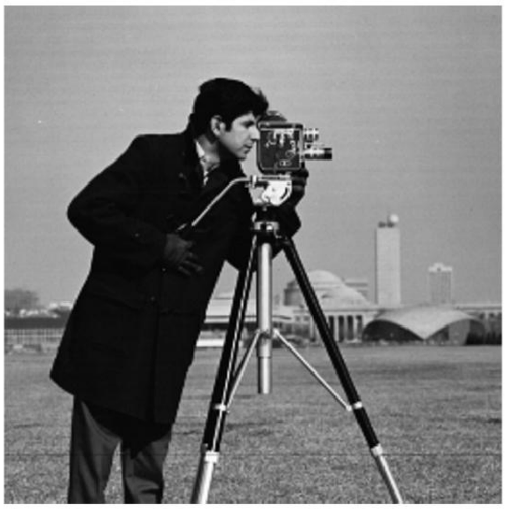

Figure 26: The filtered image at iteration 0 using first form of $g(\cdot)$

C.2.2 Camera Man

For this image there is only the filtered image for the final result. The parameters we use are exactly the same as used for wheel noise.

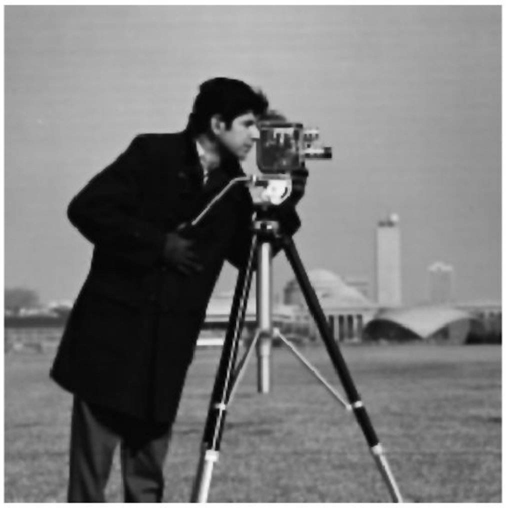

Figure 27: The filtered image at iteration 5 using first form of $g(\cdot)$.

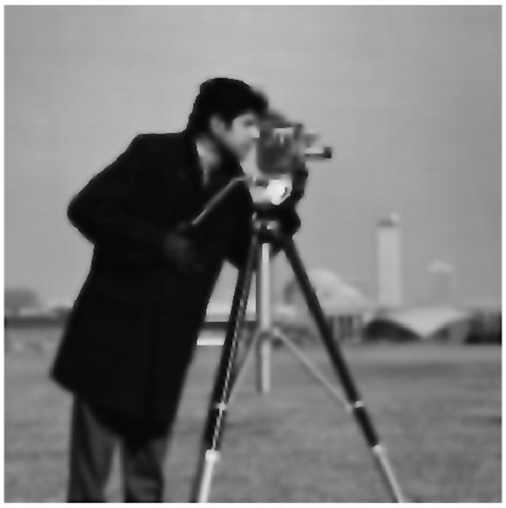

Figure 28: The filtered image at iteration 20 using first form of $g(\cdot)$.

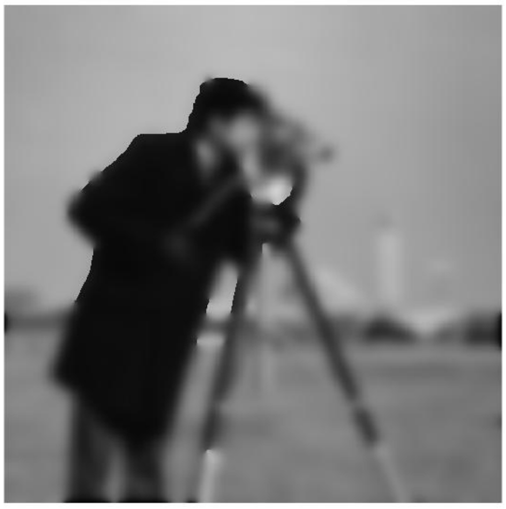

Figure 29: The filtered image at iteration 100 using first form of $g(\cdot)$.

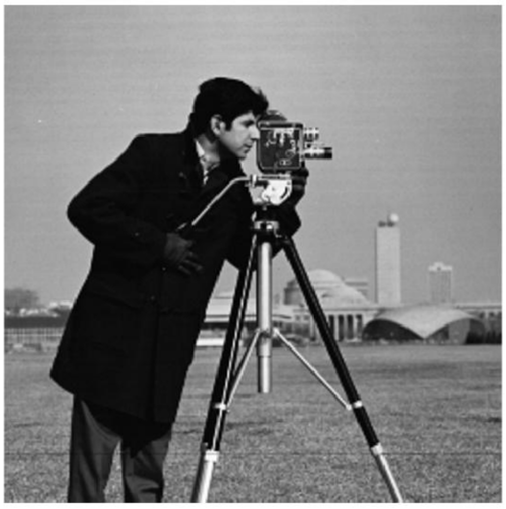

Figure 30: The filtered image at iteration 0 using second form of $g(\cdot)$

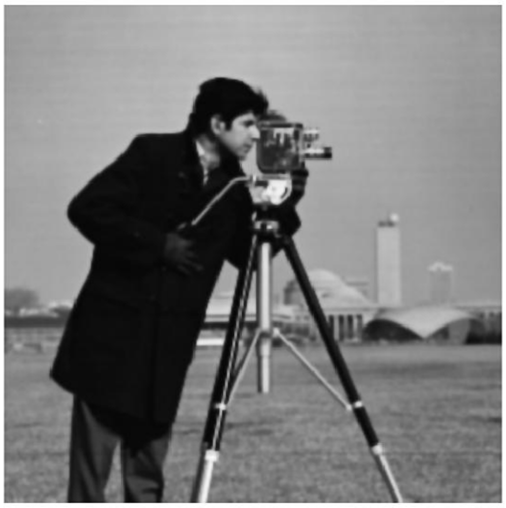

Figure 31: The filtered image at iteration 5 using second form of $g(\cdot)$.

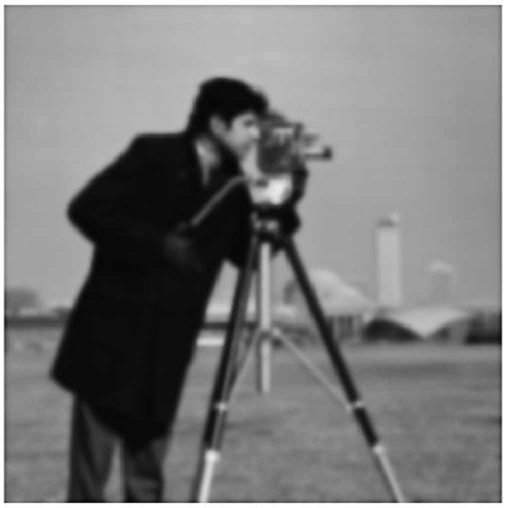

Figure 32: The filtered image at iteration 20 using second form of $g(\cdot)$.

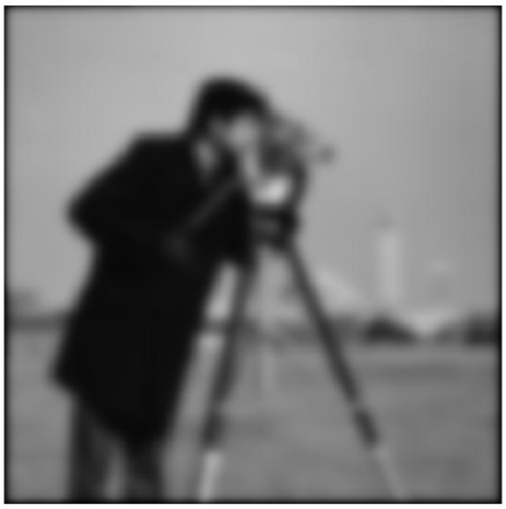

Figure 33: The filtered image at iteration 100 using second form of $g(\cdot)$.

C.3 Discussion for Anisotropic Diffusion

The following discussion follows the order stated in the handout.

The result significantly changes with the number of iterations. The most strong observation is that as the number of iterations goes up, the image loses noises. You can see from the plot graph for wheel noise. Since the we pick the middle point as the plot, the "contour" of the plot becomes clearer, more of a symmetrical shape. This indication gives us belief the more iterations helps the noise reduction. Another viewpoint is that the histogram for the original image is a uniform shape, and that is because the "noise" pixels are uniformly distributed throughout the original image. During our experiment, K is the factor that adjust how blur an image is. The large K we set, the images looks more of smoothness. This smoothness can be good and bad, since we still need to keep the shape of the object while reducing some noises within the object.

There are two forms of $g(\cdot)$. The second form gives strong dark coloring compared with the first one. As shown in the wheel noise, the segmented for the second form looks much darker on the border side, thus leaving the outer border looks like it is part of the object, compared with the first form. Similarly with the histogram, the volume for the second for is much more than the volume shown in the first form. One little detail we also noticed in the filtered image is that for the second form the blurred area includes the whole images, where the first form only leaves the "spokes" blurred.

For the cameraman, the image is more complex, so the noise is hard to be captured even for humans. The final result of the cameraman is more of a blurred version of the original image, as the algorithm hardly captures the noise while including the object as part of the noise.

D Conclusions

In this project, we learned different kinds of filters and know how do they works. Also we learned anisotropic diffusion algorithm. We successfully done every task and got reasonable results.