Assignment 4
CSC 8260 – Advanced Image Processing
Due Date: March 9th Thursday, 2023, 11:59 pm

Q1. (2 points) The two subimages shown were extracted from the top right corners of Figs. 5.7(c) and (d), respectively. Thus, the subimage on the left is the result of using an arithmetic mean filter of size 3x3, and the other subimage is the result of using a geometric mean filter of the same size.

a. Explain why the subimage obtained with a geometric mean filter is less blurred. (Hint: Start your analysis by examining a 1-D step transition in intensity.)
b. Explain why the black components in the right image are thicker.

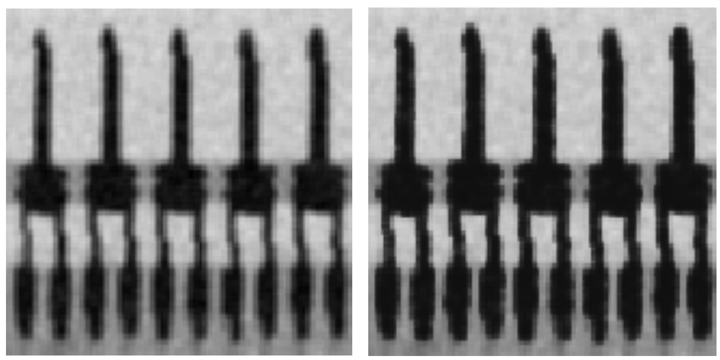
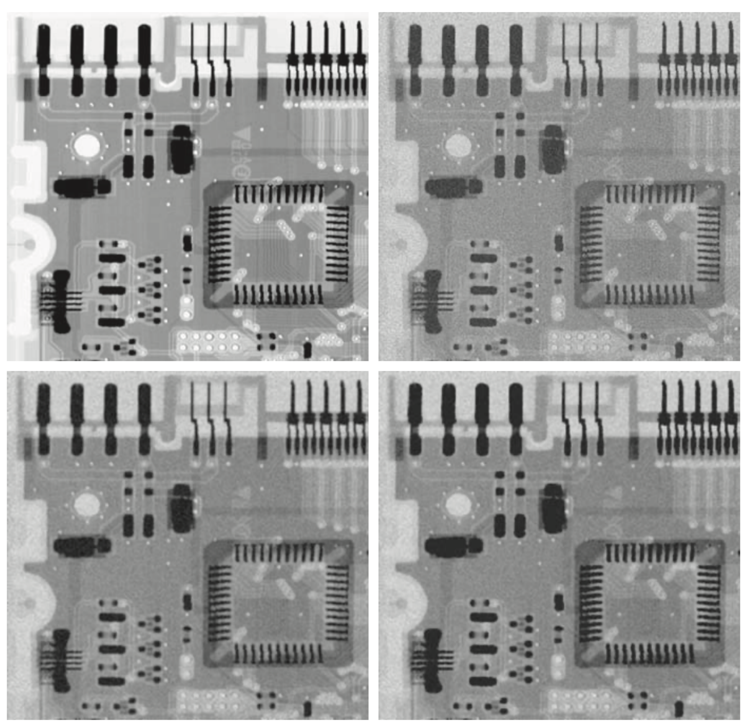


Figure 5.7 below for reference to relate it with figures in your textbook editions

330 **Chapter 5** Image Restoration and Reconstruction

a	b
c	d

FIGURE 5.7
(a) X-ray image of circuit board. (b) Image corrupted by additive Gaussian noise. (c) Result of filtering with an arithmetic mean filter of size 3×3. (d) Result of filtering with a geometric mean filter of the same size. (Original image courtesy of Mr. Joseph E. Pascente, Lixi, Inc.)



Q2. (2 points) With reference to the alpha trimmed filter defined in Eq (5.31)

a. Explain why setting d = 0 in the filter reduces it to an arithmetic mean filter
b. Explain why setting d = mn − 1 turns the filter into a median filter

Equation 5.3 below for reference to relate it with equations in your textbook editions

5.3 Restoration in the Presence of Noise Only—Spatial Filtering 333

$$\hat{f}(x,y) = \frac{1}{mn-d} \sum_{(r,c)\in S_{xy}} g_R(r,c) \qquad (5\text{-}31)$$

Q3. (2 points) Consider a linear, position invariant image degradation system with impulse response

$$h(x,y) = e^{-[(x-\alpha)^2 + (y-\beta)^2]}$$

where x and y are continuous variables. Suppose that the input to the system is a binary image consisting of a white vertical line of infinitesimal width located at $x = a$, on a black background. Such an image can be modeled as $f(x,y) = \delta(x - a)$. Assume negligible noise and use Eq. (5-61) to find the output image, $g(x, y)$.

Equation 5.61 below for reference

In this case, Eq. (5-59) reduces to

$$g(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\alpha,\beta)h(x-\alpha, y-\beta)\,d\alpha\,d\beta \qquad (5\text{-}61)$$

Q4. (2 points) Consider the problem of image blurring caused by uniform acceleration in the x-direction. If the image is at rest at time $t = 0$ and accelerates with a uniform acceleration $x_0(t) = at^2/2$ for a time T, find the blurring function H(u,v). You may assume that shutter opening and closing times are negligible.

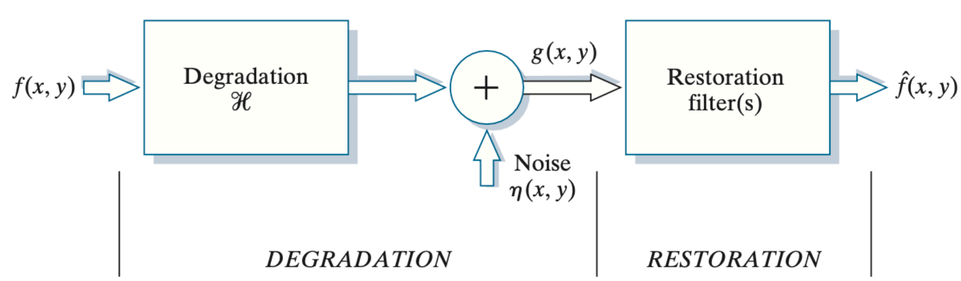
Q5. (2 points) Assume that the model in Fig. 5.1 is linear and position invariant, and that the noise and image are uncorrelated. Show that the power spectrum of the output is

$$|G(u,v)|^2 = |H(u,v)|^2\,|F(u,v)|^2 + |N(u,v)|^2$$

[*Hint:* Refer to Eqs. (5-65) and (4-89).]

Figure 5.1 below for reference

FIGURE 5.1 A model of the image degradation/restoration process.



Equation 5.65 below for reference

$$G(u,v) = H(u,v)F(u,v) + N(u,v) \tag{5-65}$$

Equation 4.89 below for reference

$$\begin{aligned} P(u,v) &= |F(u,v)|^2 \\ &= R^2(u,v) + I^2(u,v) \end{aligned} \tag{4-89}$$

Q6. (7 points) This project deals with extending the linear spatial filtering concepts introduced in Chapter 3. The objective is to use project function twodConv4e (NOTE: this function is referenced to a problem in chapter 3. You can choose to implement it or avoid by doing a simple convolution) to implement the mean filters discussed in Section 5.3. Although some of those filters (e.g. the geometric mean filter) perform nonlinear operations on the pixels of a neighborhood, it is possible to convert the nonlinear operations to a form that allows linear filtering (i.e., sum-of-products operations) using spatial convolution. The solution to (b) shows how do to this. In the following, g is a noisy input image, f_hat is the filtered (estimate) image, and m x n is the size of the neighborhood that defines the filter size.

a. Write a function f_hat = aMean4e(g,m,n) that implements the arithmetic mean filter defined in Eq. (5-23).
b. Write a function f_hat = geoMean4e(g,m,n) that implements the geometric mean filter defined in Eq. (5-24).
c. Write a function f_hat = harMean4e(g,m,n) that implements the harmonic mean filter defined in Eq. (5-25).
d. Write a function f_hat = ctharMean4e(g,m,n,q) that implements the contra harmonic mean filter in Eq. (5-26). (Hint: Because you will be performing a ratio computation of two filtered images, you should disable the auto-scaling in project function twodConv4e, and do the scaling at the end using project function intScaling4e.)
e. Read the image circuitboard-gaussian.tif and use function aMean4e to duplicate the result in Fig. 5.7(c).
f. Read the image circuitboard-gaussian.tif and use function geoMean4e to duplicate the result in Fig. 5.7(d).
g. Read the image circuitboard-pepper.tif and use function ctharMean4e to duplicate the result in Fig. 5.8(c).
h. Read the image circuitboard-salt.tif and use function ctharMean4e to duplicate the result in Fig. 5.8(d).

Section 5.3 below for reference

5.3 RESTORATION IN THE PRESENCE OF NOISE ONLY—SPATIAL FILTERING

When an image is degraded only by additive noise, Eqs. (5-1) and (5-2) become

$$g(x,y) = f(x,y) + \eta(x,y) \tag{5-21}$$

and

$$G(u,v) = F(u,v) + N(u,v) \tag{5-22}$$

Equation 5.23 below for reference

$$\hat{f}(x,y) = \frac{1}{mn} \sum_{(r,c) \in S_{xy}} g(r,c) \tag{5-23}$$

Equation 5.24 below for reference

$$\hat{f}(x,y) = \left[\prod_{(r,c) \in S_{xy}} g(r,c) \right]^{\frac{1}{mn}} \tag{5-24}$$

Equation 5.25 below for reference

$$\hat{f}(x,y) = \frac{mn}{\displaystyle\sum_{(r,c) \in S_{xy}} \frac{1}{g(r,c)}} \tag{5-25}$$

Equation 5.26 below for reference

$$\hat{f}(x,y) = \frac{\displaystyle\sum_{(r,c) \in S_{xy}} g(r,c)^{Q+1}}{\displaystyle\sum_{(r,c) \in S_{xy}} g(r,c)^{Q}} \tag{5-26}$$

Figure 5.7 below for reference

a b
c d

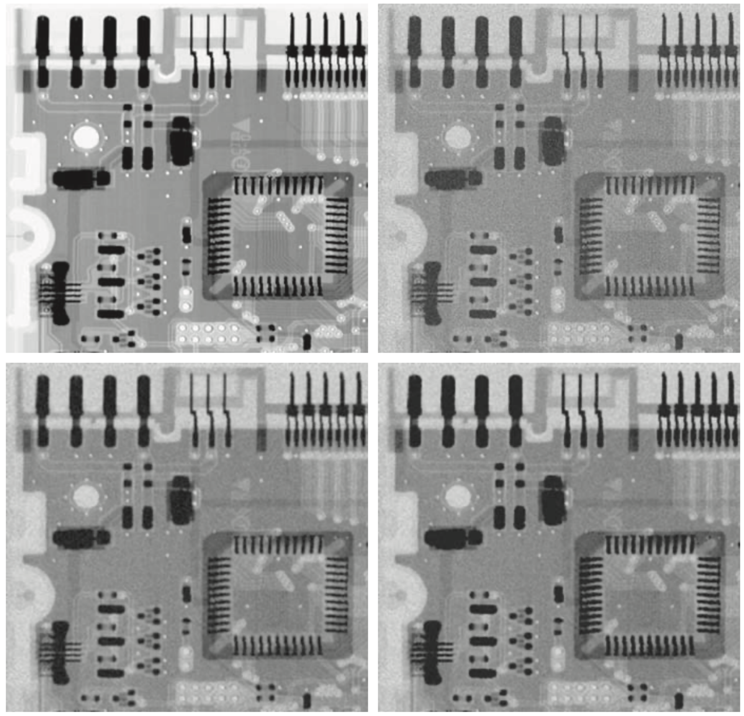


FIGURE 5.7 (a) X-ray image of circuit board. (b) Image corrupted by additive Gaussian noise. (c) Result of filtering with an arithmetic mean filter of size 3×3. (d) Result of filtering with a geometric mean filter of the same size. (Original image courtesy of Mr. Joseph E. Pascente, Lixi, Inc.)

Figure 5.8 below for reference

a b
c d

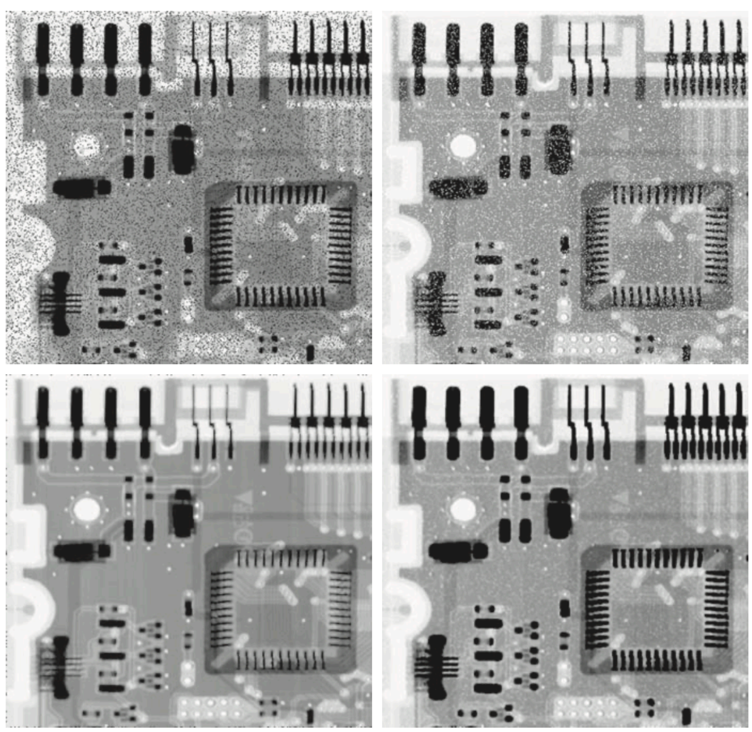


FIGURE 5.8 (a) Image corrupted by pepper noise with a probability of 0.1. (b) Image corrupted by salt noise with the same probability. (c) Result of filtering (a) with a 3×3 contra-harmonic filter $Q = 1.5$. (d) Result of filtering (b) with $Q = -1.5$.

Q7. (3 points) Blurring transfer function.

a. Write a function H = motionBlurTF4e(P,Q,a,b,T) that implements a P x Q blurring transfer function, as given in Eq. (5-77). The parameters a, b, and T are as defined for that equation. Transfer function H must be centered on the frequency rectangle.

b. Display the spectrum of H.

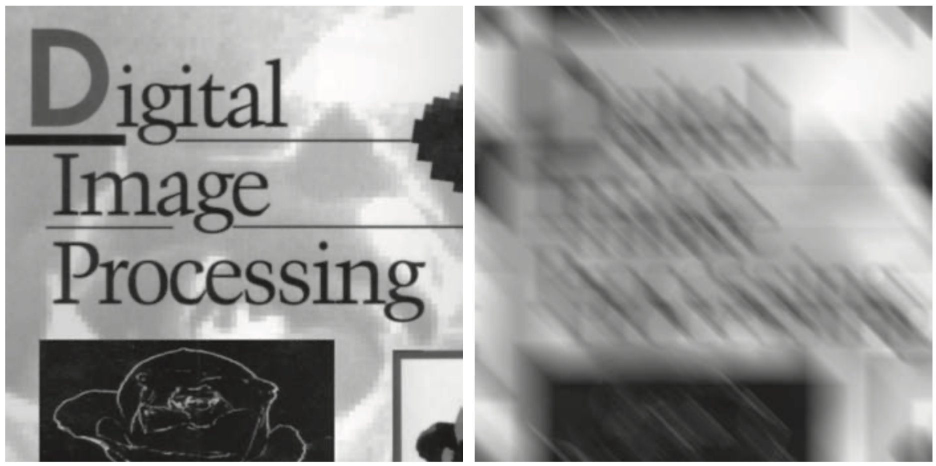
c. Read the image bookcover.tif and blur it with the same parameters used to generate the image in Fig. 5.26(b). Display your results.

Figure 5.26 and Equation 5.77 below for reference

356 **Chapter 5** Image Restoration and Reconstruction

a b

FIGURE 5.26
(a) Original image. (b) Result of blurring using the function in Eq. (5-77) with $a = b = 0.1$ and $T = 1$.



If we allow the y-component to vary as well, with the motion given by $y_0(t) = bt/T$, then the degradation function becomes

$$H(u,v) = \frac{T}{\pi(ua + vb)} \sin\left[\pi(ua + vb)\right] e^{-j\pi(ua + vb)} \qquad (5\text{-}77)$$

Chapter 3 – Reference for Assignment Question 6

Q. Two-dimensional convolution

a. Write a function g = twodConv4e(f, W) for performing 2-D convolution of image f and kernel w in the language you are using for your projects. This function should use replicate padding by default. If you are using MATLAB, use function conv2 as a starting point. This function uses zero-padding by default but you can get around this by pre-padding 1 with replicate padding and then stripping out the excess rows and columns due to the extra replicate padding before outputting 0. Function conv2 requires floating point inputs. Your function should by default scale the input to the range [0, 1] using the default mode in project function intScaling4e. However, the function also has to have the capability of disabling this automatic scaling.

b. Create an image of size 512 × 512 pixels that consists of a unit impulse at location (256,256) and zeros elsewhere. Use this image and a kernel of your choice to confirm that your function is indeed performing convolution. Display your results and explain what you did and why.

Bonus Section

Q2. (5 points) Parametric Wiener filter.

a. Write a function W = pWienerTF4e(H, K) that implements the parametric Wiener filter transfer function given in Eq. (5-85), where H is a degradation transfer function, and K is a scalar parameter.

b. Read the image bookcover-blurred.tif and restore it using the degradation function from Eq. (5-77), without padding. (Hint: the image bookcover-blurred.tif was created using the degradation function in Eg. (5-77) with a=b=0.1 and T= 1. Also, the value of K needed for proper restoration is in the range 10-3 to 10-4.)

c. Read the image bookcover.tif (this is the unblurred image) and explain why your restored image is not quite as good as the original.

d. Repeat (b), but this time first pad image bookcover-blurred.fit with zero padding of size P=2M and Q=2N. (Note that the estimate of the blurring function will now be of size 2M × 2N.) You will get an unexpected result.

e. Read the image bookcover.tif, pad it with zeros to size 2M × 2N first, and then blur it using project function motionBlurtf4e with parameters a = 0.1, b = -0.1, and T = 1.0. Repeat (b) and display your result after cropping the image to its original M × N size. You will find that this approach works as expected.

f. Explain why (d) and (e) gave such different results. Your explanation will reveal why we did not use image padding in the image restoration techniques discussed in this chapter.

g. Show how you would go about blurring the original bookcover.tif image, and then restoring it so that the result is identical to the original.

h. In order to focus on the effects of noise on restoration, we worked with the original (complex) blurred image in Examples 5.11 and 5.12. That is the reason the results in the low-noise case were so close to the original. In order to see what the results would have been if we had started instead with a version of the low-noise blurred image that was converted to tif for archival, repeat (b) using image bookcover-blurred.tif. The blurring parameters in the estimate of Hare a = b = 0.1, and T = 1.0. Use the same K as in (b).

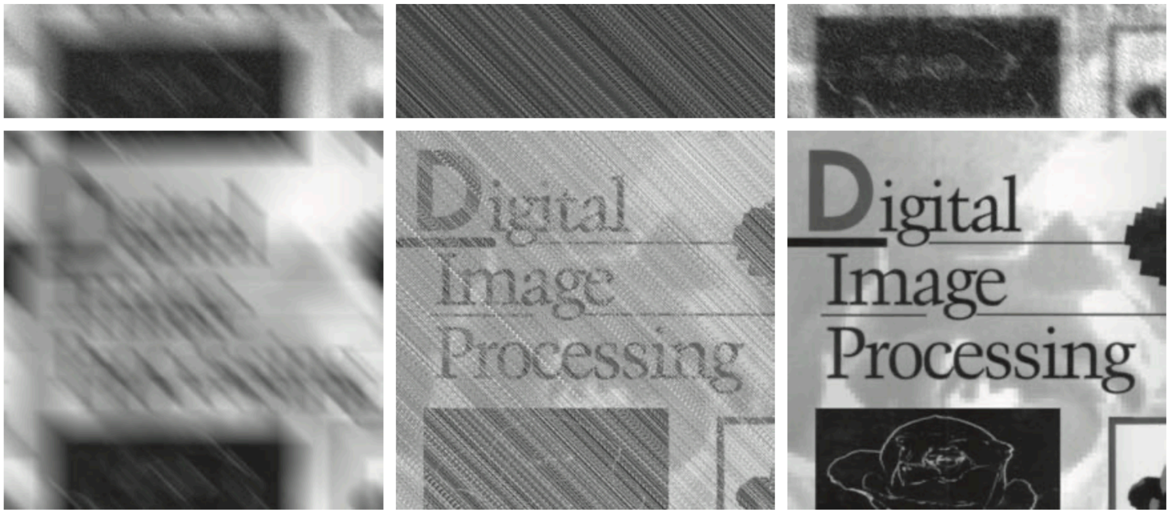
Equation 5.85 below for reference

$$\hat{F}(u,v) = \left[\frac{1}{H(u,v)} \frac{|H(u,v)|^2}{|H(u,v)|^2 + K} \right] G(u,v) \qquad (5\text{-}85)$$

Example 5.11 below for reference

EXAMPLE 5.11: More deblurring examples using Wiener filtering.

The first row of Fig. 5.29 shows, from left to right, the blurred image of Fig. 5.26(b) heavily corrupted by additive Gaussian noise of zero mean and variance of 650; the result of direct inverse filtering; and the result of Wiener filtering. The Wiener filter of Eq. (5-85) was used, with $H(u,v)$ from Example 5.8, and with K chosen interactively to give the best possible visual result. As expected, direct inverse filtering produced an unusable image. Note that the noise in the inverse filtered image is so strong that it masks completely the content of the image. The Wiener filter result is by no means perfect, but it does give us a hint as to image content. The text can be read with moderate effort.



a b c
d e f
g h i

FIGURE 5.29 (a) 8-bit image corrupted by motion blur and additive noise. (b) Result of inverse filtering. (c) Result of Wiener filtering. (d)–(f) Same sequence, but with noise variance one order of magnitude less. (g)–(i) Same sequence, but noise variance reduced by five orders of magnitude from (a). Note in (h) how the deblurred image is quite visible through a "curtain" of noise.

Example 5.12 below for reference

EXAMPLE 5.12: Comparison of deblurring by Wiener and constrained least squares filtering.

Figure 5.30 shows the result of processing Figs. 5.29(a), (d), and (g) with constrained least squares filters, in which the values of γ were selected manually to yield the best visual results. This is the same procedure we used to generate the Wiener filter results in Fig. 5.29(c), (f), and (i). By comparing the constrained least squares and Wiener results, we see that the former yielded better results (especially in terms of noise reduction) for the high- and medium-noise cases, with both filters generating essentially

a b c

FIGURE 5.30 Results of constrained least squares filtering. Compare (a), (b), and (c) with the Wiener filtering results in Figs. 5.29(c), (f), and (i), respectively.