Homework 4

13331314 叶嘉祺

1 Exercises

1.1 Color Spaces (10 Points)

1. Give one example where the HSI color space is advantageous to the RGB color space. Also give one example where RGB is advantageous to HSI. (5 Points)

Example where the HSI color space is advantageous to the RGB color space:
The HSI model fully reflects the basic attributes of human perception of color, and the results of human perception of color one one, therefore, the HSI model is widely used in human visual system for image representation and processing system. For computer vision technology, using HSI is better than using RGB.

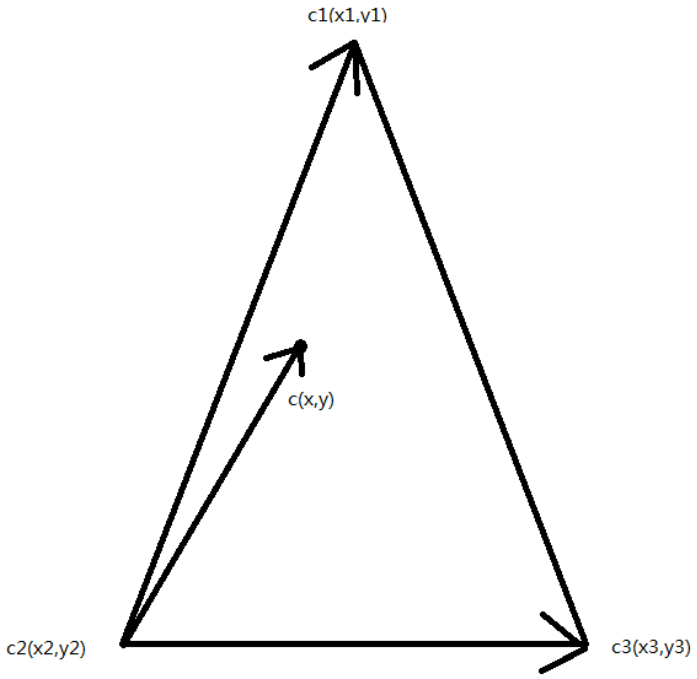
Example where RGB is advantageous to HSI:
RGB is a form of representation for computer information. If we want to display an image on the screen, using RGB is much advantageous.

Conclusion: RGB and HSI are different representations of the same physical quantity. It depends when we are going to use which of them.

2. What is the effect of adding 60◦ to the Hue components on the R, G, and B components of a given image? (5 Points)
It's equivalent to the RGB representation of the three-dimensional space in which the H plane is rotated by 60 degrees.
Acording to the equations in the text book.
If H' < 120: R decrease, G increase, B not effect, the overall tone tends to be yellow.
If H' > 120 && H < 240: R not effect, G increase, B decrease, the overall tone tends to be green.
If H' > 240 && H < 360: R decrease, G not effect, B increase, the overall tone tends to be green.

1.2 Color Composition (10 Points)

Consider any three valid colors c1 , c2 and c3 with coordinates (x 1 , y 1), (x 2 , y 2) and (x 3 , y 3), in the chromaticity diagram in Fig. 1 (or Fig. 6.5 of the textbook). Derive the general expression(s) for computing the relative percentages of c1 , c2 and c3 composing a given color that is known to lie within the triangle whose vertices are at the coordinates of c 1 , c 2 and c 3 .



Use vecotor in plane geometry:

v1 = c1 - c2 = (x1-x2,y1-y2)
v2 = c3 - c2 = (x3-x2, y3-y2)
v3 = c - c2 = (x - x2, y - y2)

According to the basic theorem of plane vector:

v3 = av1 + bv2 = av1 + (-a-b)v2 + bv3

We can calculate:

$$a = \frac{x(y3 - y2) - y(x3 - x2)}{(x1 - x2)(y3 - y2) - (y1 - y2)(x3 - x2)}$$

$$b = -\frac{x(y1 - y2) - y(x1 - x2)}{(x1 - x2)(y3 - y2) - (y1 - y2)(x3 - x2)}$$

$$c = p1 \cdot c1 + p2 \cdot c2 + p3 \cdot c3$$

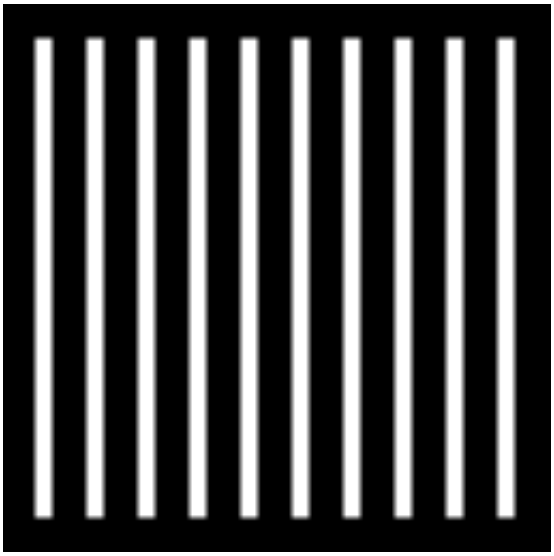
$$p1 = a, p2 = -a - b, p3 = b$$

2 Programming Tasks

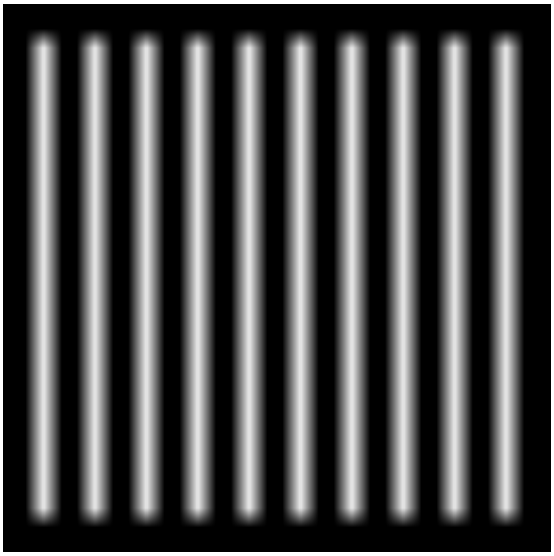
2.2 Image Filtering (10 Points)

Target The white bars in Fig. 2 are 8 pixels wide and 224 pixels high. The separation between bars is 16 pixels. For other details please refer to your input image.
Finish the following applications (reuse the function "filter2d" in HW2 if you like):

1. Filter your input image with 3 × 3 and 9 × 9 arithmetic mean filters respectively. Paste your two results on the report. Also briefly describe what each result looks like, e.g. the width/height/color of bars in the report. (2 Points)
3x3 filter



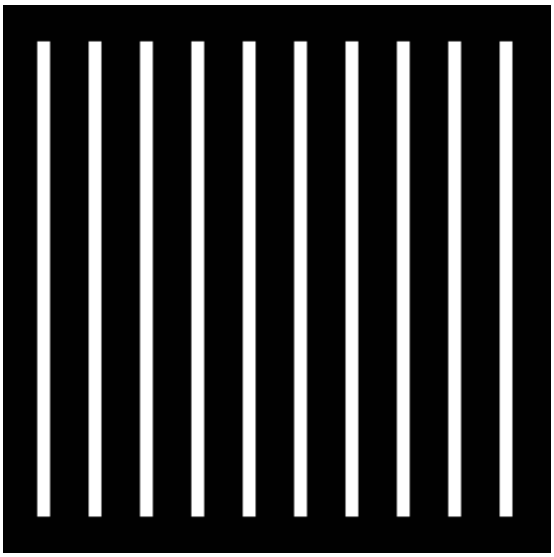
9x9 filter



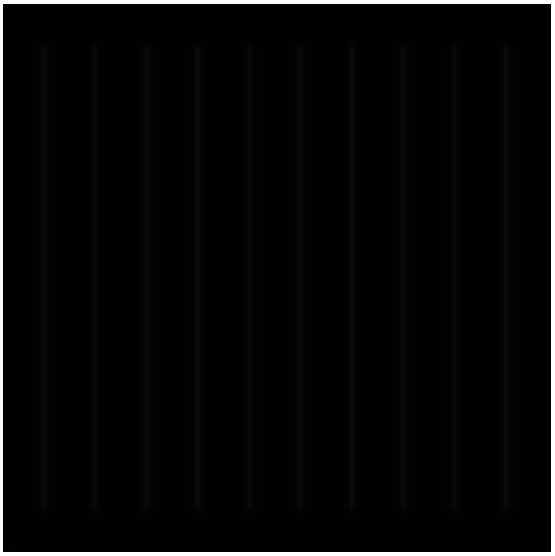
The width and the height of the bar decreases. Also the image becomes blurred.

2. Repeat the first application with 3 × 3 and 9 × 9 harmonic mean filters. Paste your results in the report and briefly describe what each result looks like. (4 Points)

3x3 filter



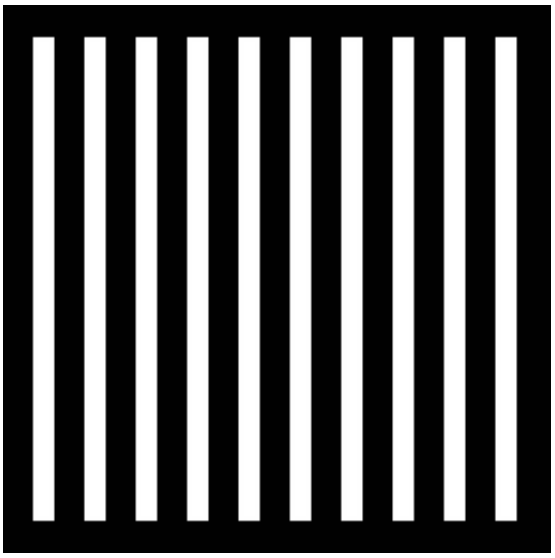
9x9 filter



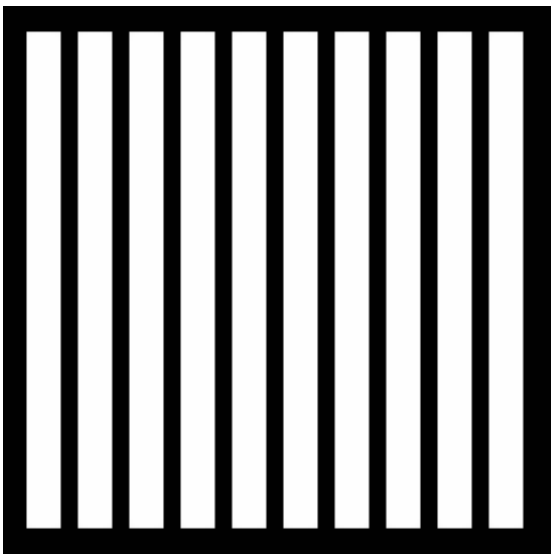
The width and the height of the bar decreases. The second image narrowing to black.

3. Repeat the first application with 3 × 3 and 9 × 9 contraharmonic mean filters with Q = −1.5. Paste your results in the report and briefly describe what each result looks like. (4 Points)

3x3 filter



9x9 filter



The width and the height of the bar increases. The bar broaden.

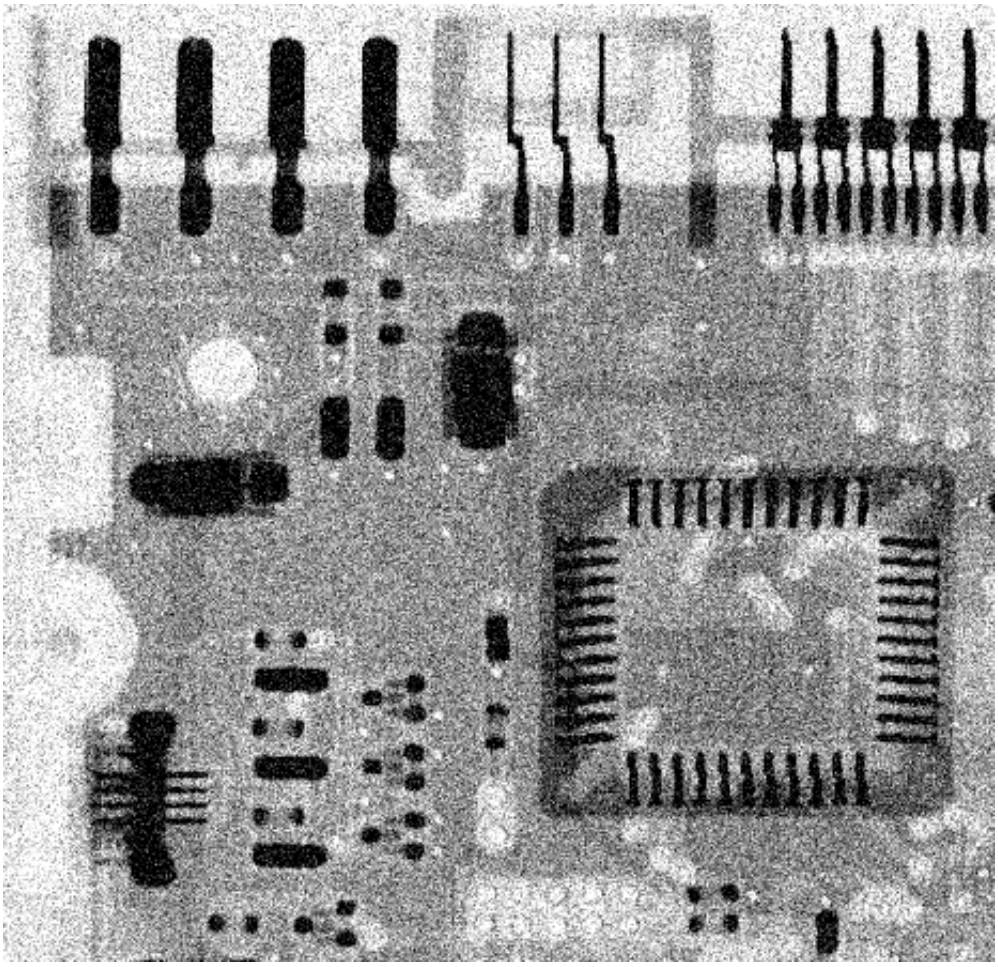
2.3 Image Denoising (40 Points)

1. Write a noise generator to add Gaussian noise or salt-and-pepper (impulse) noise to an image. Your generator should be able to specify the noise mean and standard variance forGaussian noise, and the probabilities of each of the two noise components for salt-and-pepper noise. (5 Points)

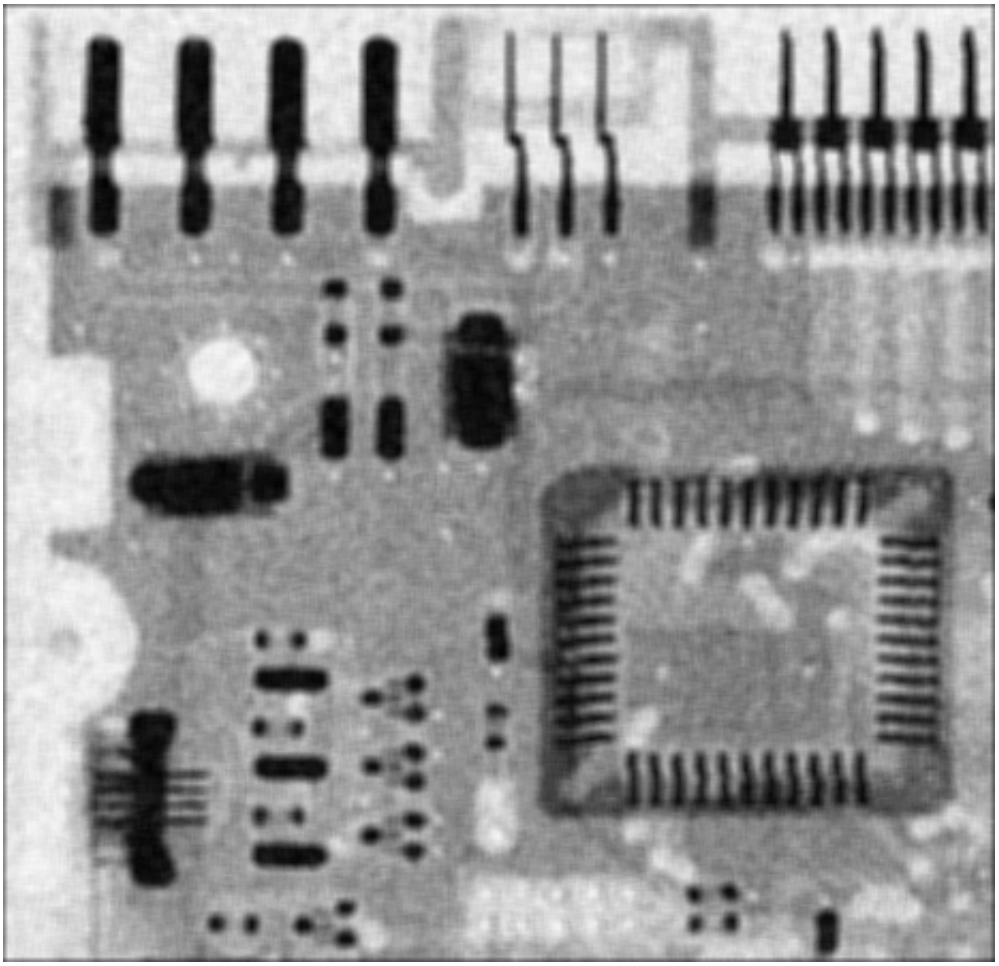
See the code file for detail ($Project$/task2/Noise.py)

2. Add Gaussian noise to your input image with mean 0 and standard variance 40, and paste the noisy image in your report. Then try to denoise it via arithmetic mean filtering, geometric mean filtering, and median filtering. Paste your filtering results in the report. Compare these results, and discuss which one looks better / worse, and why, within 1 page. (10 Points)

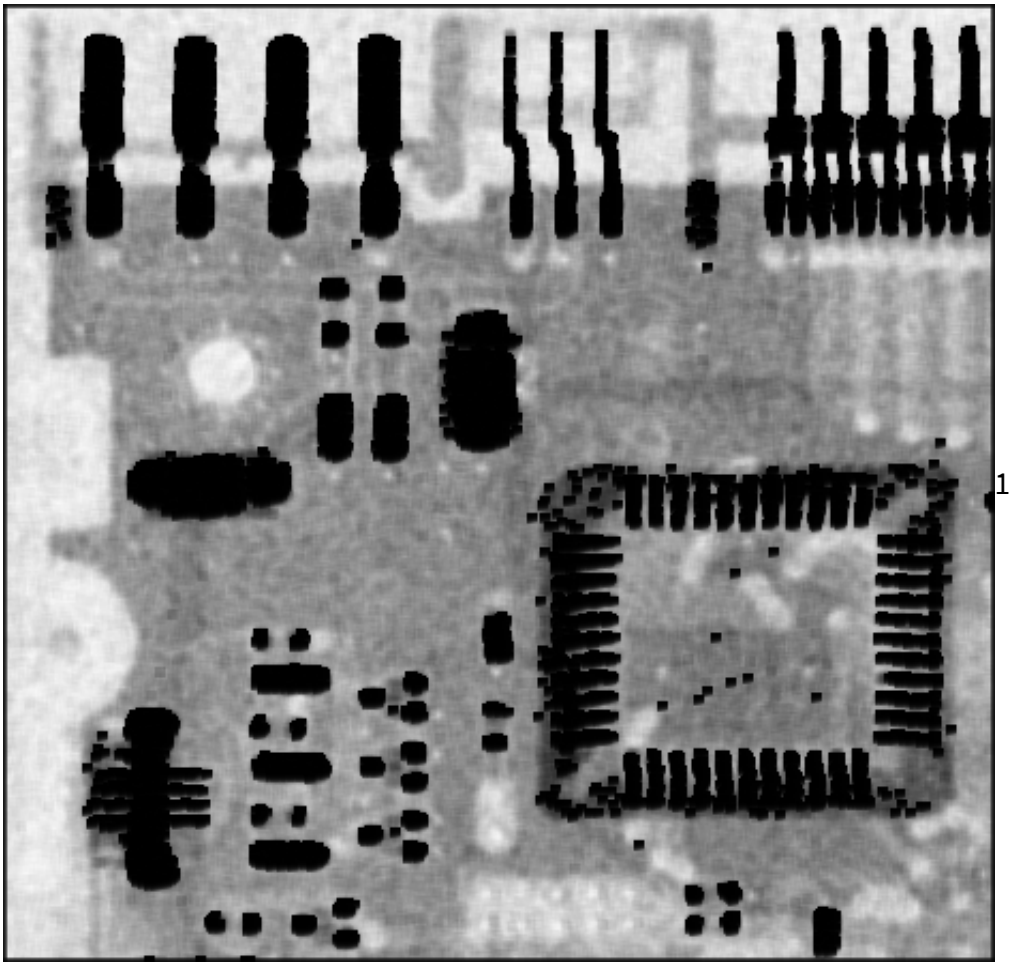
Gaussian Picture:



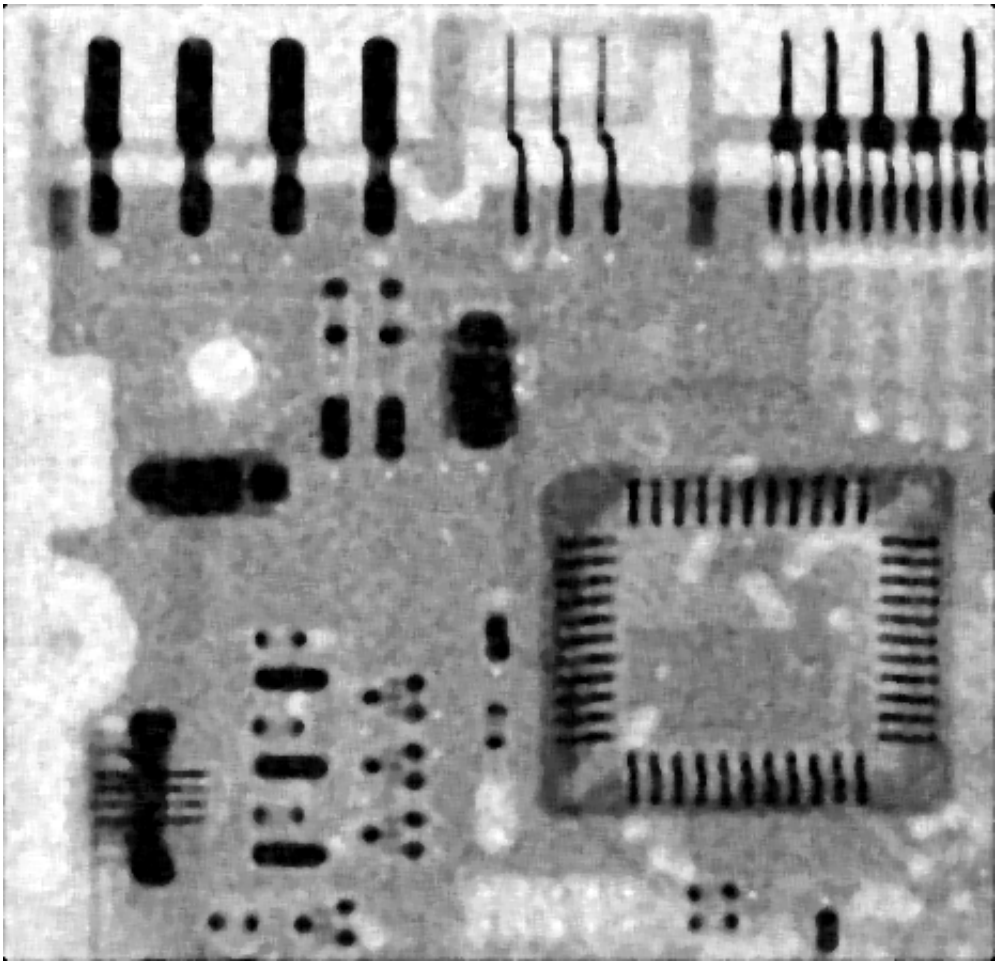
Arthmetic Mean Filter:



Geometric Mean Filter:



Median Filter:



Arthmetic Mean Filter can really reduce some noise because it's a low pass filter. But it also blurred the image. It looks OK by using 5x5 filter. 9x9 filter is bad in practical.

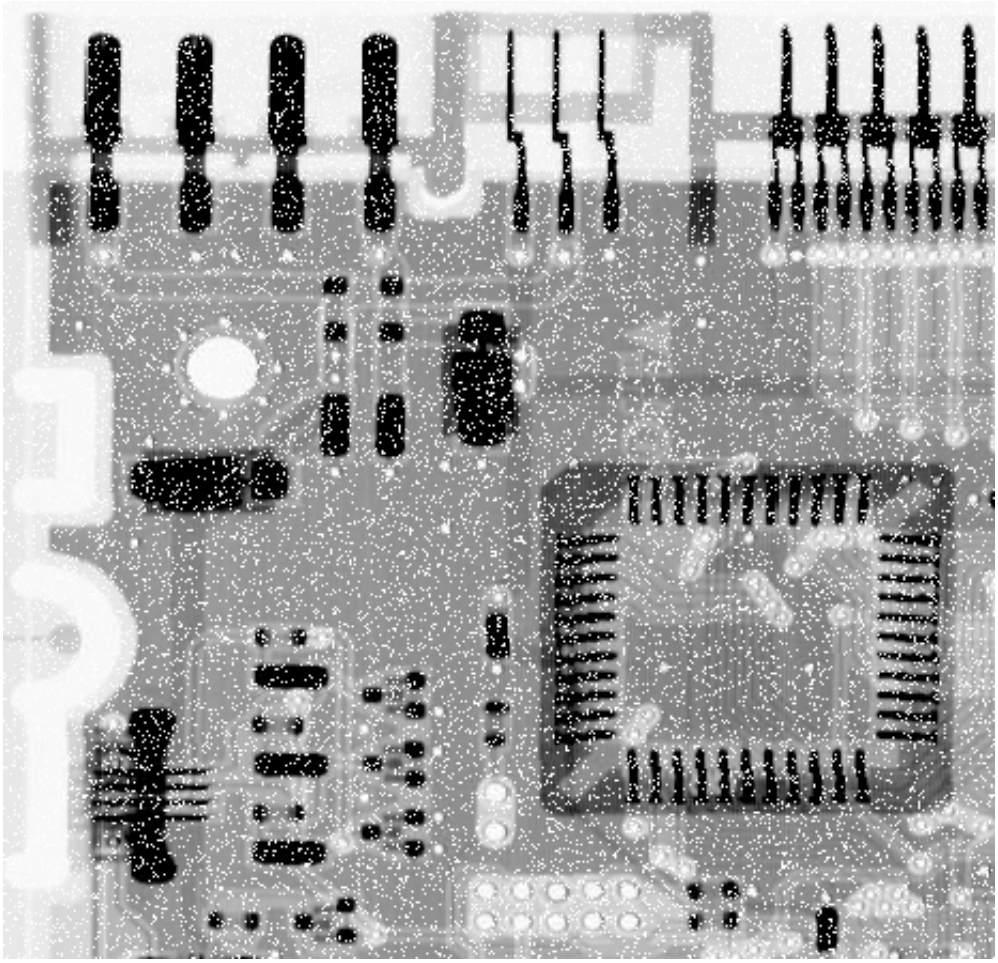
Geometric Mean Filter seems like that it can reduce more noise. It looks OK overall. But it also bolds the edge of the black elements in the image, because one zero will lead to result 0 when filtering.

Median Filter can have the effect that Arthmetic Mean Filter has. It can denois the picture and also make the picture look better.

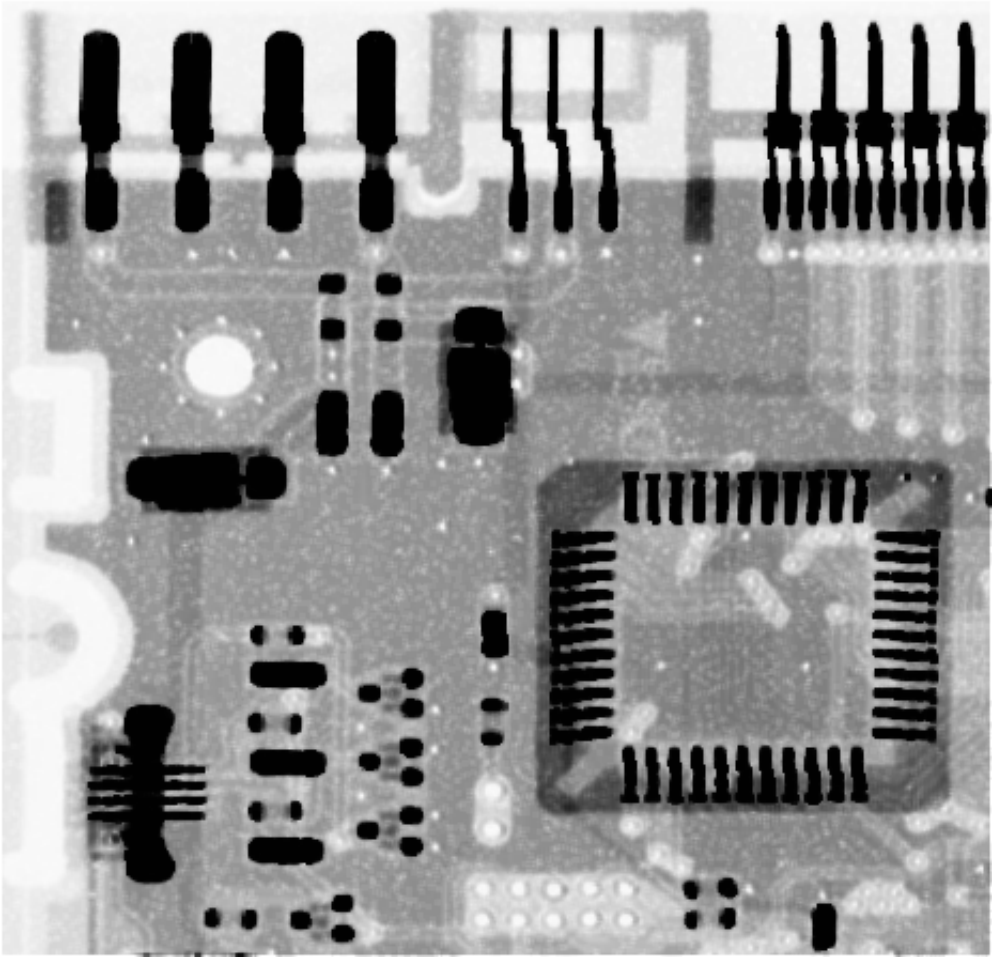
In a word, I think that median filter is better when filtering Guassian Noise.

3. Add salt noise to your input image by setting its probabilities to 0.2, and paste the noisy image in your report. Then try to denoise it via harmonic mean filtering and contraharmonic mean filtering. Paster your filtering results in the report. In addition, for contraharmonic mean filtering, you are required to paste two results: one for Q > 0 and the other for Q < 0. Discuss why setting a wrong value for Q would lead to terrible results within 1 page. (10 Points)

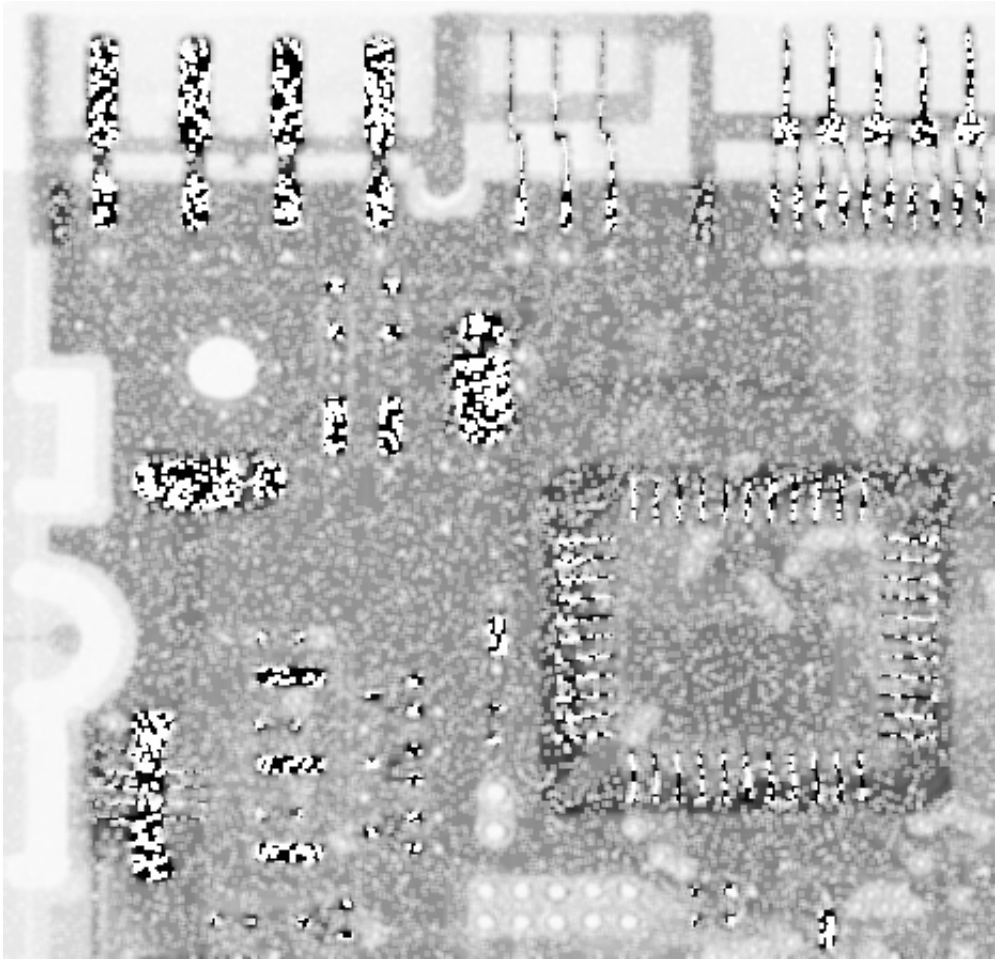
Salt noise:



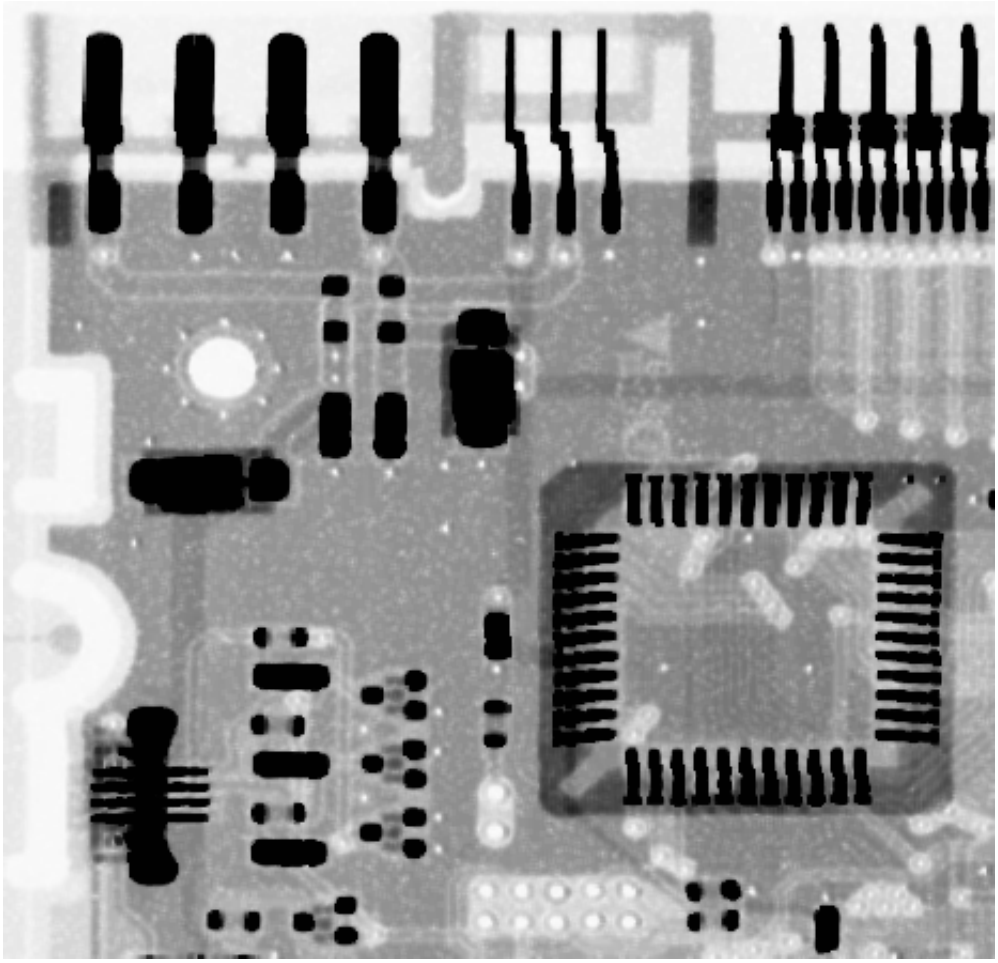
harmonic mean filtering:



contraharmonic mean filtering Q = 1.5



contraharmonic mean filtering Q = -1.5



Setting a wrong Q value can lead to terrible results because the contraharmonic mean filter can not filter both pepper noise and salt noise.

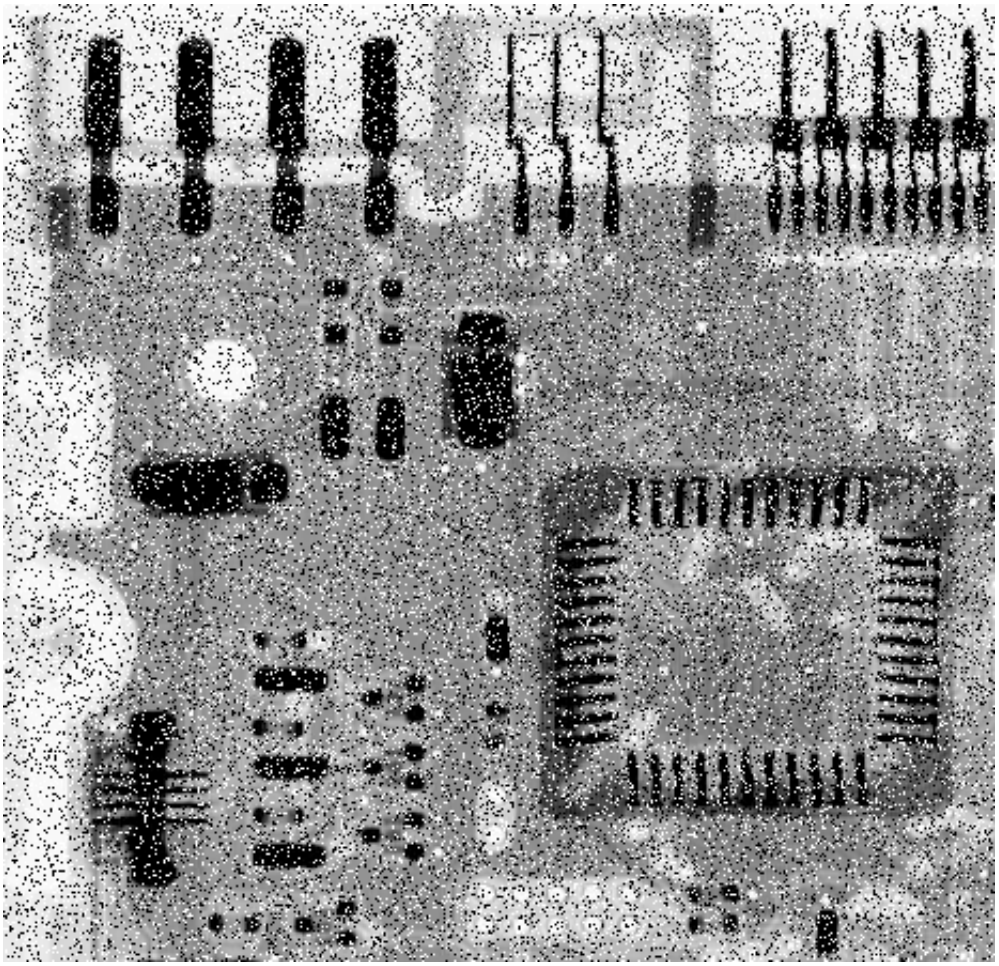
When Q is positive, the filter can reduce pepper noise. Because the most significant value is big value and the pepper value is small numbers, the black points can be removed by using this way.

When Q is negative, the filter can reduce salt noise. Because the most significant value is small value(Reciprocal Sum). The Salt value is small numbers, the white points can be remove by using this way.

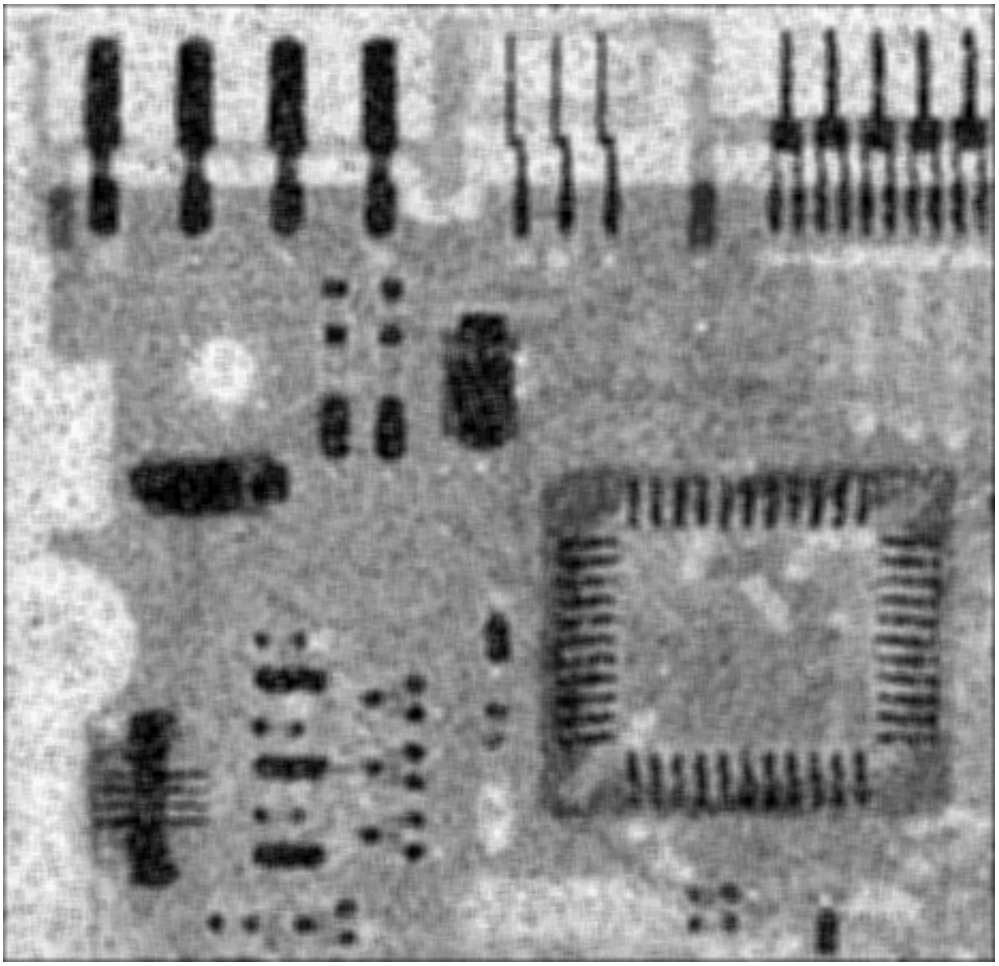
However, if you use the wrong method, terrible things will happen like using Q=1.5 to reduce salt noise, because the most significant value is big value, the white noise will become louder in this way which will lead to terrible result.

4. Add salt-and-pepper noise to your input image by setting both of the probabilities to 0.2, and paste the noisy image in your report. Then try to denoise it via arithmetic mean filtering, geometric mean filtering, max filtering, min filtering and median filtering. Paste your filtering results in the report. Compare these results, and discuss which one looks better / worse, and why, within 1 page. (10 Points)

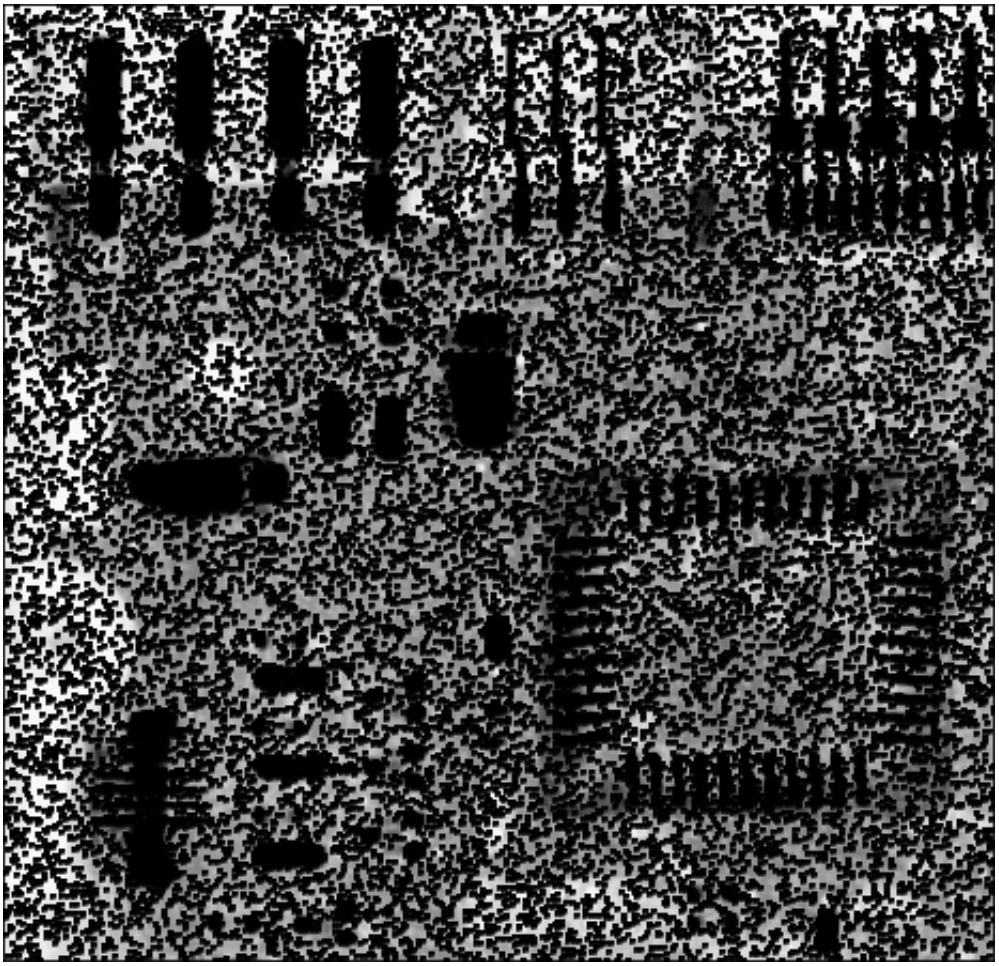
Salt and pepper noise:



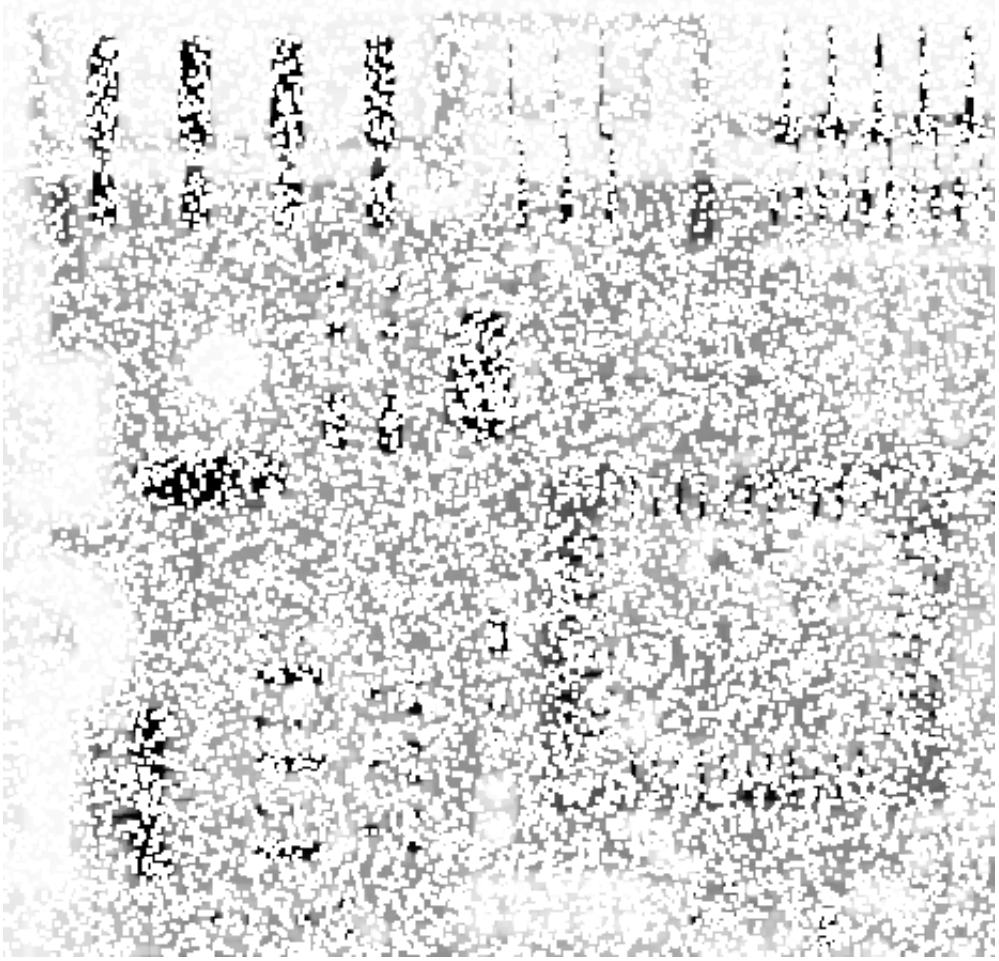
arithmetic mean filtering:



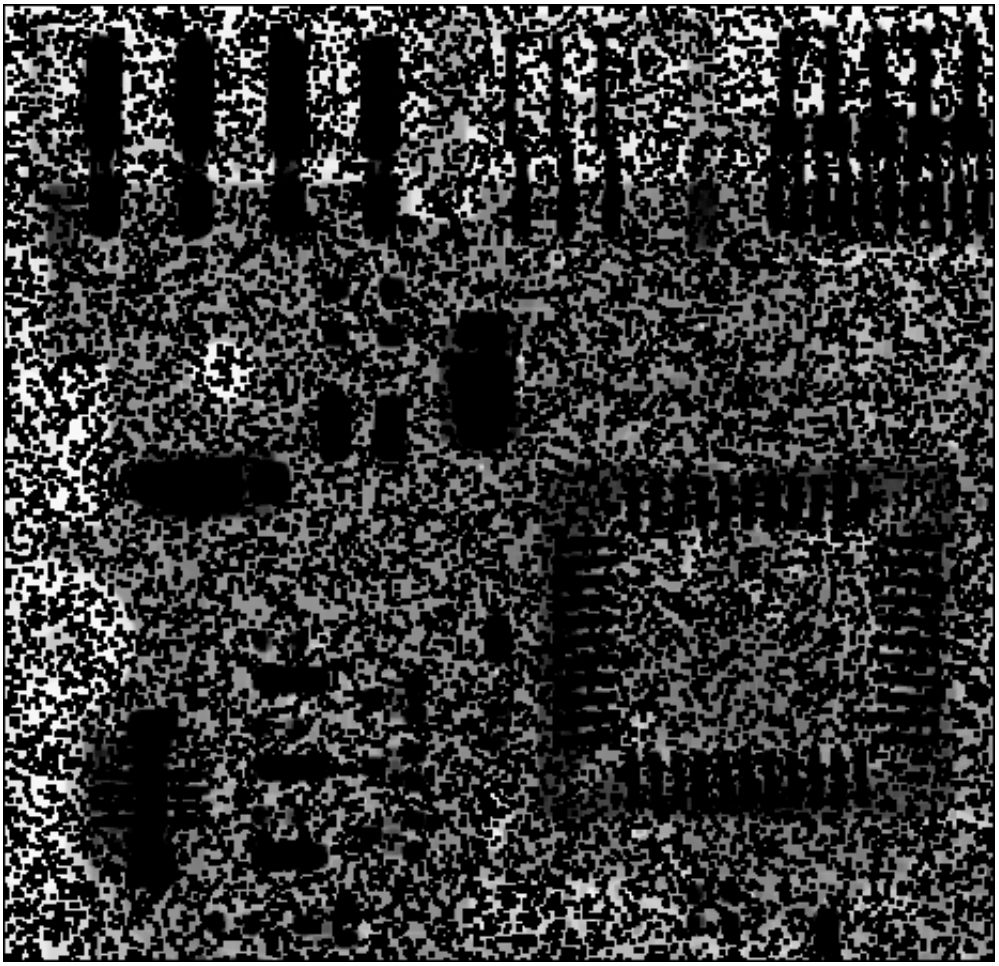
geometric mean filtering:



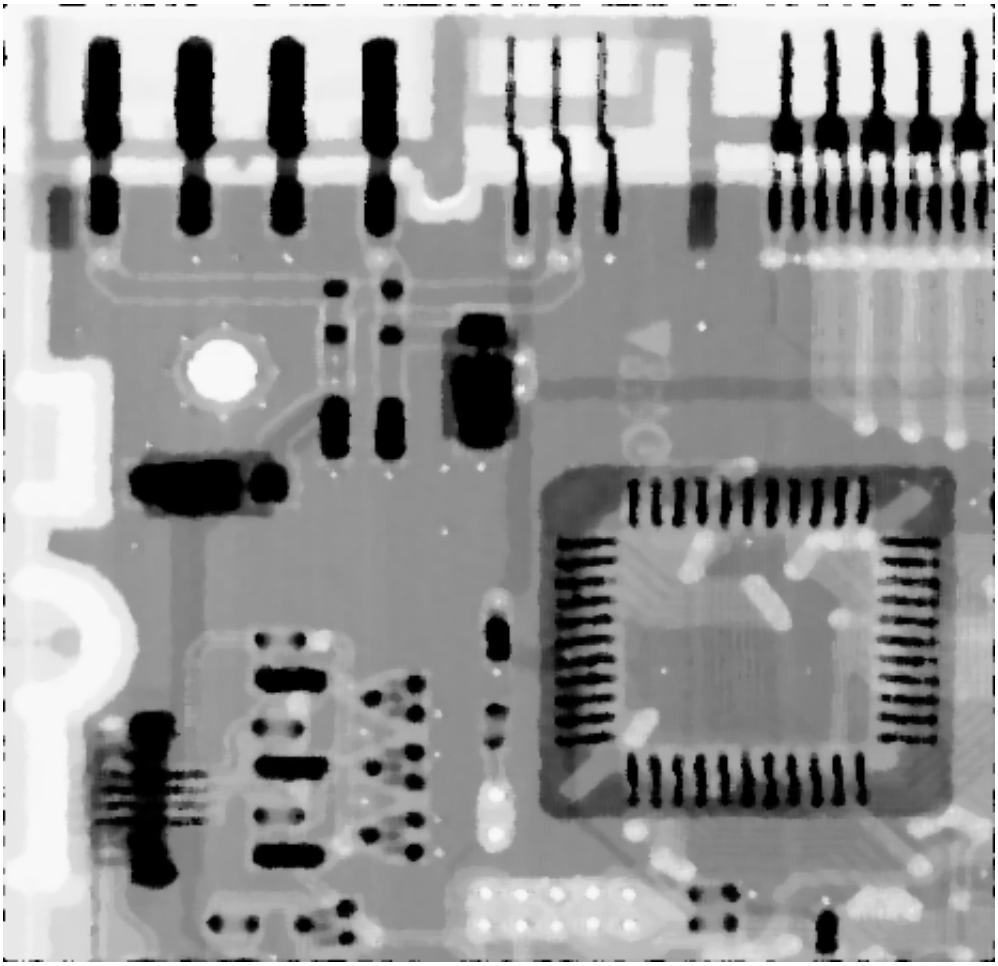
max filtering:



min filtering:



median filtering:



Median Filter looks best, and then arithmetic mean. Other filter looks very ugly.

Median Filter can do the best. In image processing, such as edge detection in the further processing of this before, usually need a certain degree of reduction Median filtering is a common step in image processing. It is especially useful for speckle noise and impulse noise. Save the edge of the feature so that it does not want to appear on the edge of the occasion is also very useful.

arithmetic mean filtering looks ok here. It's the same in Guasian Noise, Arthmetic Mean Filter can really reduce some noise because it's a low pass filter. But it also blurred the image. It looks OK by using 5x5 filter. 9x9 filter is bad in practical.

geometric mean filtering is bad, because one zero will lead to result 0 when filtering. The points near one pepper noise will lead to a big area of black, it's very terrible.

max and min is bad, because they can only reduce one kind of noise not both pepper and salt. Max can reduce pepper noise very well and Min can reduce salt noise very well.

5. Discuss how you implement all of the above filtering operations, i.e., arithmetic, geometric, harmonic, contraharmonic, max, min and median filtering, in less than 1 page. (5 Points)

arithmetic: $\frac{\sum_{S(x,y)} f(x,y)}{m \times n}$
geometric: Also using the defined formula.

harmonic: Notice that we defined

$$\frac{1}{0} = max(\frac{1}{f(x,y)}) = 1$$

```
    if img[img_x, img_y] == 0:
       centre_gray_value += 1
    else:
       centre_gray_value += 1.0 / (img[img_x, img_y] * img_filter[i][j])
```

contraharmonic: Calculate two converlosion

```
            centre_gray_value1 = Convolution(img, img_filter, x, y, method,
    q+1)
            centre_gray_value2 = Convolution(img, img_filter, x, y, method, q)
            if centre_gray_value2 != 0:
                return int(float(centre_gray_value1) / centre_gray_value2)
            else:
                return 0
```

max, median and min: Write another function to process the points, store them to new array.

```
        if method == "median":
            array.sort()
            if filter_size ** 2 % 2 == 0:
                return array[filter_size*filter_size / 2]
            else:
                return int(float(array[filter_size*filter_size / 2] \
                + array[filter_size * filter_size / 2 -1])/2)
        elif method == "max":
            return np.amax(array)
        elif method == "min":
            return np.amin(array)
```

2.4 Histogram Equalization on Color Images (30 Points)

origin image:

1. Use the function "equalize hist" that you have written in HW2 to process the R, G, B channels separately. Rebuild an RGB image from these three processed channels and paste it in the report. (10 Points)

2. Calculate the histogram on each channel separately, and then compute an average histogram from these three histograms. Use the average histogram as the basis to obtain a single histogram equalization intensity transformation function. Apply this function to the R, G and B channels individually, and again rebuild an RGB image from the three processed channels. Paste the RGB image in the report. (10 Points)

3. Compare and explain the differences in the results produced by the above two applications within 1 page. (10 Points)

Color histogram equalization is actually the image of one or more color channels for gray histogram equalization operations, common in the following ways:

Statistics of all RGB color channels of the histogram of the data and do a balanced operation, and then according to the balance of the mapping table to replace the G, B, R channel color value.

R, G, B color channels of the data and do a balanced operation, and then according to the R, B, G mapping table to replace the R, G, B channel color value.

Calculate the luminance channel by using the average value of the or the RGB, then calculate the histogram of the luminance channel and do a balanced operation, then replace the G, B, R channel color values by the mapping table.

Histogram equalization is a non-linear process. Channel splitting and equalizing each channel separately is not the proper way for equalization of contrast. Equalization involves Intensity values of the image not the color components. So for a simple RGB color image, HE should not be applied individually on each channel. Rather, it should be applied such that intensity values are equalized without disturbing the color balance of the image. So, the first step is to convert the color space of the image from RGB into one of the color space which separates intensity values from color components.

We can learn from it from another point.

If process the R, G, B channels separately, it will only have the information from R,G,B separately. If we use the whole average, it use all the information from R,G,B.

Reference:

From Wikipedia

> The above describes histogram equalization on a grayscale image. However it can also be used on color images by applying the same method separately to the Red, Green and Blue components of the RGB color values of the image. However, applying the same method on the Red, Green, and Blue components of an RGB image may yield dramatic changes in the image's color balance since the

relative distributions of the color channels change as a result of applying the algorithm. However, if the image is first converted to another color space, Lab color space, or HSL/HSV color space in particular, then the algorithm can be applied to the luminance or value channel without resulting in changes to the hue and saturation of the image.[4] There are several histogram equalization methods in 3D space. Trahanias and Venetsanopoulos applied histogram equalization in 3D color space[5] However, it results in “whitening” where the probability of bright pixels are higher than that of dark ones.[6] Han et al. proposed to use a new cdf defined by the iso-luminance plane, which results in uniform gray distribution.[7]