# *Computer Vision*

Course 6

**Course 5**

## Discrete Fourier Transform (DFT)

$f(x)=\{f(0), f(1), \ldots, f(N-1)\}$

One-dimensional DFT

$$F(u)=\frac{1}{N}\sum_{x=0}^{N-1} f(x)e^{-i\frac{2\pi u x}{N}},\ u=0,1,...,N-1$$

The inverse transformation

$$f(x)=\sum_{u=0}^{N-1} F(u)e^{i\frac{2\pi u x}{N}},\ x=0,1,...,N-1$$

$f = \{f(x,y);\ x=0,1,\ldots,M-1\ ;\ y=0,1,\ldots,N-1\}$

Two-dimensional DFT

$$F(u,v) = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-i\,2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)},\ u=\overline{0,M-1}, v=\overline{0,N-1}$$

The inverse transformation

$$f(x,y) = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F(u,v)e^{i\,2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)},\ x=\overline{0,M-1}, y=\overline{0,N-1}$$

**Course 5**

## Fast Fourier Transform

Using the above definition, the computational complexity of the DFT is $O(N^2)$. Using a divide-and-conquer approach one can reduce the computational complexity to $O(N\log_2 N)$. This algorithm is known as the *Fast Fourier Transform* (FFT).

$$F(u)=\frac{1}{N}\sum_{x=0}^{N-1} f(x)\left(k_N\right)^{ux}, \quad k_N = e^{-i\frac{2\pi}{N}}.$$

We assume that $N=2^P=2M$.

$$F(u)=\frac{1}{2M}\sum_{x=0}^{2M-1} f(x)\left(k_{2M}\right)^{ux}=$$

$$=\frac{1}{2}\left\{\frac{1}{M}\sum_{x=0}^{M-1} f(2x)\left(k_{2M}\right)^{u(2x)}+\frac{1}{M}\sum_{x=0}^{M-1} f(2x+1)\left(k_{2M}\right)^{u(2x+1)}\right\}$$

$$=\frac{1}{2}\left\{\frac{1}{M}\sum_{x=0}^{M-1} f(2x)\left(k_{2M}\right)^{u(2x)}+\frac{1}{M}\sum_{x=0}^{M-1} f(2x+1)\left(k_{2M}\right)^{2ux} k_{2M}^{u}\right\}$$

$$F(u)=\frac{1}{2}\left\{\frac{1}{M}\sum_{x=0}^{M-1} f(2x)\left(k_{M}\right)^{ux}+\frac{1}{M}\sum_{x=0}^{M-1} f(2x+1)\left(k_{M}\right)^{ux} k_{2M}^{u}\right\}=$$

$$=\frac{1}{2}\left\{F_{\text{even}}(u)+k_{2M}^{u}F_{\text{odd}}(u)\right\} \quad \left(\text{ we used } k_{2M}^{2ux}=k_{M}^{ux}\right)$$

$F_{even}(u)$ is the DFT of the sequence composed of the even samples ***f(2x) (f(0), f(2), …, f(2M-2))*** of the original discrete signal and $F_{odd}(u)$ is the DFT is the DFT of the odd samples ***f(2x+1) (f(1), f(3), …, f(2M-1))***. Both even and odd sequences have ***N/2*** length, thus the ***N***-point DFT ***F(u)*** can be computed as the sum of two $\frac{N}{2}$-point DFT.

**Course 5**

## Filtering in the Frequency Domain

Let ***f(x,y)*** be a digital image and ***F(u,v)*** its (discrete) Fourier transform. Usually it is not possible to make direct associations between specific components of an image and its transform. We know that ***F(0,0)*** is proportional to the average intensity of the image. Low frequencies correspond to the slowly varying intensity components of an image, the higher frequencies correspond to faster intensity change in an image (edges, for ex.).

$$F(0,0) = MN\ \bar{f} \quad , \quad \bar{f} = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y)$$

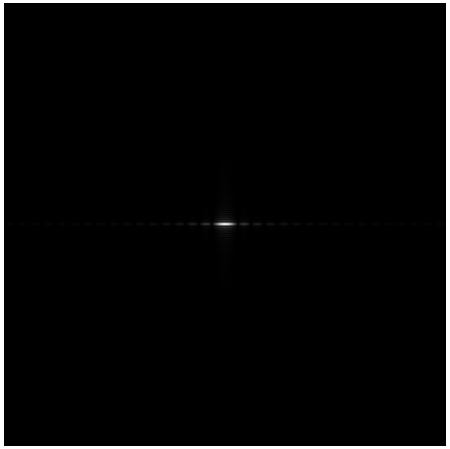
$\bar{f}$ – the average value of the image $f$

$$f(x-x_0, y-y_0) \quad \leftrightarrow \quad F(u,v)e^{-i2\pi\left(\frac{x_0 y}{M}+\frac{y_0 v}{N}\right)}$$

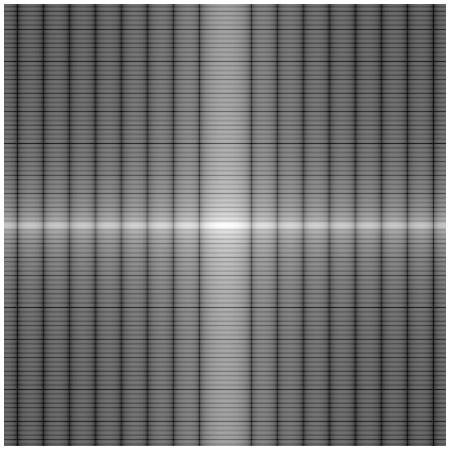
$$f(r, \theta+\theta_0) \quad \leftrightarrow \quad F(\omega, \varphi+\theta_0)$$

The spectrum is insensitive to image translation, and it rotates by the same angle as the image rotates.

image

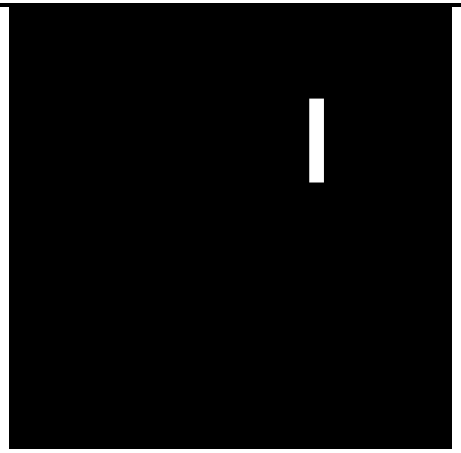


Fourier spectrum

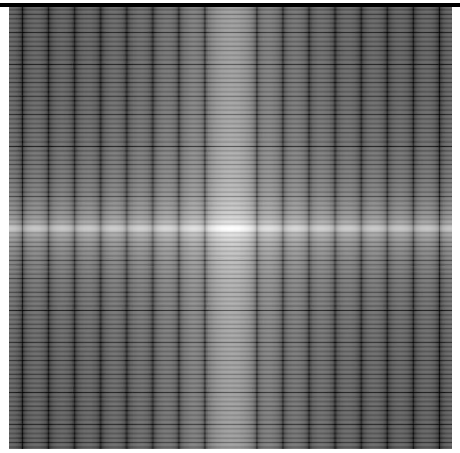


centered Fourier spectrum

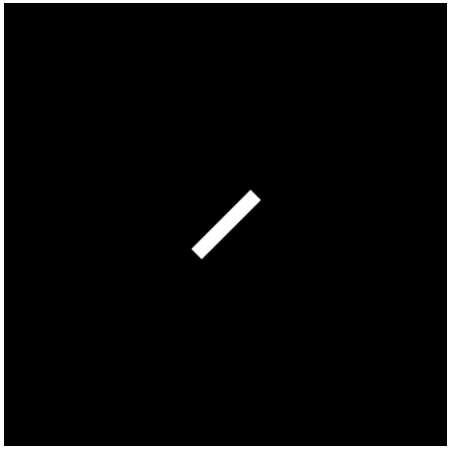
log transformed centered Fourier spectrum



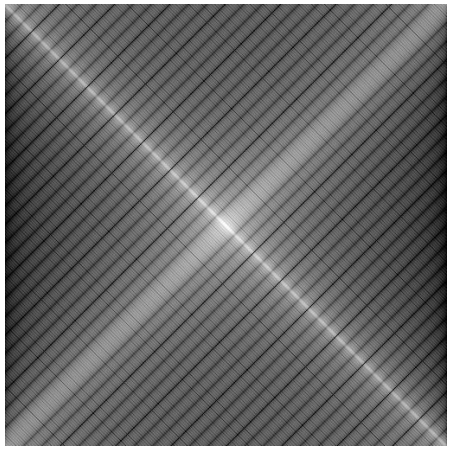
translated image



Fourier spectrum



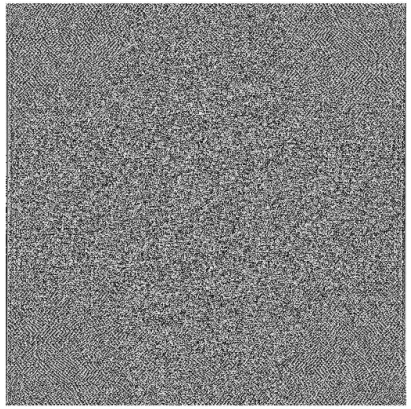
45° rotated image



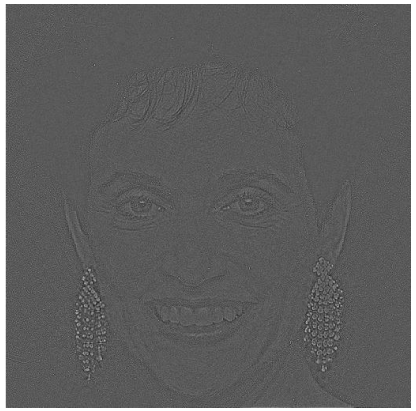
Fourier spectrum

The magnitude of the 2-D DFT is an array whose components determine the intensities in the image, the corresponding phase is an array of angles that carry the information about where discernible objects are located in the image.

Woman

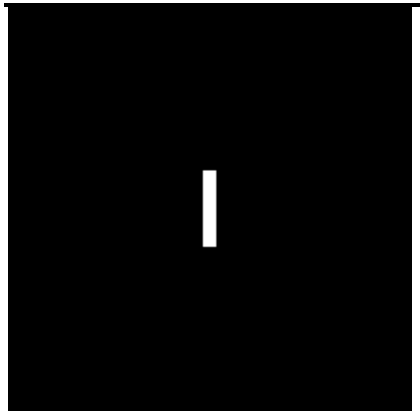


phase angle



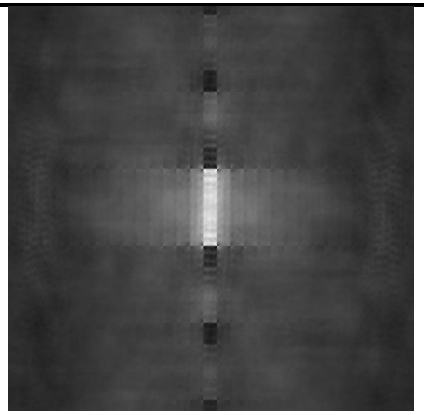
reconstruction only with phase angle

## Course 5

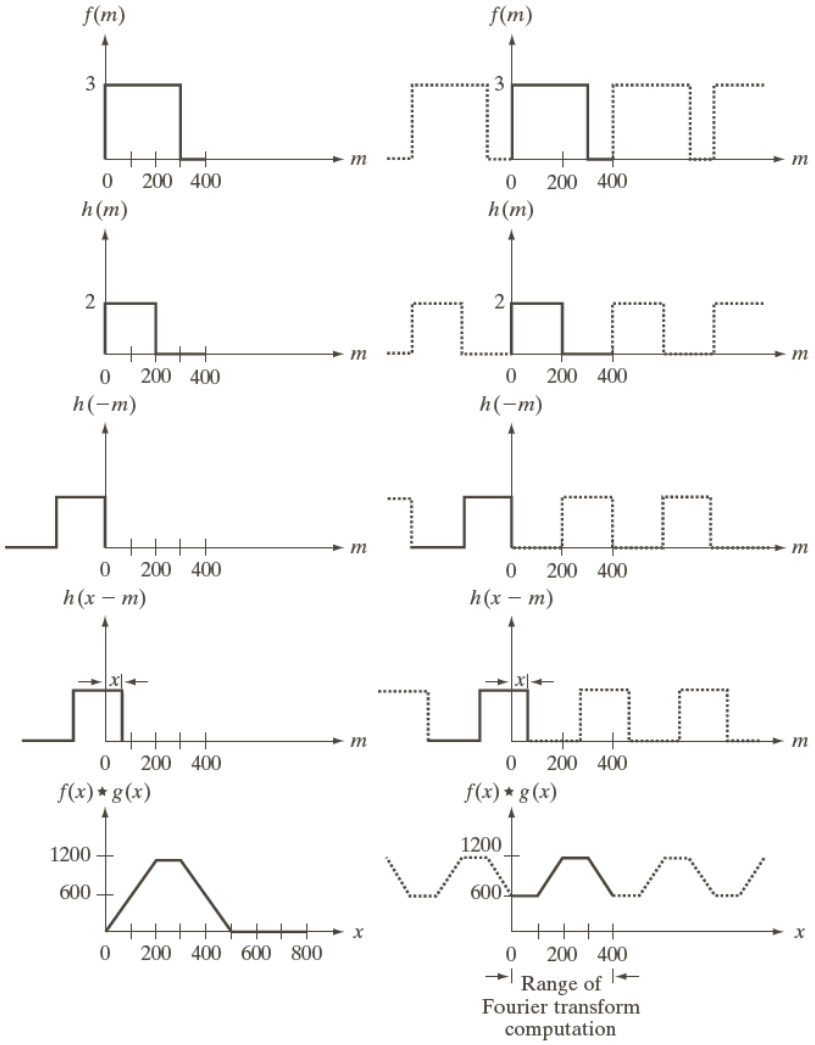


Rectangle

rectangle spectrum+phase angle woman



rectangle phase angle + spectrum woman




a f
b g
c h
d i
e j

**FIGURE 4.28** Left column: convolution of two discrete functions obtained using the approach discussed in Section 3.4.2. The result in (e) is correct. Right column: Convolution of the same functions, but taking into account the periodicity implied by the DFT. Note in (j) how data from adjacent periods produce wraparound error, yielding an incorrect convolution result. To obtain the correct result, function padding must be used.

If we use the DFT and the convolution theorem to obtain the same result in the left column of Figure 4.28, we must take into account the periodicity inherent in the expression for the DFT. The problem which appears in Figure 4.28 is commonly referred to as *wraparound error*. The solution to this problem is simple. Consider two functions ***f*** and ***h*** composed of ***A*** and ***B*** samples. It can be shown that if we append zeros to both functions so that they have the same length, denoted by ***P***, then wraparound is avoided by choosing:

$$P \geq A + B - 1$$

This process is called *zero padding*.

Let ***f(x,y)*** and ***h(x,y)*** be two image arrays of size ***A×B*** and ***C×D*** pixels, respectively. Wraparound error in their circular convolution can be avoided by padding these functions with zeros:

$$f_p(x,y)=\begin{cases} f(x,y) & 0\le x\le A-1 \text{ and } 0\le y\le B-1 \\ 0 & A\le x\le P \text{ and } B\le y\le Q \end{cases}$$

$$h_p(x,y)=\begin{cases} h(x,y) & 0\le x\le C-1 \text{ and } 0\le y\le D-1 \\ 0 & C\le x\le P \text{ and } D\le y\le Q \end{cases}$$

$$P\ge A+C-1 \quad (P\ge 2M-1) \quad , \quad Q\ge B+D-1 \quad (Q\ge 2N-1)$$

**Course 5**

## Frequency Domain Filtering Fundamentals

Given a digital image ***f(x,y)*** of size ***M×N***, the basic filtering equation has the form:

$$g(x,y) = \mathcal{F}^{-1}\left[H(u,v)F(u,v)\right] \qquad (1)$$

Where $\mathcal{F}^{-1}$ is the inverse discrete Fourier transform (IDFT), ***F(u,v)*** is the discrete Fourier transform (DFT) of the input image, ***H(u,v)*** is a *filter function* (also called *filter* or the *filter transfer function*) and ***g(x,y)*** is the filtered (output) image.

***F, H,*** and ***g*** are arrays of the same size as ***f***, ***M×N***.

***H(u,v)*** – symmetric about its center simplifies the computations and also requires that ***F(u,v)*** to be centered.

In order to obtain a centered ***F(u,v)*** the image ***f(x,y)*** is multiplied by $(-1)^{x+y}$ before computing its transform.

$$H(u,v)=\begin{cases}0 & u=M/2,\ v=N/2\ (u=v=0)\\ 1 & \text{elsewhere}\end{cases}$$

This filter rejects the *dc* term (responsible for the average intensity of an image) and passes all other terms of ***F(u,v)***. This filter will reduce the average intensity of the output image to zero.

Low frequencies in the transform are related to slowly varying intensity components in an image (such as walls in a room, or a cloudless sky) and high frequencies are caused by sharp transitions in intensity, such as edges or noise. A filter ***H(u,v)*** that attenuates high frequencies while passing low frequencies (i.e. a *lowpass filter*) would blur an image while a filter with the opposite

property (*highpass filter*) would enhance sharp detail, but cause a reduction of contrast in the image.

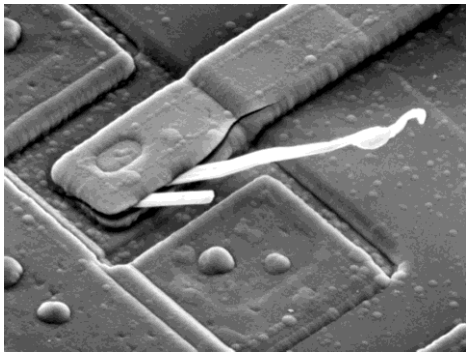
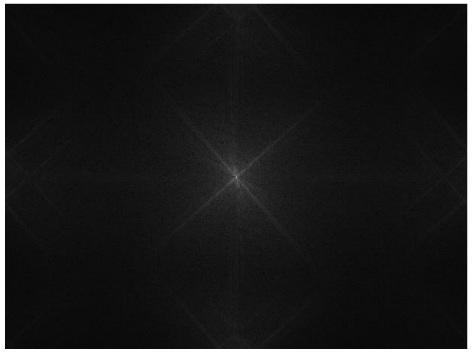
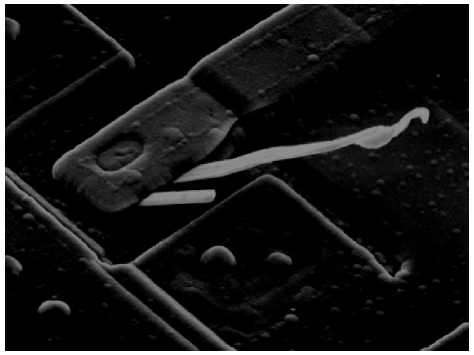


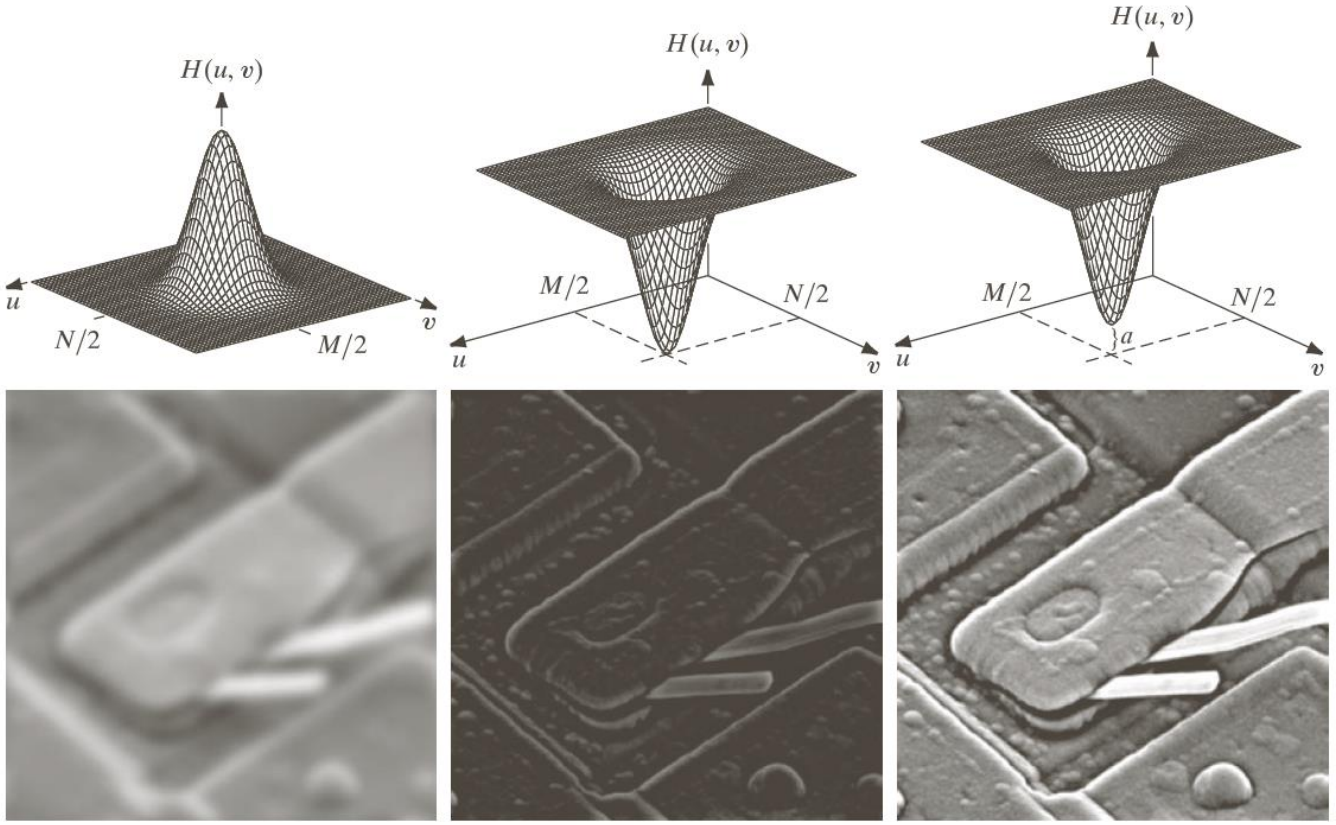
Image of damaged integrated circuit



Fourier spectrum



F(0,0)=0




a b c
d e f

**FIGURE 4.31** Top row: frequency domain filters. Bottom row: corresponding filtered images obtained using Eq. (4.7-1). We used $a = 0.85$ in (c) to obtain (f) (the height of the filter itself is 1). Compare (f) with Fig. 4.29(a).

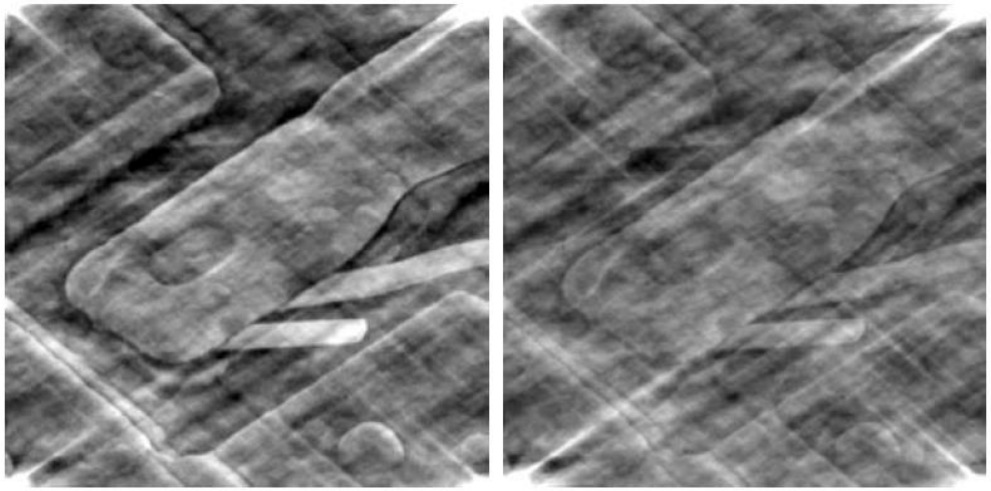
The DFT is a complex array of the form:

$$F(u,v) = R(u,v) + i\, I(u,v)$$

$$g(x,y) = \mathcal{F}^{-1}\left[H(u,v)R(u,v) + i\, H(u,v)I(u,v)\right]$$

The phase angle is not altered by filtering in this way. Filters that affect the real and the imaginary parts equally, and thus have no effect on the phase are called *zero-phase-shift* filters.

Even small changes in the phase angle can have undesirable effects on the filtered output.



a b

**FIGURE 4.35**
(a) Image resulting from multiplying by 0.5 the phase angle in Eq. (4.6-15) and then computing the IDFT. (b) The result of multiplying the phase by 0.25. The spectrum was not changed in either of the two cases.

## Main Steps for Filtering in the Frequency Domain

1. Given an input image $f(x,y)$ of size $M \times N$, obtain the padding parameters $P$ and $Q$ (usually $P=2M$ , $Q=2N$)
2. Form a padded image $f_p(x,y)$, of size $P \times Q$ by appending the necessary numbers of zeros to $f(x,y)$ ($f$ is in the upper left corner of $f_p$)
3. $f_p(x,y) = (-1)^{x+y} f_p(x,y)$ - to center its transform
4. Compute the DFT, $F(u,v)$, of the image obtain from 3.
5. Generate a real, symmetric filter function $H(u,v)$ of size $P \times Q$ with center at coordinates $\left(\frac{P}{2}, \frac{Q}{2}\right)$. Compute the array product

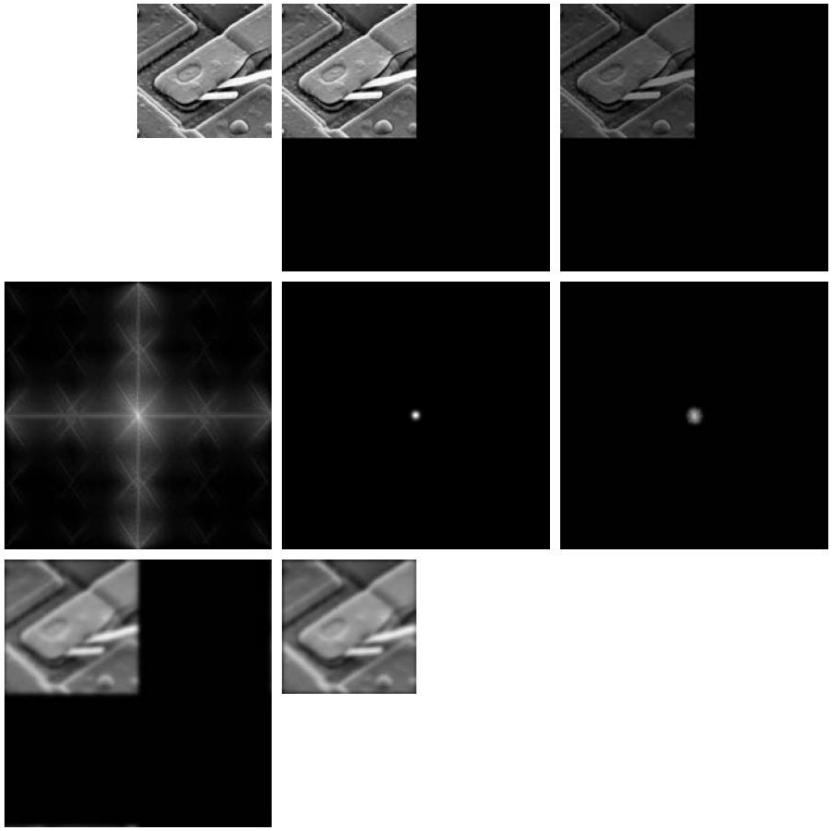
$$G(u,v) = H(u,v)F(u,v)$$

6. Obtain the processed image:

$$g_p(x,y) = \left\{ \text{real}\left[ \mathcal{F}^{-1}\left( G(u,v) \right) \right] \right\} (-1)^{x+y}$$

The real part is selected in order to ignore parasitic complex components resulting from computational inaccuracies.

7. Obtain the output, filtered image, ***g(x,y)*** by extracting the ***M×N*** region from the top, left corner of $g_p(x,y)$.



a b c
d e f
g h

**FIGURE 4.36**
(a) An $M \times N$ image, $f$.
(b) Padded image, $f_p$ of size $P \times Q$.
(c) Result of multiplying $f_p$ by $(-1)^{x+y}$.
(d) Spectrum of $F_p$. (e) Centered Gaussian lowpass filter, $H$, of size $P \times Q$.
(f) Spectrum of the product $HF_p$.
(g) $g_p$, the product of $(-1)^{x+y}$ and the real part of the IDFT of $HF_p$.
(h) Final result, $g$, obtained by cropping the first $M$ rows and $N$ columns of $g_p$.

**Course 5**

## Correspondence between Filtering in the Spatial and Frequency Domains

The link between filtering in the spatial domain and frequency domain is the convolution theorem.

Given a filter ***H(u,v)***, suppose that we want to find its equivalent representation in the spatial domain.

$$f(x,y)=\delta(x,y)\Rightarrow F(u,v)=1$$

$$g(x,y)=\mathcal{F}^{-1}\left[H(u,v)F(u,v)\right]\Rightarrow h(x,y)=\mathcal{F}^{-1}\left[H(u,v)\right]$$

The inverse transform of the frequency domain filter, ***h(x,y)*** is the corresponding filter in the spatial domain.

Conversely, given a spatial filter, ***h(x,y)*** we obtain its frequency domain representation by taking the Fourier transform of the spatial filter:

$$h(x,y) \leftrightarrow H(u,v)$$

***h(x,y)*** is sometimes called as the *(finite) impulse response* (FIR) of ***H(u,v)***.

One way to take advantage of the properties of both domains is to specify a filter in the frequency domain, compute its IDFT, and then use the resulting, full-size spatial filter as a guide for constructing smaller spatial filter masks.

Let ***H(u)*** denote the 1-D frequency domain Gaussian filter:

$$H(u) = Ae^{-\frac{u^2}{2\sigma^2}} \quad , \quad \sigma - \text{ the standard deviation}$$

The corresponding filter in the spatial domain is obtained by taking the inverse Fourier transform of ***H(u)***:

$$h(x) = \sqrt{2\pi}\sigma A e^{-2\pi^2\sigma^2 x^2}$$

which is also a Gaussian filter. When ***H(u)*** has a broad profile (large value of $\sigma$), ***h(x)*** has a narrow profile and vice versa. As $\sigma$ approaches infinity, ***H(u)*** tends toward a constant function and ***h(x)*** tends towards an impulse, which implies no filtering in the frequency and spatial domains.

**Course 5**

## Image Smoothing Using Frequency Domain Filters

Smoothing (blurring) is achieved in the frequency domain by high-frequency attenuation that is by *lowpass* filtering. We consider three types of lowpass filters:

*ideal*, *Butterworth*, *Gaussian*

The Butterworth filter has a parameter called the *filter order*. For high order values, the Butterworth filter approaches the ideal filter and for low values is more like a Gaussian filter.

All filters and images in these sections are consider padded with zeros, thus they have size $\boldsymbol{P \times Q}$. The Butterworth filter may be viewed as providing a transition between the other two filters.

**Ideal Lowpass Filters** (ILPF)

$$H(u,v)=\begin{cases}1 & \text{if } D(u,v)\le D_0\\ 0 & \text{if } D(u,v)> D_0\end{cases}$$

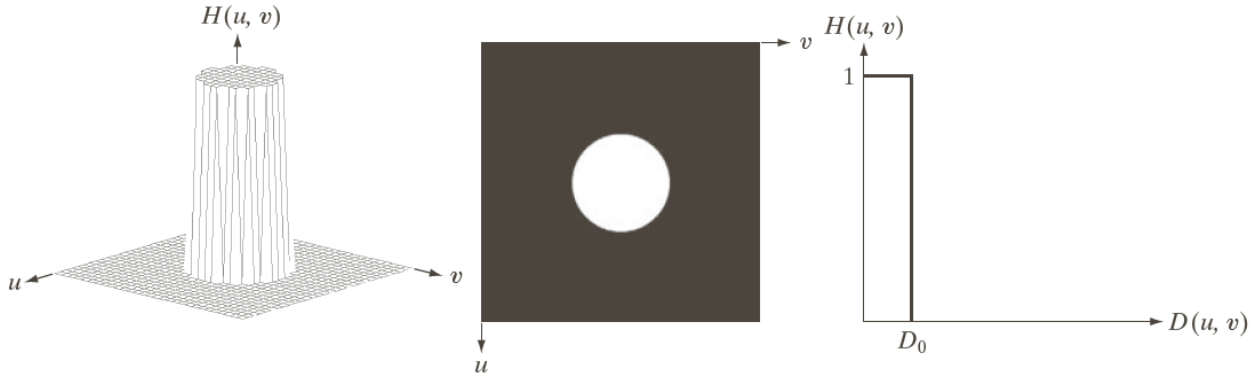
Where $D_0 \ge 0$ is a positive constant and $D(u,v)$ is the distance between $(u,v)$ and the center of the frequency rectangle:

$$D(u,v)=\sqrt{\left(u-\frac{P}{2}\right)^2+\left(v-\frac{Q}{2}\right)^2} \qquad \text{(DUV)}$$

The name *ideal* indicates that all frequencies on or inside the circle of radius $D_0$ are passed without attenuation, whereas all frequencies outside the circle are completely eliminated (filtered out).

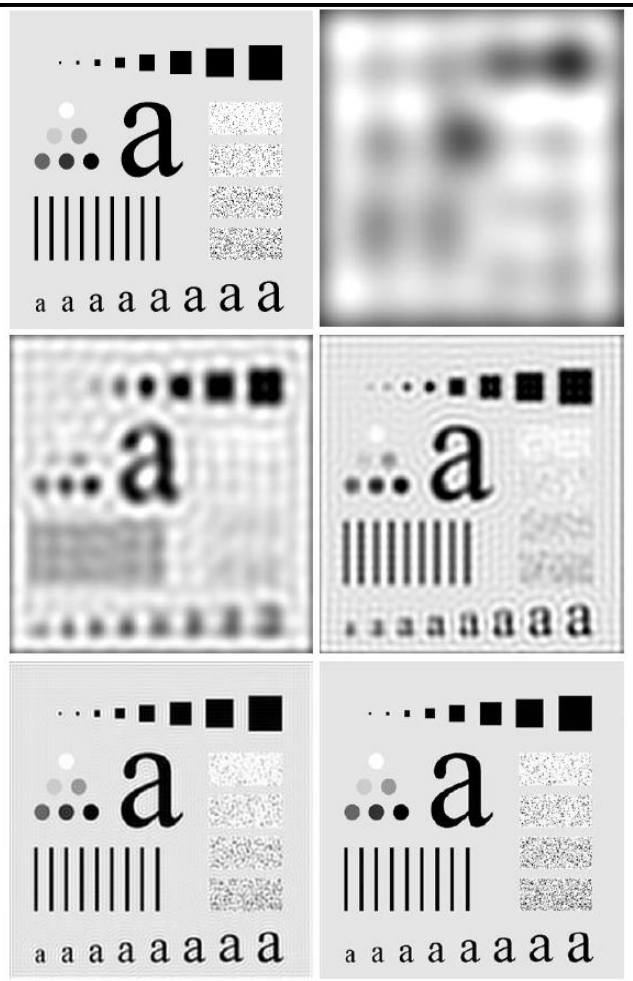
For an ILPF cross section, the point of transition between ***H(u,v)=1*** and ***H(u,v)=0*** is called the *cutoff frequency*. The sharp cutoff frequencies of an ILPF cannot be realized with electronic components, but they can be simulated in a computer.

We can compare the ILPF by studying their behavior with respect to the cutoff frequencies.




a b c

**FIGURE 4.40** (a) Perspective plot of an ideal lowpass-filter transfer function. (b) Filter displayed as an image. (c) Filter radial cross section.



a b
c d
e f

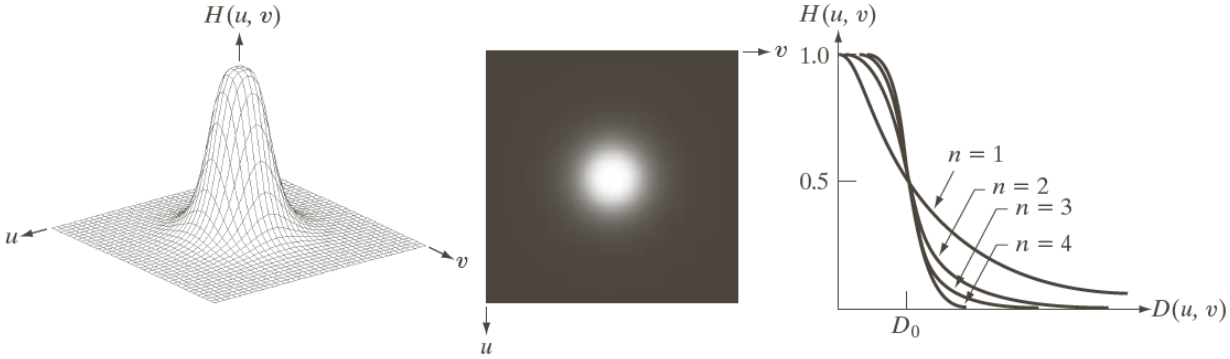
**FIGURE 4.42** (a) Original image. (b)–(f) Results of filtering using ILPFs with cutoff frequencies set at radii values 10, 30, 60, 160, and 460, as shown in Fig. 4.41(b). The power removed by these filters was 13, 6.9, 4.3, 2.2, and 0.8% of the total, respectively.

**Butterworth Lowpass Filter** (BLPF)

The transfer function of a Butterworth lowpass filter of order ***n*** and with cutoff frequency at distance $D_0$ from the origin is:

$$H(u,v) = \frac{1}{1 + \left[\frac{D(u,v)}{D_0}\right]^{2n}}$$

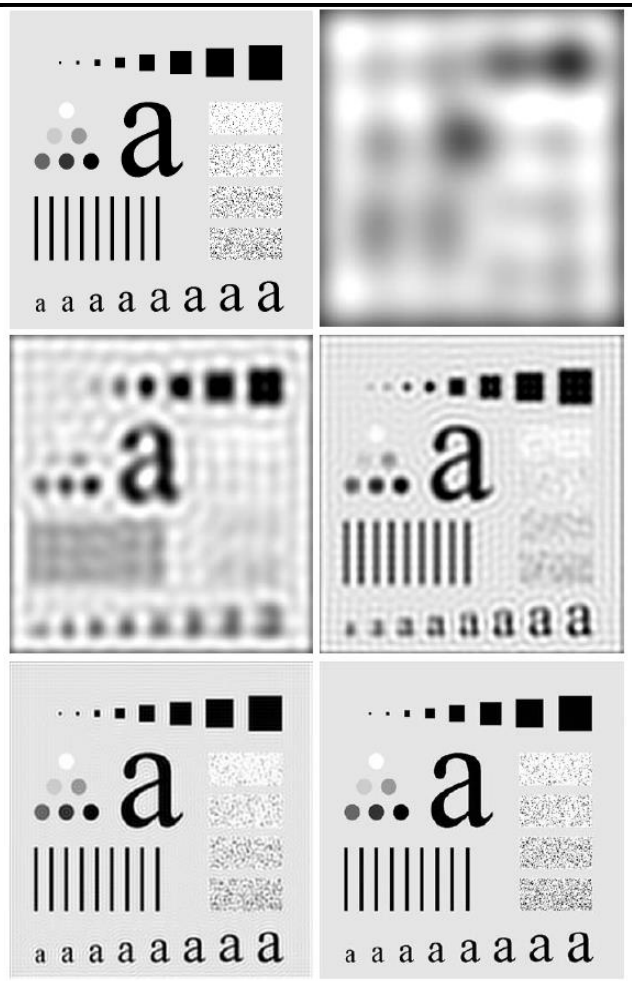
where ***D(u,v)*** is given by the relation (DUV).




a b c

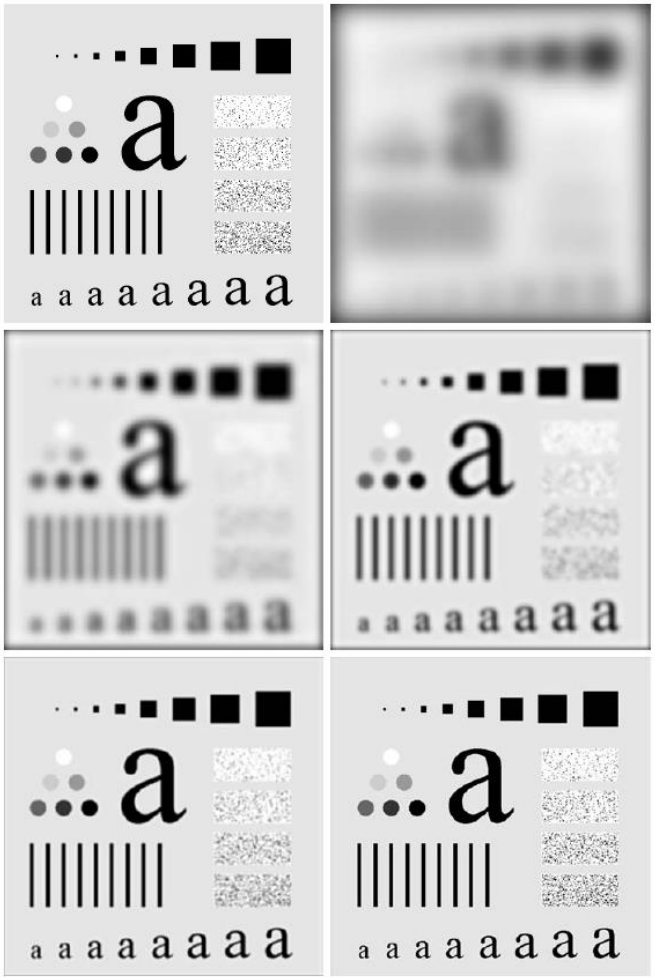
**FIGURE 4.44** (a) Perspective plot of a Butterworth lowpass-filter transfer function. (b) Filter displayed as an image. (c) Filter radial cross sections of orders 1 through 4.

The BLPF transfer function does not have a sharp discontinuity that gives a clear cutoff between passed and filtered frequencies. For filters with smooth transfer functions, defining a cutoff frequency locus is made at points for which ***H(u,v)*** is down to a certain fraction of its maximum value.



a b
c d
e f

**FIGURE 4.42** (a) Original image. (b)–(f) Results of filtering using ILPFs with cutoff frequencies set at radii values 10, 30, 60, 160, and 460, as shown in Fig. 4.41(b). The power removed by these filters was 13, 6.9, 4.3, 2.2, and 0.8% of the total, respectively.



a b
c d
e f

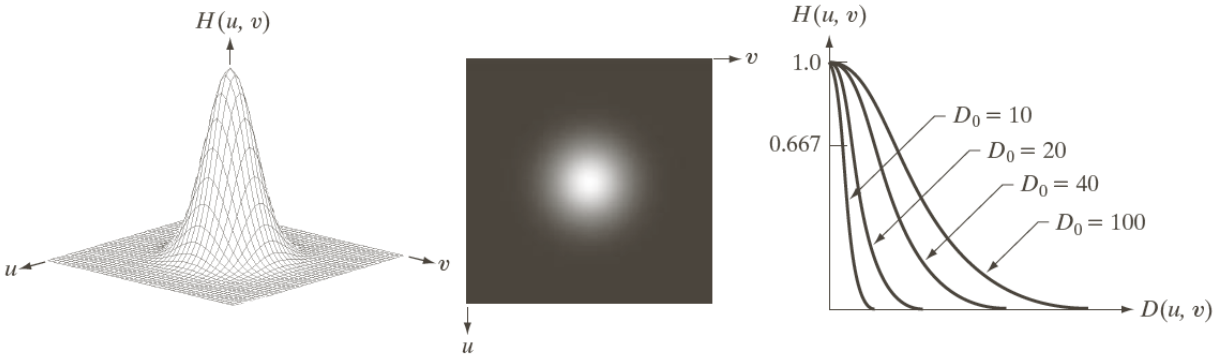
**FIGURE 4.45** (a) Original image. (b)–(f) Results of filtering using BLPFs of order 2, with cutoff frequencies at the radii shown in Fig. 4.41. Compare with Fig. 4.42.

**Course 5**

## **Gaussian Lowpass Filter** (GLPF)

$$H(u,v) = e^{-\frac{D^2(u,v)}{2\sigma^2}} = e^{-\frac{D^2(u,v)}{2D_0^2}}$$

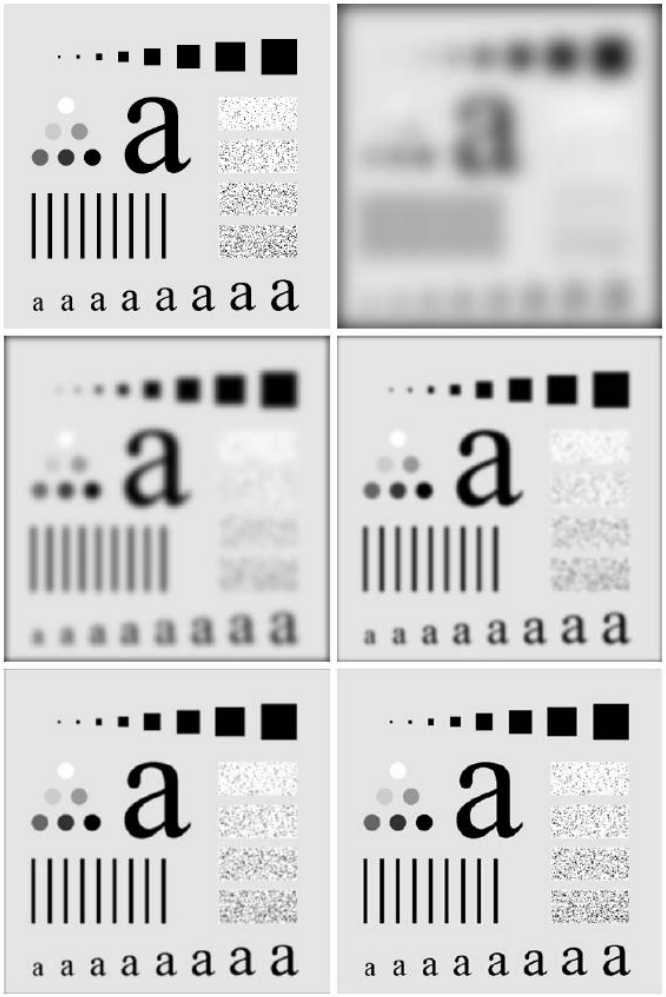
$D_0$ is the cutoff frequency. When $D(u,v) = D_0$ the GLPF is down to $0.607$ of its maximum value.




a b c

**FIGURE 4.47** (a) Perspective plot of a GLPF transfer function. (b) Filter displayed as an image. (c) Filter radial cross sections for various values of $D_0$.

## Course 5



a b
c d
e f

**FIGURE 4.48** (a) Original image. (b)–(f) Results of filtering using GLPFs with cutoff frequencies at the radii shown in Fig. 4.41. Compare with Figs. 4.42 and 4.45.

**TABLE 4.4**
Lowpass filters. $D_0$ is the cutoff frequency and $n$ is the order of the Butterworth filter.

| Ideal | Butterworth | Gaussian |
|---|---|---|
| $H(u,v) = \begin{cases} 1 & \text{if } D(u,v) \le D_0 \\ 0 & \text{if } D(u,v) > D_0 \end{cases}$ | $H(u,v) = \dfrac{1}{1 + [D(u,v)/D_0]^{2n}}$ | $H(u,v) = e^{-D^2(u,v)/2D_0^2}$ |

**Course 5**

## Image Sharpening Using Frequency Domain Filters

Edges and other abrupt changes in intensities are associated with high-frequency components, image sharpening can be achieved in the frequency domain by highpass filters, which attenuates the low-frequency components without changing the high-frequency information in the Fourier transform.

A highpass filter is obtained from a given lowpass filter using the equation:

$$H_{HP}(u,v) = 1 - H_{LP}(u,v)$$

where $H_{LP}(u,v)$ is the transfer function of a lowpass filter.

# Course 5

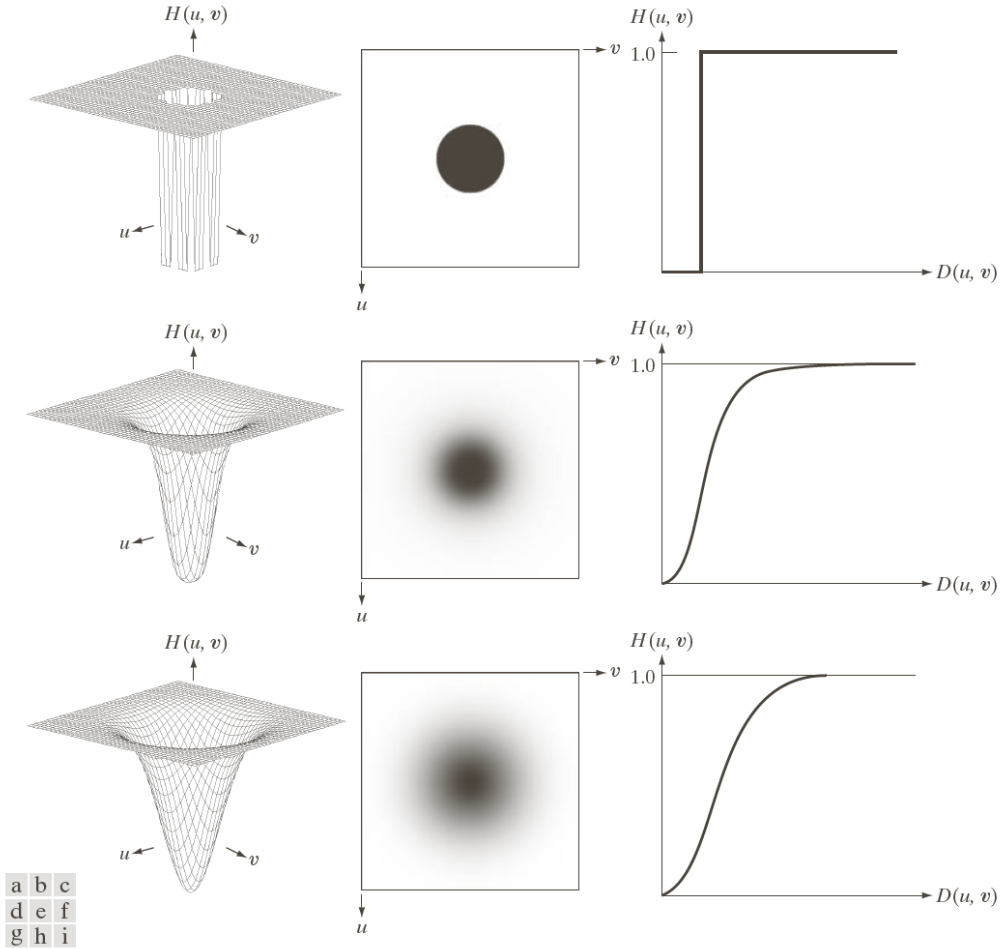



**FIGURE 4.52** Top row: Perspective plot, image representation, and cross section of a typical ideal highpass filter. Middle and bottom rows: The same sequence for typical Butterworth and Gaussian highpass filters.

## Ideal Highpass Filter

A 2-D *ideal highpass filter* (IHPF) is defined as:

$$H(u,v) = \begin{cases} 0 & \text{if } D(u,v) \le D_0 \\ 1 & \text{if } D(u,v) > D_0 \end{cases}$$

where $D_0$ is the cutoff frequency and $D(u,v)$ is given by equation (DUV). As for ILPF, the IHPF is not physically realizable.

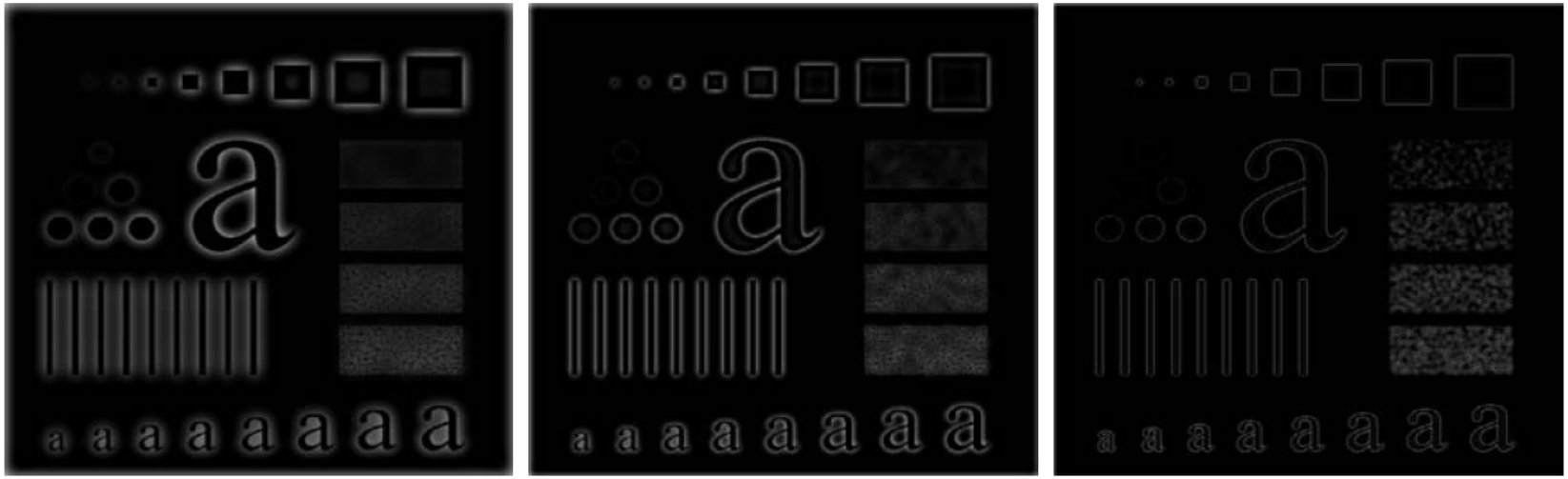
a b c

**FIGURE 4.54** Results of highpass filtering the image in Fig. 4.41(a) using an IHPF with $D_0 = 30$, 60, and 160.

**Butterworth Highpass Filter** (BHPF)

The transfer function of a Butterworth highpass filter of order ***n*** and with cutoff frequency at distance $D_0$ from the origin is:

$$H(u,v) = \frac{1}{1 + \left[\frac{D_0}{D(u,v)}\right]^{2n}}$$



a b c

**FIGURE 4.55** Results of highpass filtering the image in Fig. 4.41(a) using a BHPF of order 2 with $D_0 = 30, 60,$ and 160, corresponding to the circles in Fig. 4.41(b). These results are much smoother than those obtained with an IHPF.

## **Gaussian Highpass Filter** (GLPF)

$$H(u,v) = 1 - e^{-\frac{D^2(u,v)}{2D_0^2}}$$

a b c

**FIGURE 4.56** Results of highpass filtering the image in Fig. 4.41(a) using a GHPF with $D_0$ = 30, 60, and 160, corresponding to the circles in Fig. 4.41(b). Compare with Figs. 4.54 and 4.55.

**TABLE 4.5**
Highpass filters. $D_0$ is the cutoff frequency and $n$ is the order of the Butterworth filter.

| Ideal | Butterworth | Gaussian |
|---|---|---|
| $H(u,v) = \begin{cases} 1 & \text{if } D(u,v) \le D_0 \\ 0 & \text{if } D(u,v) > D_0 \end{cases}$ | $H(u,v) = \dfrac{1}{1 + [D_0/D(u,v)]^{2n}}$ | $H(u,v) = 1 - e^{-D^2(u,v)/2D_0^2}$ |

a b c

**FIGURE 4.57** (a) Thumb print. (b) Result of highpass filtering (a). (c) Result of thresholding (b). (Original image courtesy of the U.S. National Institute of Standards and Technology.)

Figure 4.57(a) is a ***1026×962*** image of a thumb print in which smudges are present. A keystep in automated figerprint recognition is enhancement of print ridges and the reduction of smudges. In this example a highpass filter was used to enhance ridges and reduce the effects of smudging. Enhancement of the ridges is accomplished by the fact that they contain high frequencies, which are unchanged by a highpass filter. This filter reduces low frequency components which correspond to slowly varying intensities in the image, such as background and smudges.

Figure 4.57(b) is the result of using a BHPF of order ***n=4***, with a cutoff frequency $\boldsymbol{D_0=50}$.

Figure 4.57(c) is the result of setting to black all negative values and to white all positive values in Figure 4.57(b) (a threshold intensity transformation)

## The Laplacian in the Frequency Domain

The Laplacian can be implemented in the frequency domain using the filter:

$$H(u,v) = -4\pi^2\left(u^2 + v^2\right)$$

The centered Laplacian is:

$$H(u,v) = -4\pi^2\left[\left(u - \frac{P}{2}\right)^2 + \left(v - \frac{Q}{2}\right)^2\right] = -4\pi^2 D^2(u,v)$$

The Laplacian image is obtained as:

$$\nabla^2 f(x,y) = \mathcal{F}^{-1}\left\{H(u,v)F(u,v)\right\}$$

Enhancement is obtained with the equation:

$$g(x,y) = f(x,y) - \nabla^2 f(x,y) \qquad (1)$$

Computing $\nabla^2 f(x,y)$ with the above relation introduces DFT scaling factors that can be several orders of magnitude larger than the maximum value of ***f***. To fix this problem, we normalize the values of ***f(x,y)*** to the range ***[0,1]*** (before computing its DFT) and divide $\nabla^2 f(x,y)$ by its maximum value which will bring it to ***[-1,1]***.

## Course 5

### Spatial Correlation and Convolution

*Correlation* is the process of moving a filter mask over the image and computing the sum of products at each location. *Convolution* is similar with correlation, except that the filter is first rotated by ***180°***.

*Correlation*

$$w(x,y) \lozenge f(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x+s, y+t)$$

*Convolution*

$$w(x,y) \blacklozenge f(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x-s, y-t)$$

A function that contains a single ***1*** and the rest being ***0***s is called a *discrete unit impulse*. Correlating a function with a discrete unit impulse produces a rotated version of the filter at the location of the impulse.

# Types of Noise and Their Characteristics

Noise = a variation of the signal from its true value by a small (random) amount due to external or internal factors in the image processing pipeline. (Solomon & Breckon, Fundamentals of Digital Image Processing)

Noise = unwanted information in the image

The noise may be correlated or uncorrelated, signal dependent or independent, and so on. The knowledge about the imaging system and the visual perception of the image helps in

generating the noise model and estimating of the statistical characteristics of noise embedded in an image is important because it helps separating the noise from the useful image signal. We consider four classes of noise:

1. **Additive noise**: sometimes the noise generated from sensors are normal white Gaussian, which is essentially additive and signal independent,

$$g(x,y) = f(x,y) + \eta(x,y).$$

2. **Multiplicative noise**: the graininess noise from photographic plates is usually multiplicative.

$$g(x,y) = f(x,y) * \eta(x,y).$$

3. **Impulse noise**: often the noisy sensors genrate impulse noise:

$$g(x,y) = (1-p)\, f(x,y) + p\, i(x,y),$$

where $i(x,y)$ is the impulsive noise and $p$ is a binary parameter. The impulse noise can be easily detected from noisy images because of the contrast anomalies.

4. **Quantization noise**, is essentially an image dependent noise. This noise is characterized by the size of signal quantization interval. Such noise produces image-like artifacts and may produce false contours around the objects. The quantization noise also removes the image details which are of low contrast.

Two dimensional images may be degraded due to several reasons, e.g.

- Imperfection of the imaging system
- Imperfection in the transmission channel
- Degradation due to atmospheric conditions
- Degradation due to relative motion between the object and the camera

## Noise Models

$$g(x,y) = f(x,y) + \eta(x,y)$$

The main sources of noise in digital images arise during image acquisition and/or transmission (environmental conditions during image acquisition, the quality of the sensors).

Parameters that define the spatial characteristics of the noise and whether the noise is correlated with the image are important properties to be studied. We assume that the noise

is independent of spatial coordinates and that it is uncorrelated with the image itself (i.e. there is no correlation between pixel values and the values of noise components).

## Some Important Noise Probability Density Functions

The noise may be considered a random variable, characterized by a probability denisty function (pdf).

### Gaussian noise

$$p(z) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(z-\bar{z})^2}{2\sigma^2}}$$

where $z$ represents intensity, $\bar{z}$ is the mean value, and $\sigma$ is its standard deviation, $\sigma^2$ is called variance of $z$.

## Rayleigh noise

$$p(z)=\begin{cases}\dfrac{2}{b}(z-a)e^{-\frac{(z-a)^2}{b}} & \text{for } z\geq a\\ 0 & \text{for } z<a\end{cases}$$

The mean and variance for this pdf are:

$$\bar{z}=a+\sqrt{\frac{\pi b}{4}}$$

$$\sigma^2=\frac{b(4-\pi)}{4}$$

## Erlang (gamma) noise

$$p(z)=\begin{cases}\dfrac{a^b z^{b-1}}{(b-1)!}e^{-az} & \text{for } z\geq 0\\ 0 & \text{for } z<0\end{cases}, \quad a,b>0,\ b\in\mathbb{N}$$

$$\bar{z}=\frac{b}{a}$$

$$\sigma^2=\frac{b}{a^2}$$

## Exponential noise

$$p(z) = \begin{cases} ae^{-az} & \text{for } z \geq 0 \\ 0 & \text{for } z < 0 \end{cases}, \quad a > 0$$

$$\bar{z} = \frac{1}{a}$$

$$\sigma^2 = \frac{1}{a^2}$$

(Erlang $b=1$)

## Uniform noise

$$p(z)=\begin{cases}\dfrac{1}{b-a} & \text{for } a\le z\le b\\ 0 & \text{otherwise}\end{cases},$$

$$\bar{z}=\frac{a+b}{2}$$

$$\sigma^2=\frac{(b-a)^2}{12}$$

## Impulse (salt-and-pepper) noise

The pdf of *(bipolar) impulse* noise is given by

$$p(z) = \begin{cases} P_a & \text{for } z = a \\ P_b & \text{for } z = b \\ 0 & \text{otherwise} \end{cases}$$

$b > a$ – intensity $b$ appear as a light dot in the image

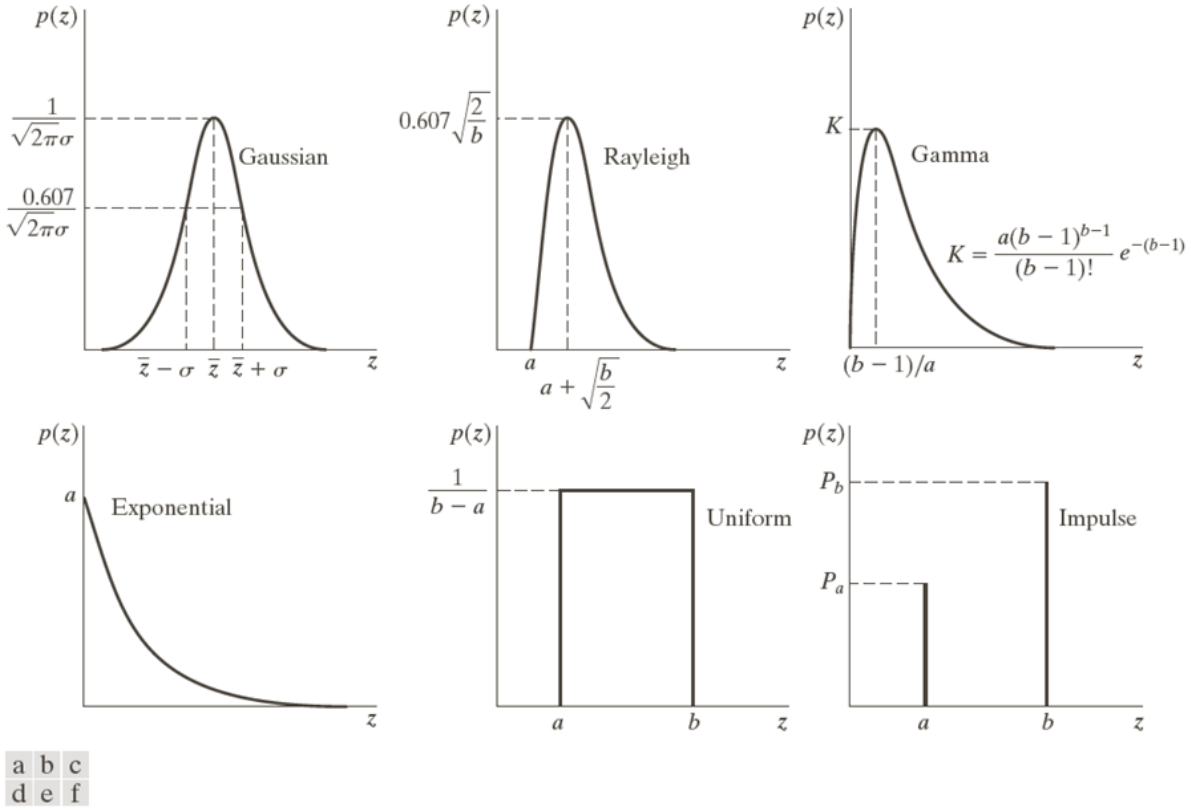
$b < a$ – intensity $b$ appear as a dark dot in the image

$P_a = 0$ or $P_b = 0$ the impule noise is called *unipolar*

$P_a \approx P_b$ - impulse noise values will ressemble salt and pepper granules randomly distributed over the image. For this reason, bipolar impulse noise is called also *salt-and-pepper* noise.

Noise impulses can be negative or positive. Because impulse corruption usually is large compared with the strength of the image signal, impulse noise generally is digitized as extreme (pure black or white) values in an image. Thus, the assumption is that $a$ and $b$ are equal to the minimum and maximum allowed values in the digitized image. As a result,

negative impulses appear as black (pepper) points in an image, and positive impulses appear as white (salt) points.

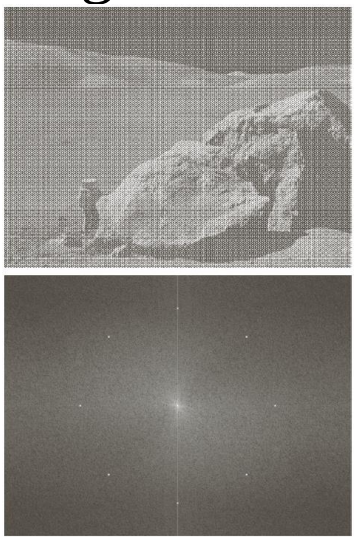



a b c
d e f

Some important probability density functions.

# Periodic noise

Periodic noise arises from electrical or electromechanical interference during image acquisition. This type of noise is spatially dependent and can be reduced significantly via frequency domain filtering.



a
b

**FIGURE 5.5** (a) Image corrupted by sinusoidal noise. (b) Spectrum (each pair of conjugate impulses corresponds to one sine wave). (Original image courtesy of NASA.)

Figure 5.5(a) is corrupted by sinusoidal noise of various frequencies. The Fourier transform of a pure sinusoid is a pair of conjugate impulses located at the conjugate frequencies of the sine wave. If the amplitude of a sine wave in the spatial domain is strong enough, we would expect to see in the spectrum of the image a pair of impulses for each sine wave in the image. In Figure 5.5(b) we can see the impulses appearing in a circle.

Gaussian noise – electronic circuit noise, sensor noise due to poor illumination and/or high temperature

Rayleigh noise – noise in radar range imaging

Exponential and gamma noise – laser imaging

Salt & pepper noise – „sharp and sudden disturbances in the image signal” (Wiki)

Uniform noise – theoretical use

## Estimation of Noise Parameters

The parameters of periodic noise are estimated by inspection of the Fourier spectrum of the image. Sometimes it is possible to deduce the periodicity of noise just by looking at the image.

The parameters of noise pdf's may be known partially from sensors specifications. If the image system is available, one simple way to study the characteristics of system noise is to capture a set of images of „flat" environments (in the case of

an optical sensor, this means taking images of a solid gray board that is illuminated uniformly). The resulting images are good indicators of system noise.

When only images already generated by a sensor ar available, frequently it is possible to estimate the parameters of the pdf from small portions of the image that are of constant background intensity.

The simplest use of the data from the image strips is for calculating the mean and the variance of intensity levels.

Consider a subimage $S$ and let $p_S(z_i)$, $i=0,1,2,\ldots,L-1$ denote the probability estimates (normalized histogram values) of the intensities of the pixels in $S$, where $L$ is the number of possible intensities in the entire image. We estimate the mean and the variance of the pixels in $S$:

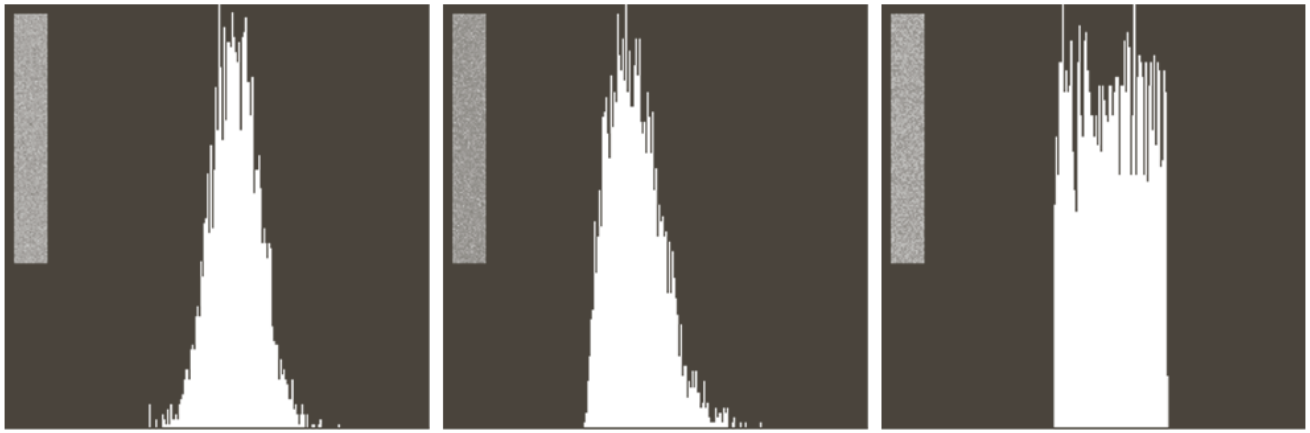
$$\bar{z} = \sum_{i=0}^{L-1} z_i \, p_S(z_i)$$

$$\sigma^2 = \sum_{i=0}^{L-1} (z_i - \bar{z})^2 \, p_S(z_i)$$

The shape of the histogram identifies the closest pdf match. If the shape is almost Gaussian then the mean and the variance are all we need. For the other shapes, we use the mean and the variance to solve for parameters ***a*** and ***b***.

Impulse noise is handled differently because the estimate needed is of the actual probability of occurrence of white and black pixels. Obtaining this estimate requires that both black and white pixels be visible, so a midgray, relatively constant area is needed in the image in order to be able to compute a

histogram. The heights of the peaks corresponding to black and white pixels are the estimates of $\boldsymbol{P_a}$ and $\boldsymbol{P_b}$.



a b c

**FIGURE 5.6** Histograms computed using small strips (shown as inserts) from (a) the Gaussian, (b) the Rayleigh, and (c) the uniform noisy images in Fig. 5.4.

# Denoising filters

## Mean Filters

Suppose $S_{xy}$ represent a recatngular neighborhood of $m \times n$ size centered at point $(x,y)$.

### Arithmetic mean filter

$$\hat{f}(x,y) = \frac{1}{m \cdot n} \sum_{(s,t) \in S_{xy}} g(s,t)$$

A mean filter smooths local variations of an image and noise is reduced as a result of blurring.

## Geometric mean filter

$$\hat{f}(x,y)=\left[\prod_{(s,t)\in S_{xy}} g(s,t)\right]^{\frac{1}{m\cdot n}}$$

A geometric mean filter achieves smoothing comparable to the arithmetic mean filter, but it tends to lose less image detail.

## Harmonic mean filter

$$\hat{f}(x,y) = \frac{m \cdot n}{\sum_{(s,t)\in S_{xy}} \frac{1}{g(s,t)}}$$

Harmonic mean filter works well for salt noise, but fails for pepper noise. It also works well on Gaussian noise.

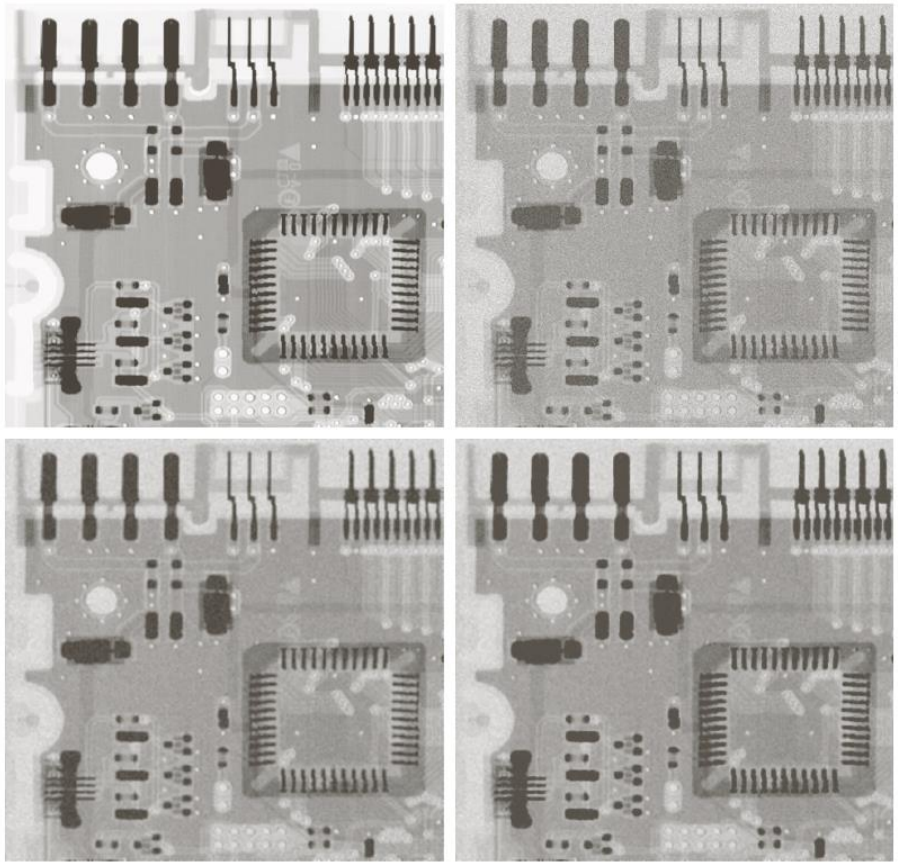
## Contraharmonic mean filter

$$\hat{f}(x,y) = \frac{\sum_{(s,t)\in S_{xy}} g(s,t)^{Q+1}}{\sum_{(s,t)\in S_{xy}} g(s,t)^{Q}} \quad , \quad Q - \text{the order of the filter}$$

This filter is good for reducing or virtually eliminating the effects of salt-and-pepper noise.

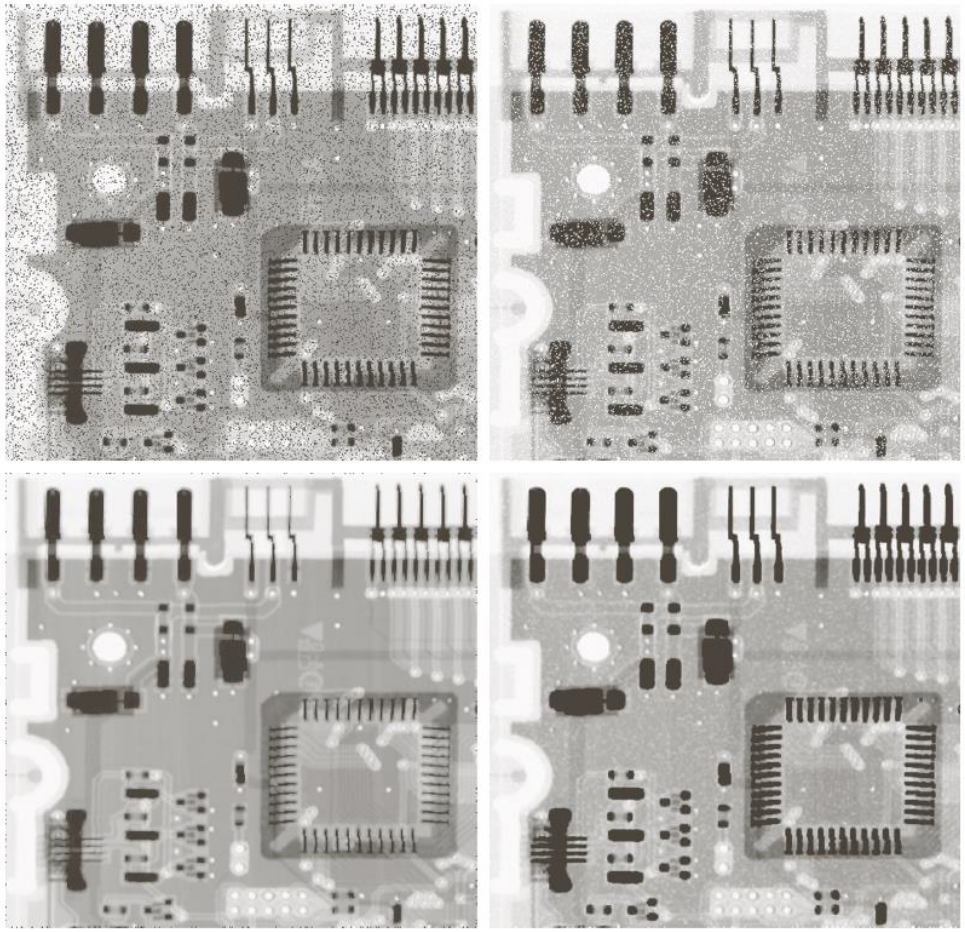
For $Q > 0$ the filter eliminates pepper noise, for $Q < 0$ the filter eliminates salt noise, but it cannot do both simultaneously.

$Q = 0$ – arithmetic mean filter, $Q = -1$ – harmonic mean filter



a b
c d

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

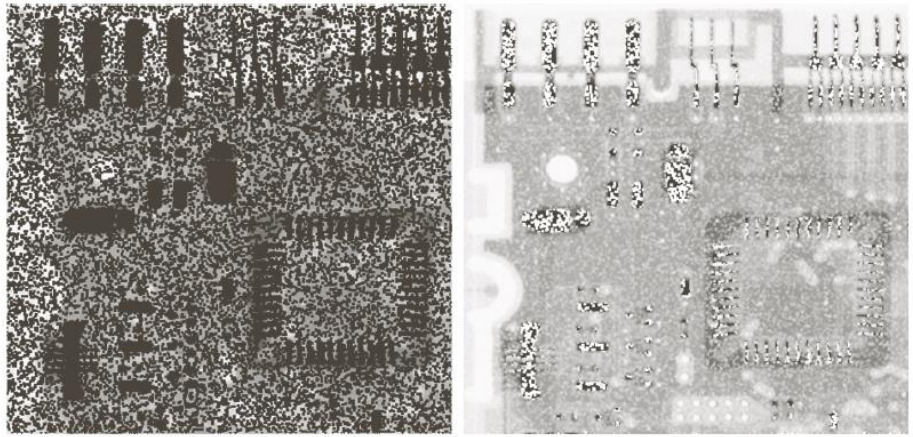


a b
c d

**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 $3 \times 3$ contra-harmonic filter of order 1.5. (d) Result of filtering (b) with $Q = -1.5$.

a b

**FIGURE 5.9**
Results of selecting the wrong sign in contraharmonic filtering.
(a) Result of filtering Fig. 5.8(a) with a contraharmonic filter of size $3 \times 3$ and $Q = -1.5$.
(b) Result of filtering 5.8(b) with $Q = 1.5$.



# Order-Statistic Filters

## Median filter

$$\hat{f}(x,y) = \text{median}\{g(s,t);(s,t) \in S_{xy}\}$$

Median filters have excellent noise-reduction capabilities, with less blurring than linear smoothing filters. Median filters are particularly effective in the presence of bipolar and unipolar impulse noise.

## Max and min filters

$$\hat{f}(x,y) = \max\{g(s,t);(s,t) \in S_{xy}\}$$

This filter is useful for finding the brightest points in an image. This filter reduces pepper noise.

$$\hat{f}(x,y) = \min\{g(s,t);(s,t) \in S_{xy}\}$$

This filter is useful for finding the darkest points in an image. This filter reduces salt noise.

## Midpoint filter

$$\hat{f}(x,y)=\frac{1}{2}\left[\max\{g(s,t);(s,t)\in S_{xy}\}+\min\{g(s,t);(s,t)\in S_{xy}\}\right]$$

It works best for randomly distributed noise, like Gaussian or uniform noise.