Digital Image Processing Using MATLAB

Malay S. Bhatt
Department of Computer Engineering
Faculty of Technology
Dharmsinh Desai University
Nadiad

IMAGE RESTORATION

The main objective of restoration is to improve the quality of a digital image which has been degraded due to Various phenomena like:

- Motion
- Improper focusing of Camera during image acquisition.
- Atmospheric turbulence
- Noise

Enhancement versus Restoration

- Both processes try to improve an image in some predefined sense
- Image enhancement is largely a subjective process, while image restoration is for the most part an objective process

Enhancement:

(1)Manipulating an image in order to take advantage of the psychophysics of the human visual system.

(2)Techniques are usually “heuristic.”

(3)Example: Contrast stretching, histogram equalization.

- **Restoration**:

  (1)A process that attempts to reconstruct or recover an image that has been degraded by using some prior knowledge of the degradation phenomenon.

  (2)Involves modeling the degradation process and applying the inverse process to recover the original image.

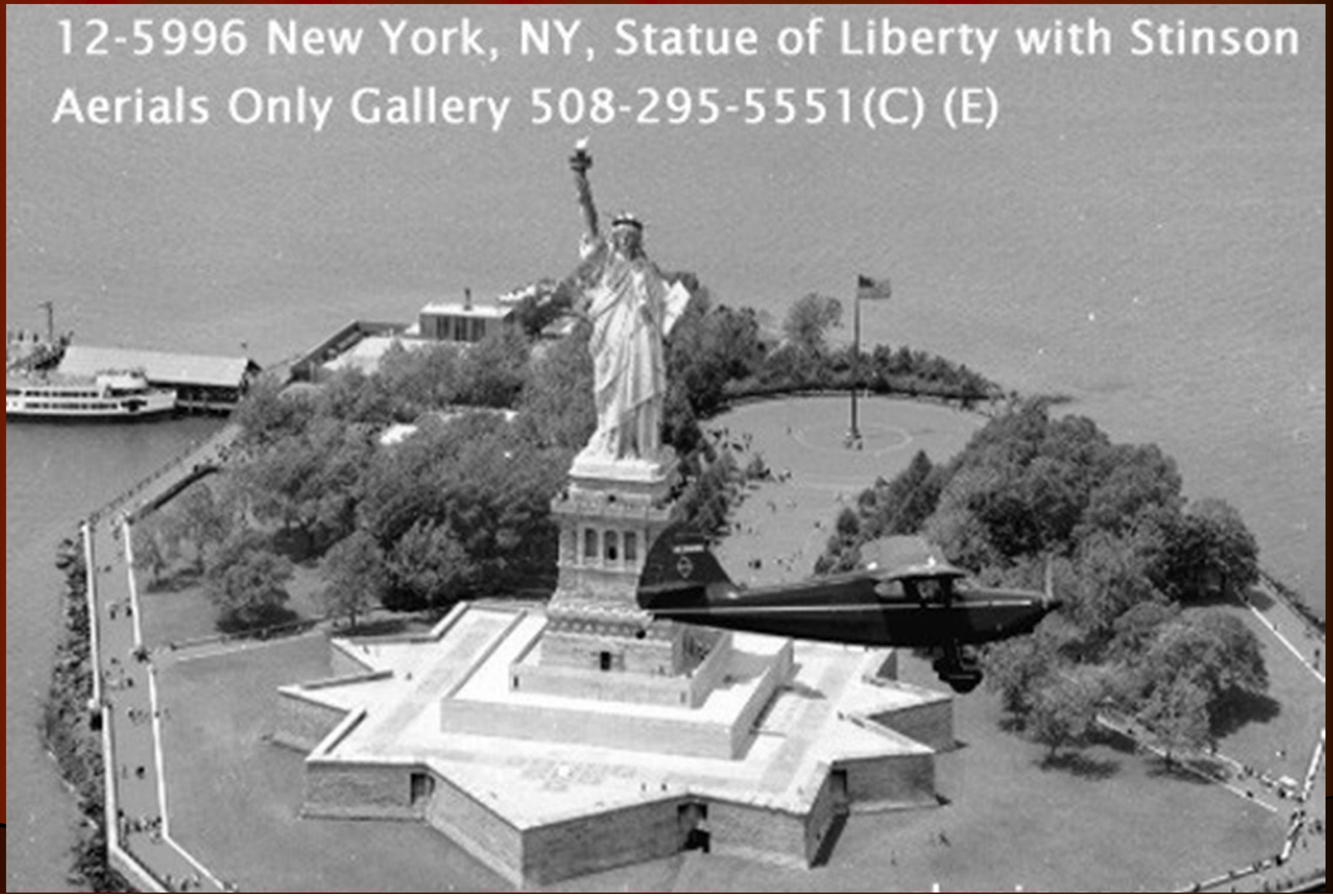
  (3)A criterion for “goodness” is required that will recover the image in an optimal fashion with respect to that criterion.

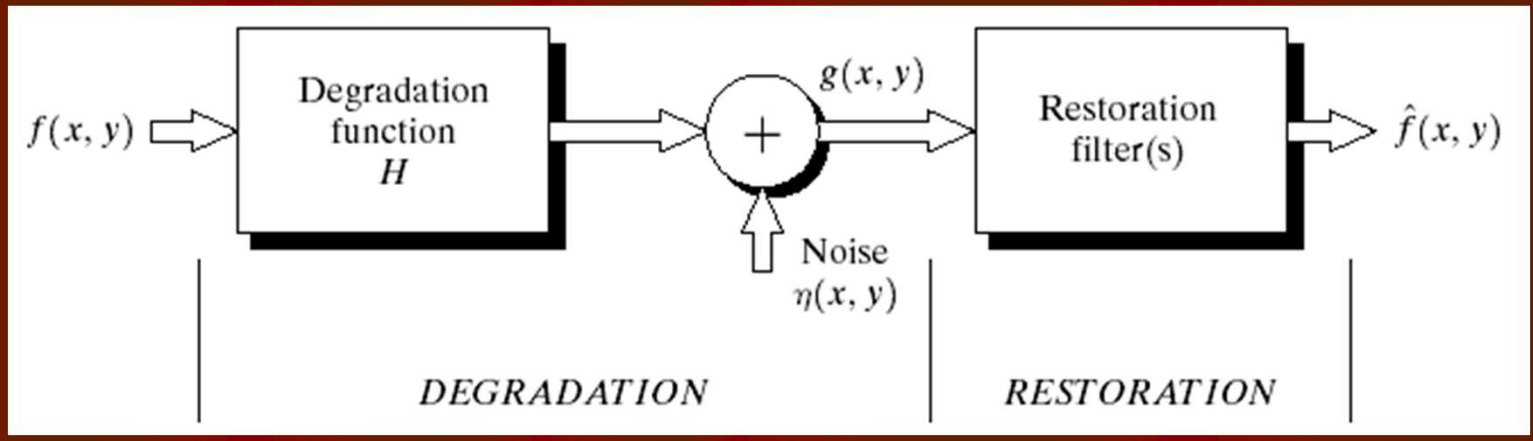
  (4) Example: removal of blur by applying a deblurring function.

Problem:

- You want to know some image X.
- But you only have a corrupted version Y .
- How do you determine X from Y ?

Blurring due to uniform motion





Degradation model:

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

where $h(x,y)$ is a system that causes image distortion and $\eta(x,y)$ is noise.

2-D Convolution (Spatial)

If H is a linear, position-invariant process, then the degraded image is given in the spatial domain by

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

2-D Convolution (Spatial)

$$A = \begin{bmatrix} 17 & 24 & 1 & 8 & 15 \\ 23 & 5 & 7 & 14 & 16 \\ 4 & 6 & 13 & 20 & 22 \\ 10 & 12 & 19 & 21 & 3 \\ 11 & 18 & 25 & 2 & 9 \end{bmatrix}$$

$$H = \begin{bmatrix} 8 & 1 & 6 \\ 3 & 5 & 7 \\ 4 & 9 & 2 \end{bmatrix}$$

To compute the (2,4) output pixel using three steps:

2-D Convolution (Spatial)

Rotate the convolution kernel 180 degrees about its center element.

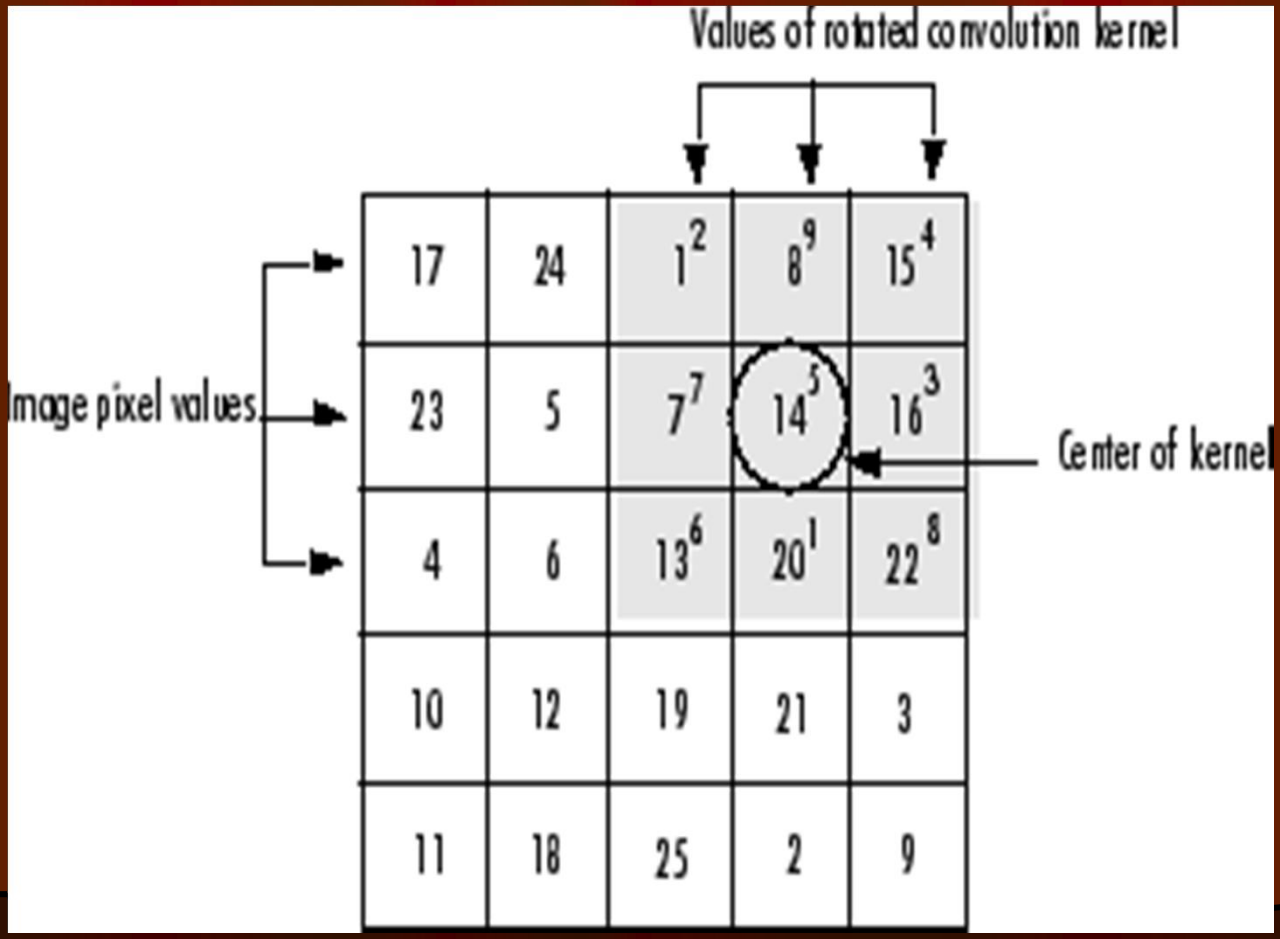
```
H = [ 8   1   6
      3   5   7
      4   9   2]
```

```
>> p = rot90(H)

     6     7     2
     1     5     9
     8     3     4
```

```
>> q = rot90(p)

     2     9     4
     7     5     3
     6     1     8
```

2-D Convolution (Spatial)

Slide the center element of the convolution kernel so that it lies on top of the (2,4) element of A.



2-D Convolution (Spatial)

Multiply each weight in the rotated convolution kernel by the pixel of A underneath.

Sum the individual products from step 3.

$$1 \cdot 2 + 8 \cdot 9 + 15 \cdot 4 + 7 \cdot 7 + 14 \cdot 5 + 16 \cdot 3 + 13 \cdot 6 + 20 \cdot 1 + 22 \cdot 8 = 575$$

Noise Sources

- The principal sources of noise in digital images arise during **image acquisition and/or transmission**

- Image acquisition

  e.g., light levels, sensor temperature, etc.

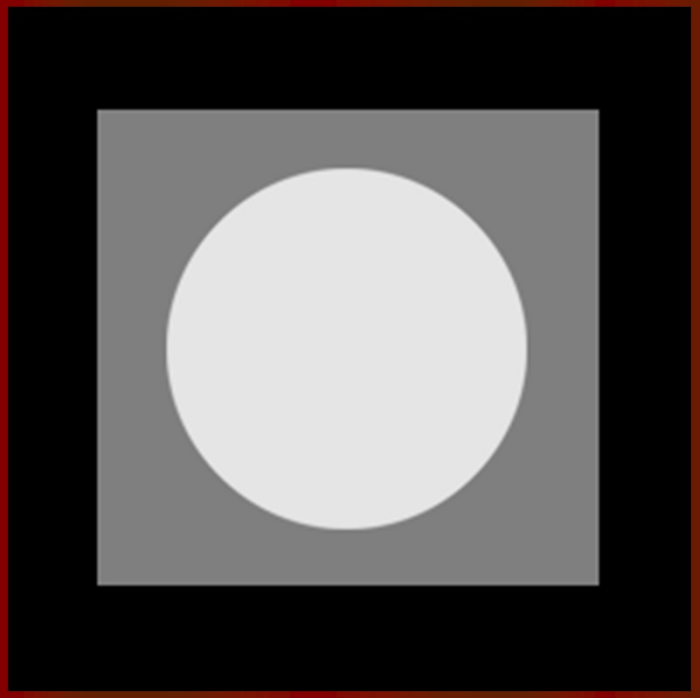
- Transmission

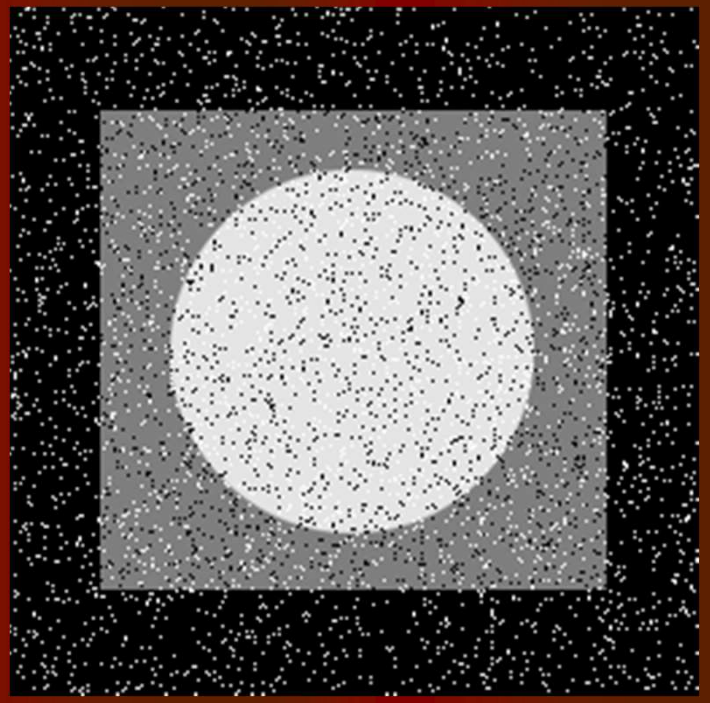
  e.g., lightning or other atmospheric disturbance in wireless network

Noise probability density functions

- Noises are taken as random variables
- Random variables
  - Probability density function (PDF)

Salt & Pepper Noise





Noise Probability Distribution (Salt & Pepper Noise)

The PDF of (bipolar) impulse noise is given by

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

if $b > a$, gray-level b will appear as a light dot, while level a will appear like a dark dot.

If either P_a or P_b is zero, the impulse noise is called *unipolar*

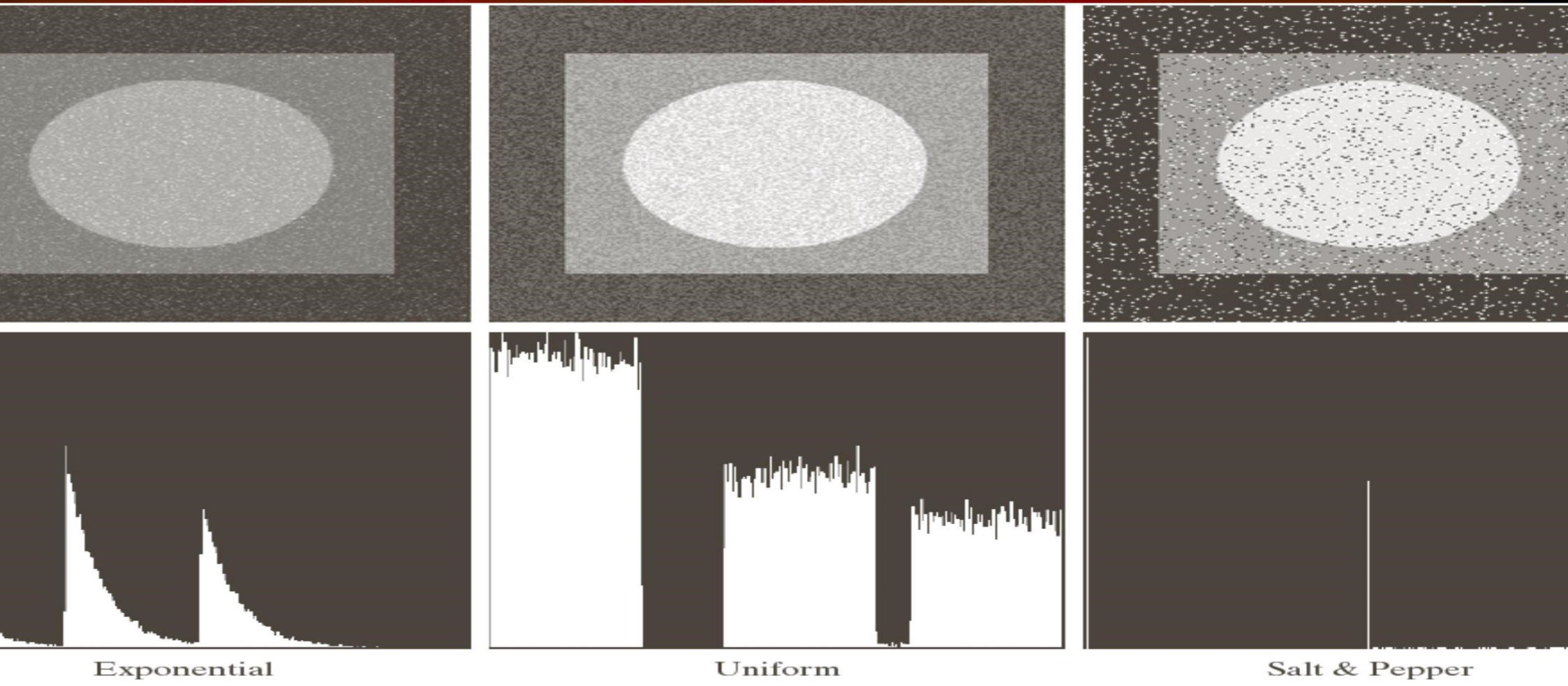
Salt & Pepper Noise

- a = imread('C:\lake2.bmp');
- a = double(a);
- a = mat2gray(a);
- imhist(a);
- b = imnoise(a,'Salt & Pepper');
- figure,imshow (b);
- figure, imhist(b)

Implementation: Salt & Pepper

```
function i= saltpepper(img1,a,b)
 [m,n]=size(img1);
 img1=mat2gray(double(img1));
 r= rand(m,n);
 x=find(r <=a);
 img1(x)=0;
 x=find(r >a & r <=(a+b));
 img1(x)=255;
 figure,imhist(img1);
 figure,imshow(img1);
imwrite(img1,'C:\board_salt.tif');
return i;
```

Noise Patterns



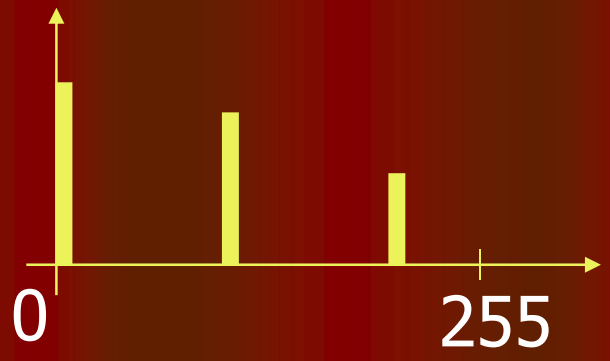
E 5.4 *(Continued)* Images and histograms resulting from adding exponential, uniform, and sa
r noise to the image in Fig. 5.3.

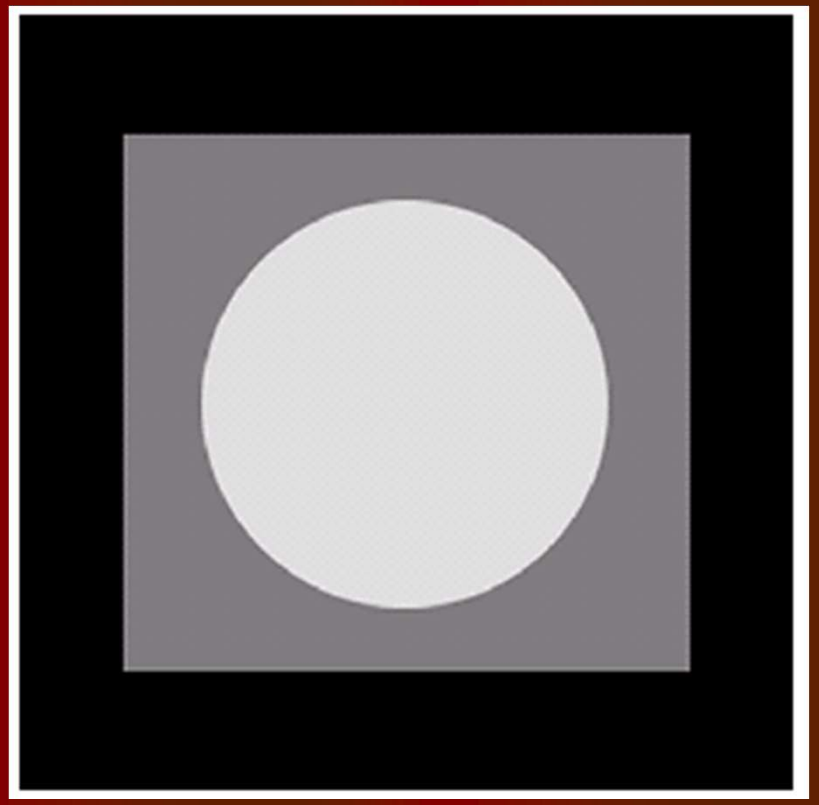
Test for noise behavior

- Uniform noise

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

Its histogram:





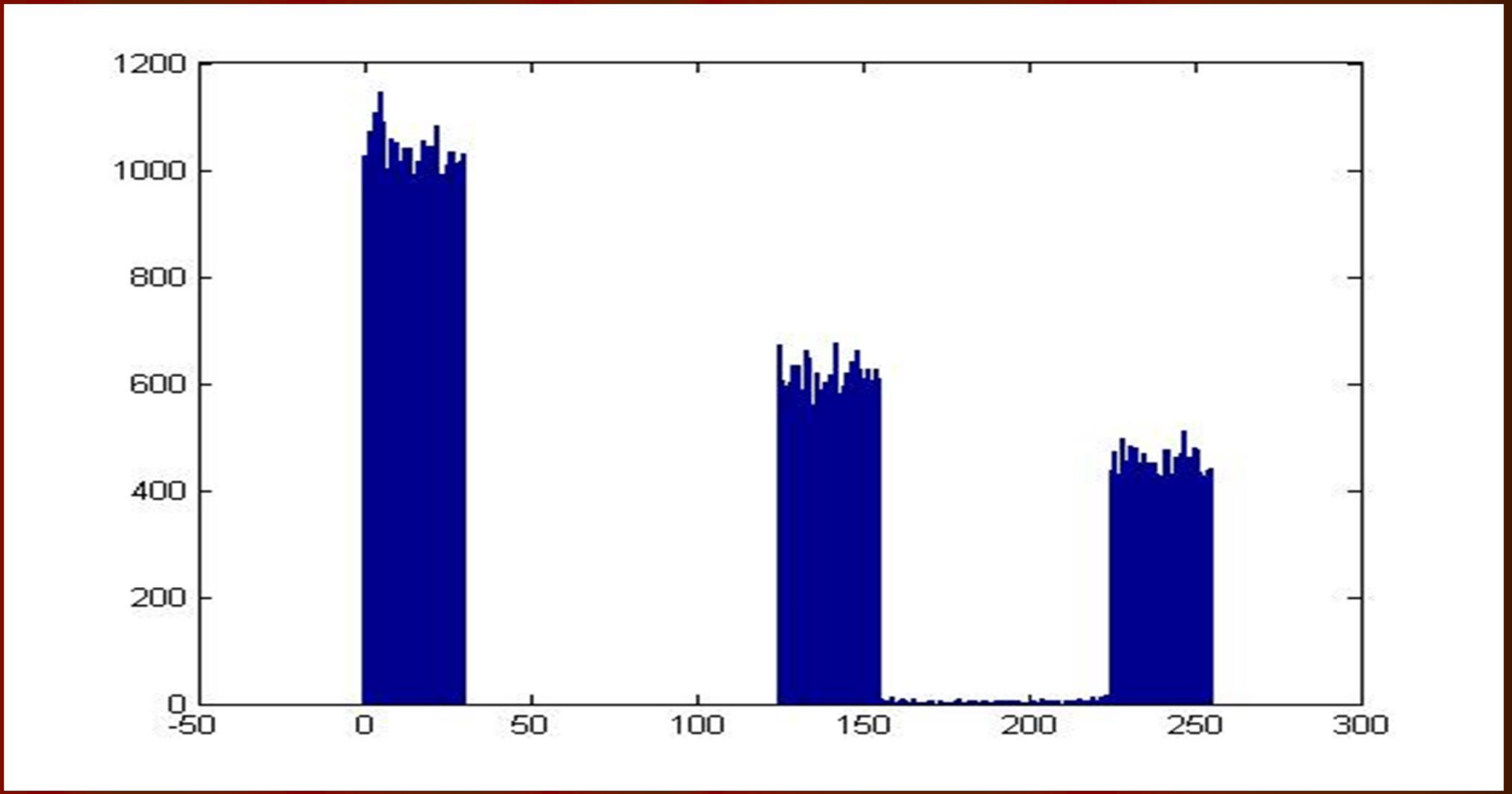
Test for noise behavior

```
a1=(imread('Fig0503 (original_pattern).tif'));
[m,n]=size(a1);
z=uint8(randi([10,40],m,n));

noizy_a=double(a1)+ double(z) ;
noisy=imhist(mat2gray(noizy_a));
original=imhist((a1));
```

Test for noise behavior

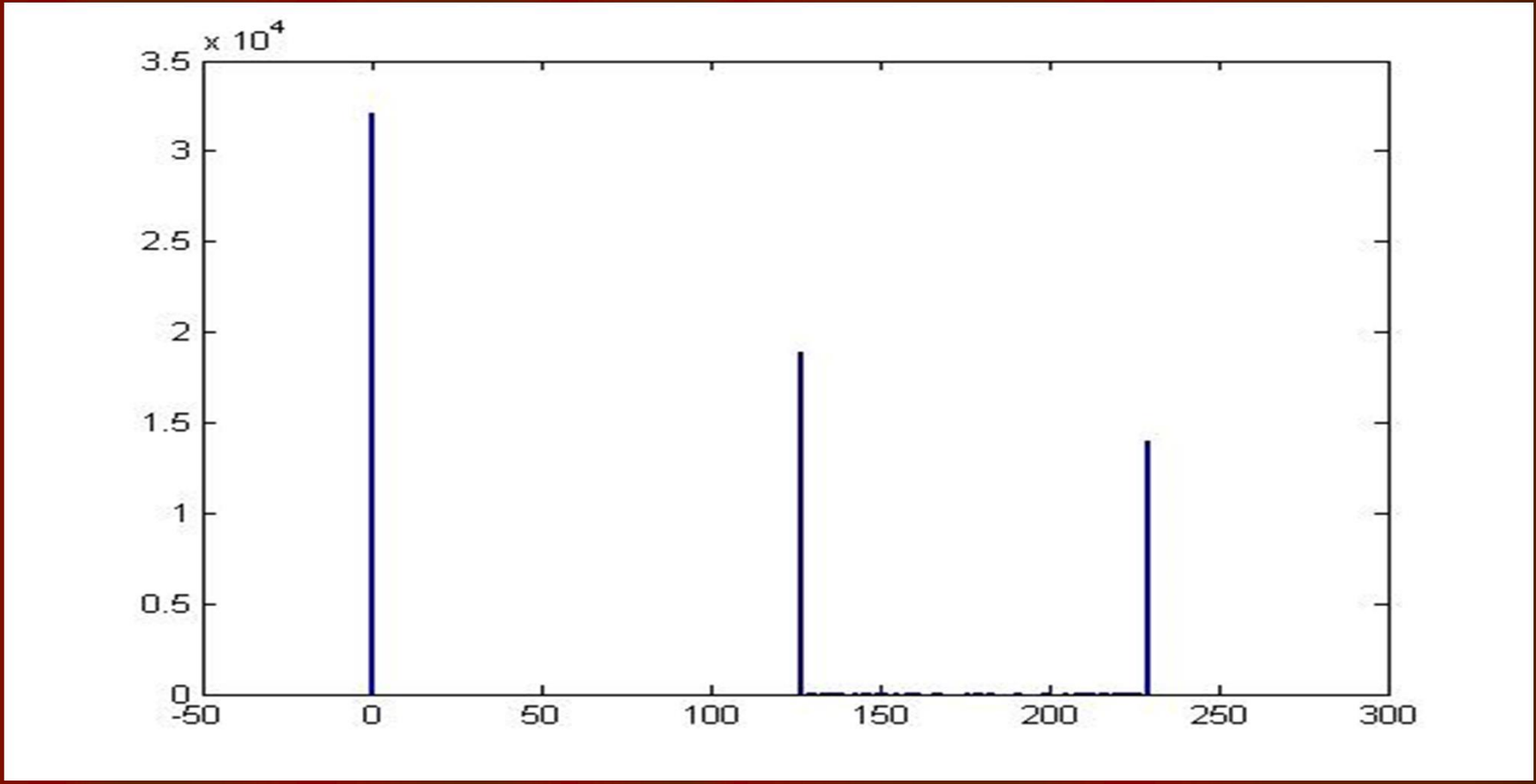
```
figure(1), bar(0:255,noisy)
figure(2) , bar(0:255,original)
figure (3)
subplot(211)
imshow(a1)
title('original image')
subplot(212)
imshow(mat2gray(noizy_a))
title('noisy image-uniform noise')
```

Test for noise behavior



Noisy Image

Test for noise behavior



Original Image

Noise Probability Distribution (Uniform Noise)

The PDF of uniform noise is given by

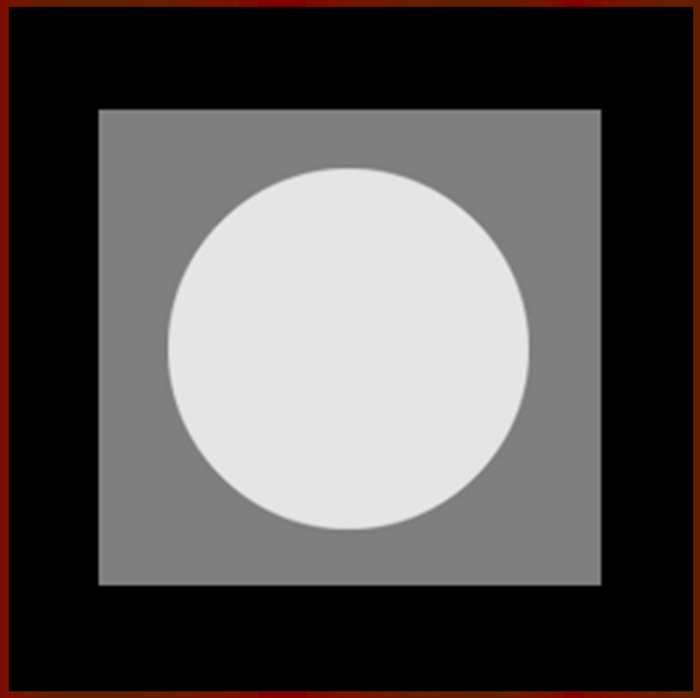
$$p(z) = \begin{cases} \dfrac{1}{b-a} & \text{for a} \le z \le b \\ 0 & \text{otherwise} \end{cases}$$

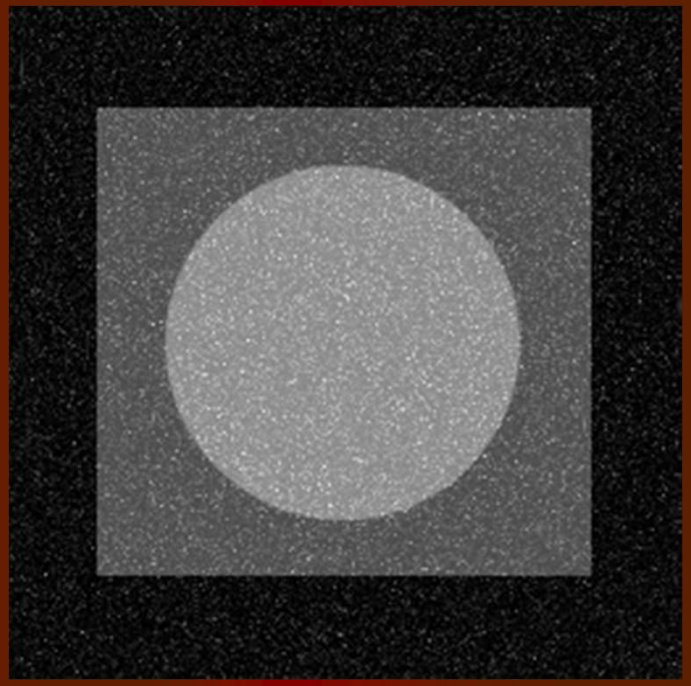
The mean and variance of this density are given by

$$\bar{z} = (a+b)/2$$

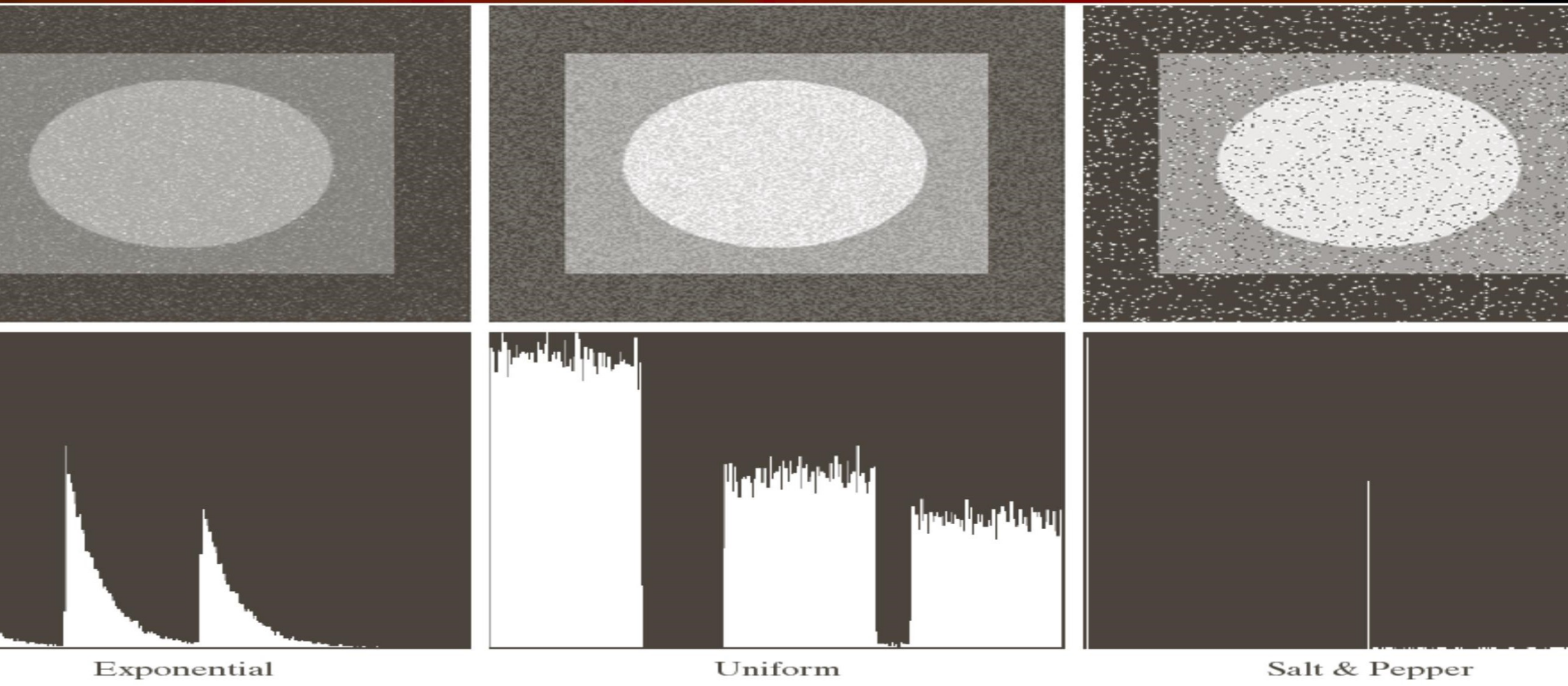
$$\sigma^2 = (b-a)^2/12$$

Exponential Noise





Noise Patterns



E 5.4 *(Continued)* Images and histograms resulting from adding exponential, uniform, and sa
r noise to the image in Fig. 5.3.

Noise Probability Distribution (Exponential Noise)

The PDF of exponential noise is given by

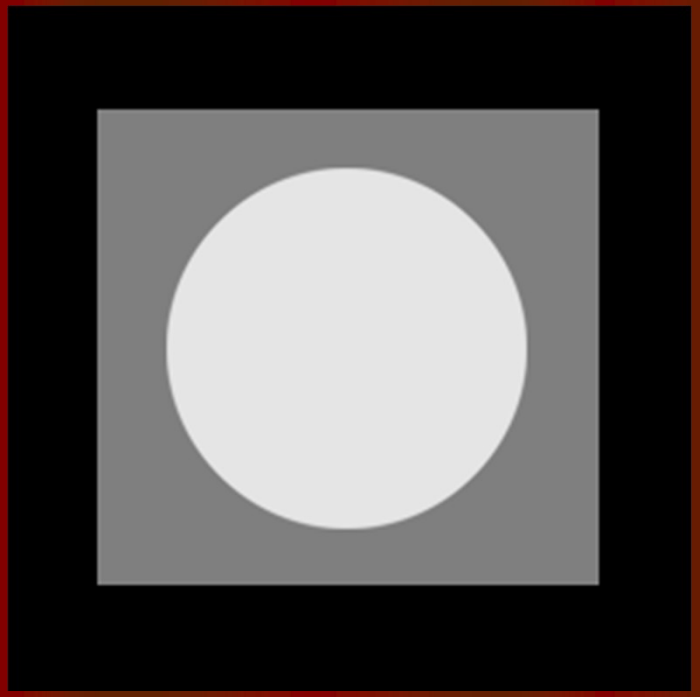
$$p(z) = \begin{cases} ae^{-az} & \text{for } z \geq 0 \\ 0 & \text{for } z < a \end{cases}$$

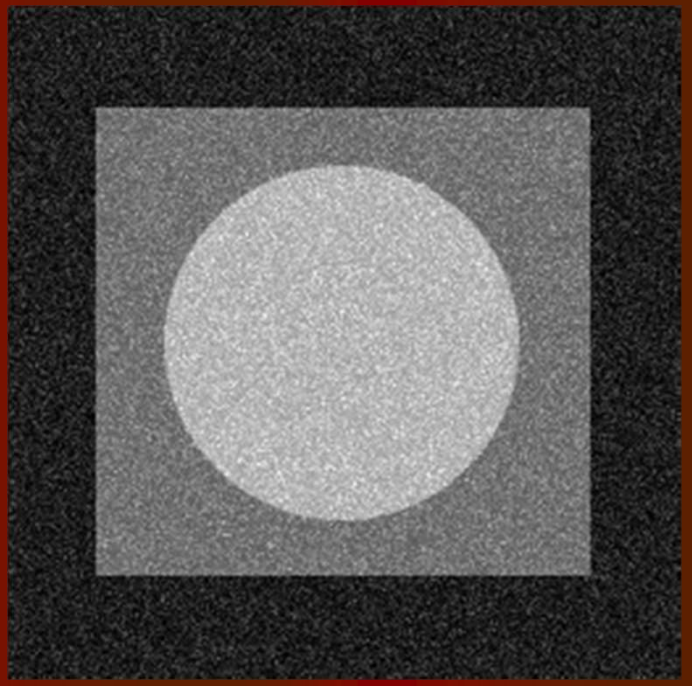
The mean and variance of this density are given by

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

$$\sigma^2 = 1/a^2$$

Rayleigh Noise





Noise Probability Distribution (Rayleigh Noise)

The PDF of Rayleigh noise is given by

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

The mean and variance of this density are given by

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

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

Gaussian Noise (Normal Noise)

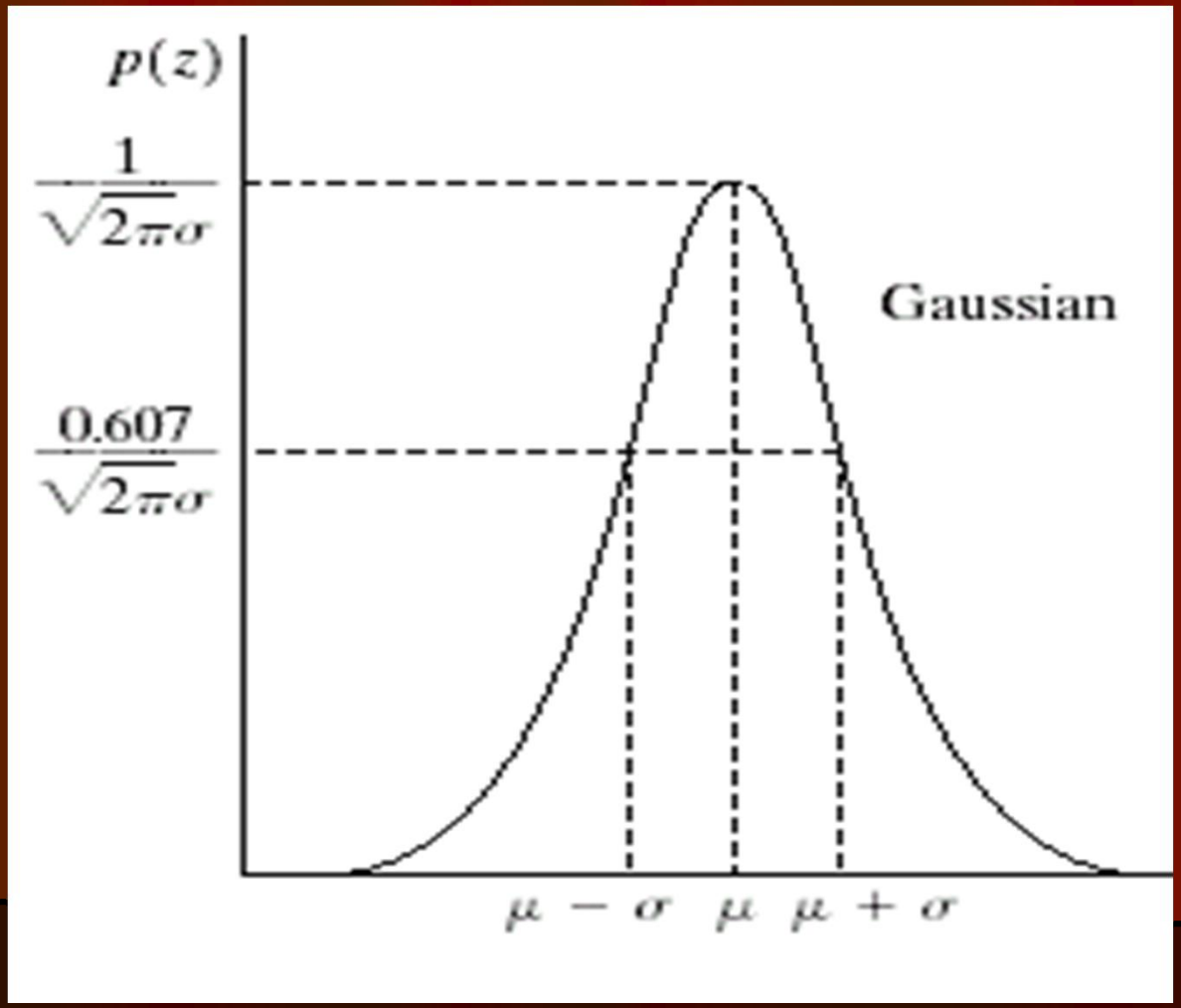
- The pdf of a Gaussian random variable z is given by:

$$p(z) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(z-\mu)^2}{2\sigma^2}\right)$$

where z represents (noise) gray value, μ is the mean, and σ is its standard deviation. The squared standard deviation σ^2 is usually referred to as variance.

Gaussian Noise

- For a Gaussian pdf, approximately 70% of its values will be in the range $[(\mu-\sigma),(\mu+\sigma)]$, and 95% of its values will be in the range $[(\mu-2\sigma),(\mu+2\sigma)]$



Gaussian Noise

```
clc;
clear;
img2=imread('D:\Image Processing\IP_sttp\lena_gray_256.tif');
imshow(img2);
img2=double(img2);
img2 = mat2gray(img2);
r = imnoise(img2, ‘Gaussian’, 20/256, 0.0001);
imshow(r);
imhist(r);
```

Gamma Noise

- The pdf of Erlang noise is given by:

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

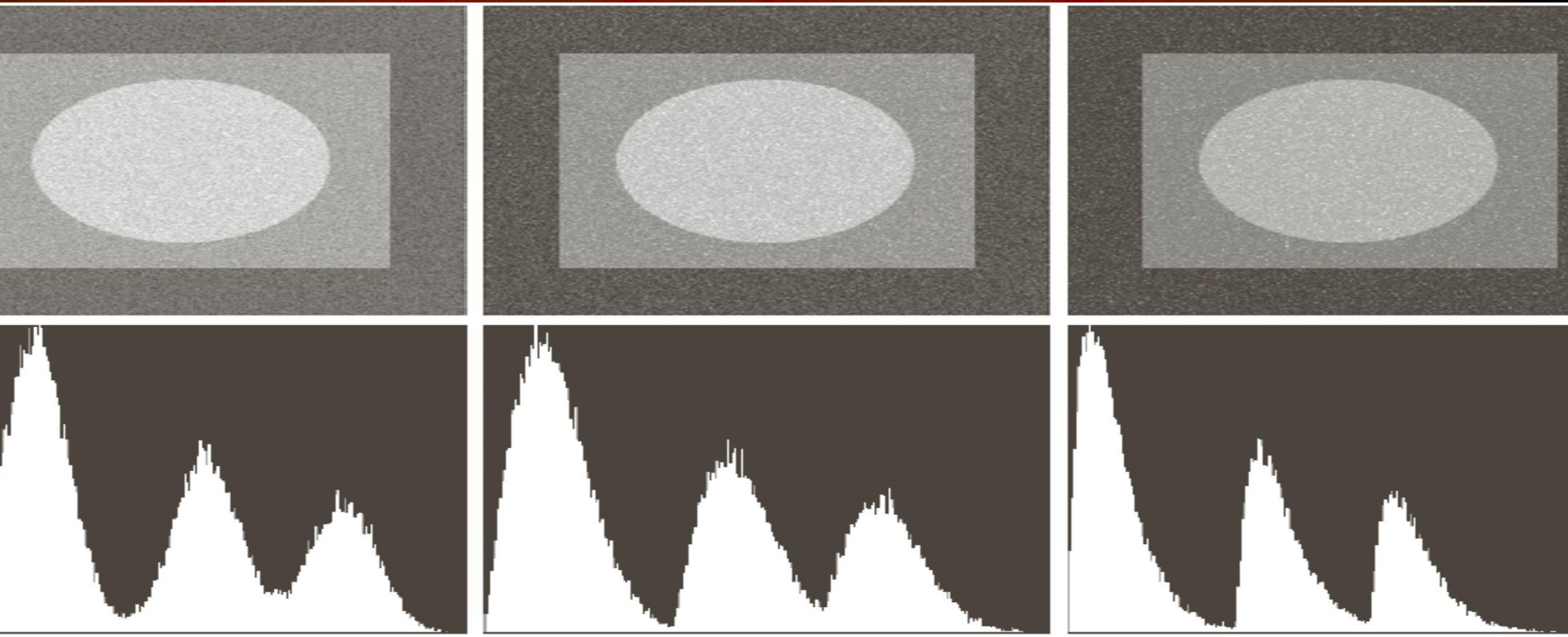
where, $a > 0$, b is an integer and "!" represents factorial.

- The mean and variance are given by:

$$\mu = b/a$$

$$\sigma^2 = b/a^2$$

Noise Patterns



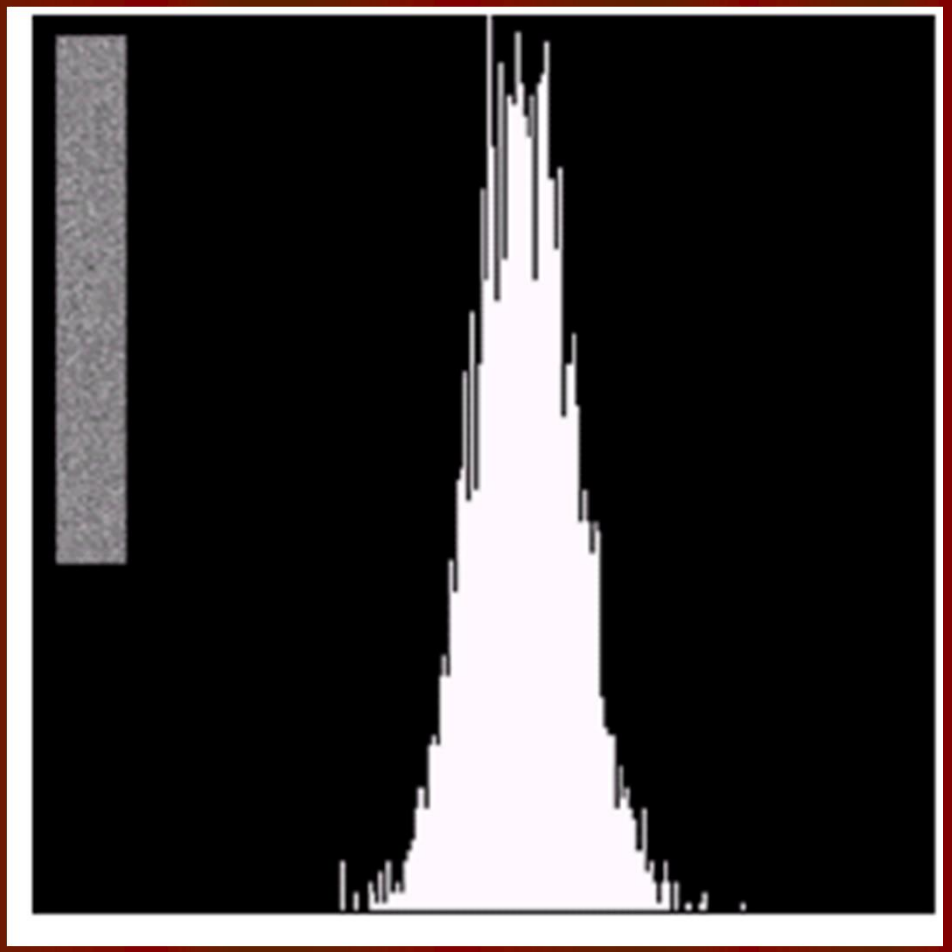
Gaussian Rayleigh Gamma

E 5.4 Images and histograms resulting from adding Gaussian, Rayleigh, and gamma noise to the in
5.3.

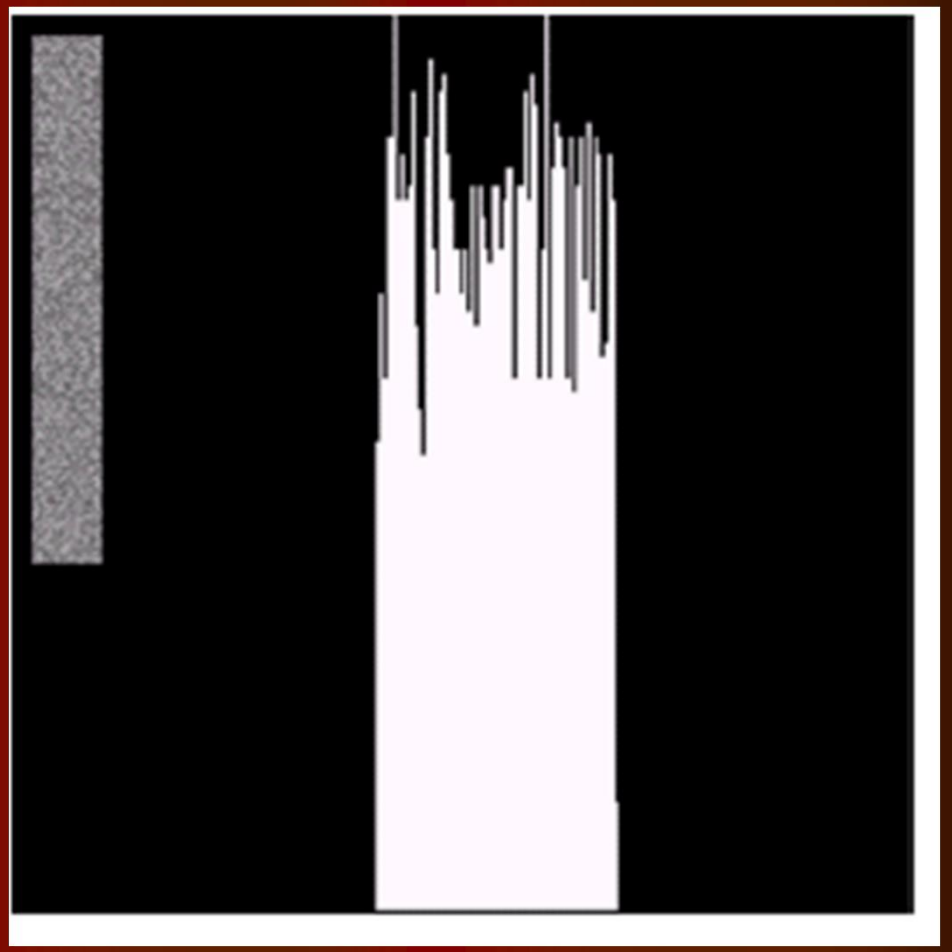
Estimation of noise parameters

- Random noise with unknown PDFs
  - Case 1: imaging system is available
    - Capture images of "flat" environment
  - Case 2: noisy images available
    - Take a strip from constant area
    - Draw the histogram and observe it
    - Measure the mean and variance

Observe the histogram



Gaussian



uniform

Estimation of Noise Parameters

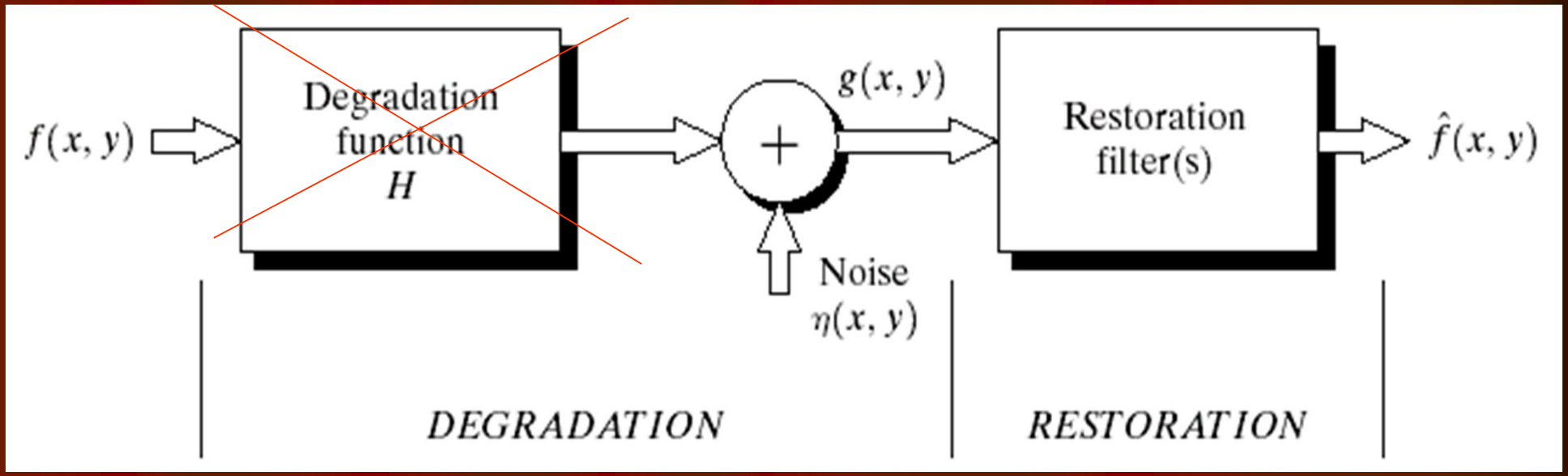
onsider a subimage denoted by S, and let $p_s(z_i)$, $i = 0, 1, \ldots, L-$
enote the probability estimates of the intensities of the pixels in
he mean and variance of the pixels in S:

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

nd

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

Additive noise only



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

$$G(u,v)=F(u,v)+N(u,v)$$

Spatial filters for de-noising additive noise

- Skills similar to image enhancement
- Mean filters
- Order-statistics filters
- Adaptive filters

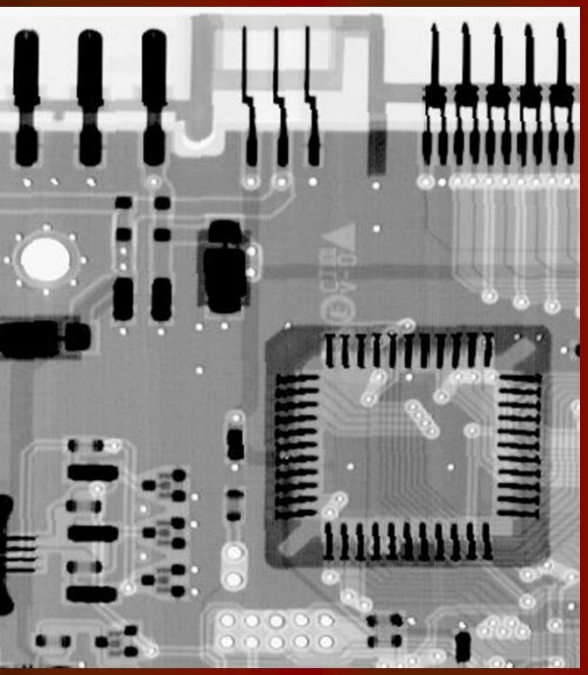
Spatial Filtering: Mean Filter

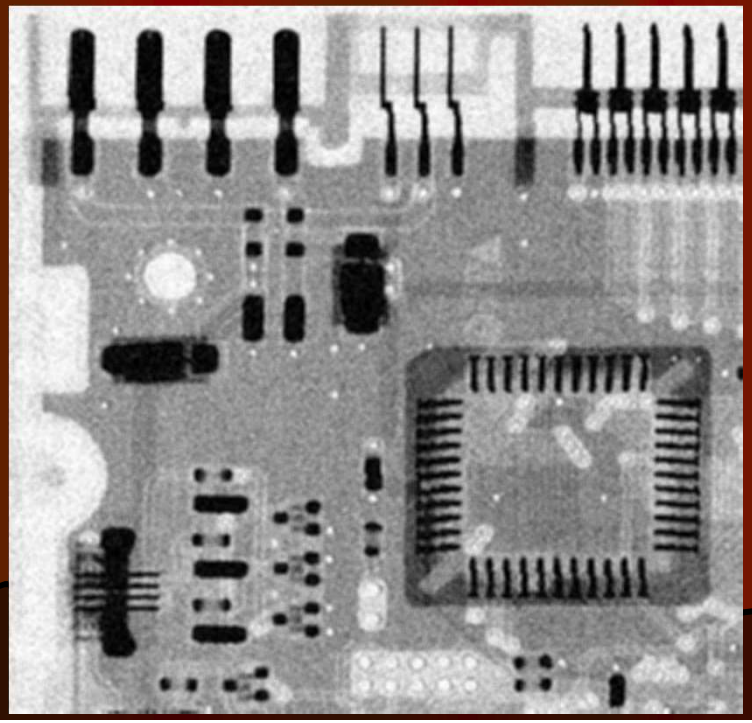
Let S_{xy} represent the set of coordinates in a rectangle subimage window of size $m \times n$, centered at (x, y).

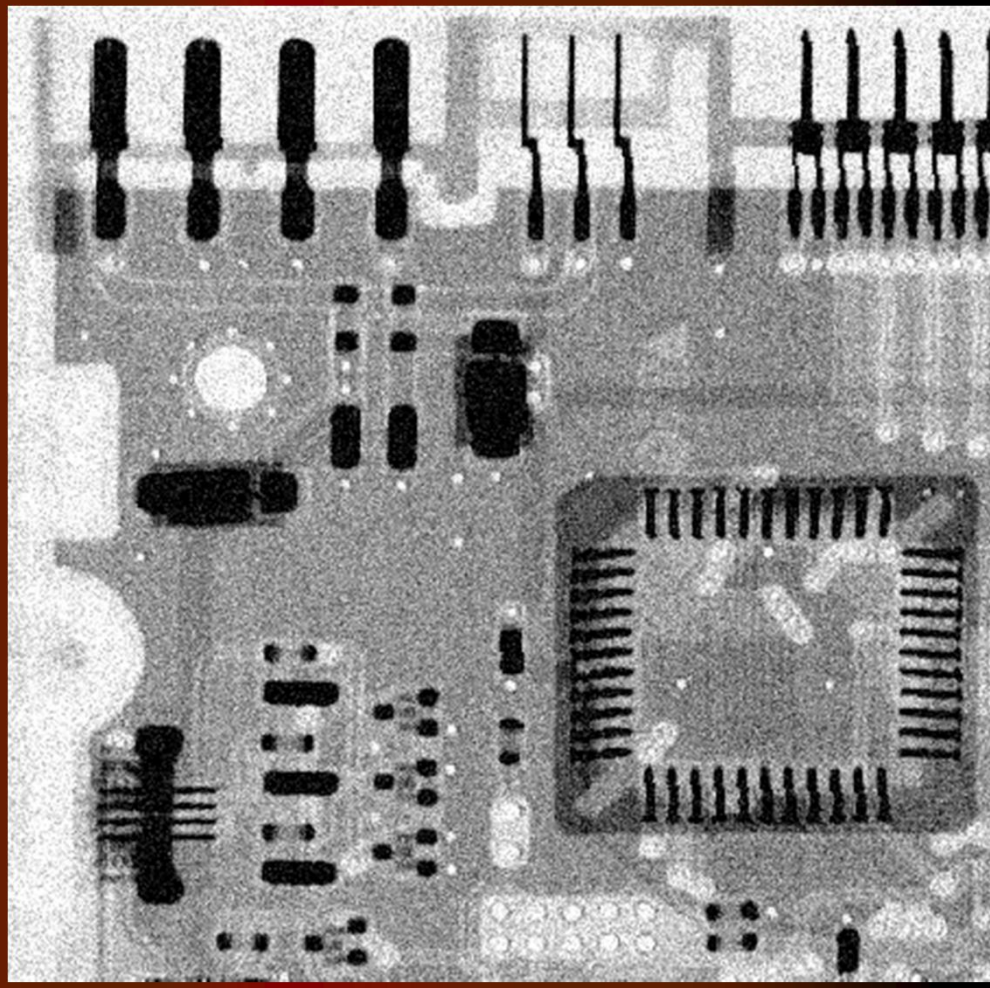
Arithmetic mean filter

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

Spatial Filtering: Mean Filter







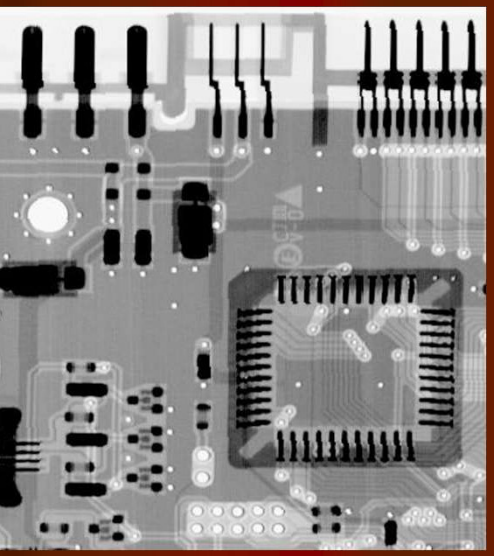
Spatial Filtering: Harmonic Mean Filter

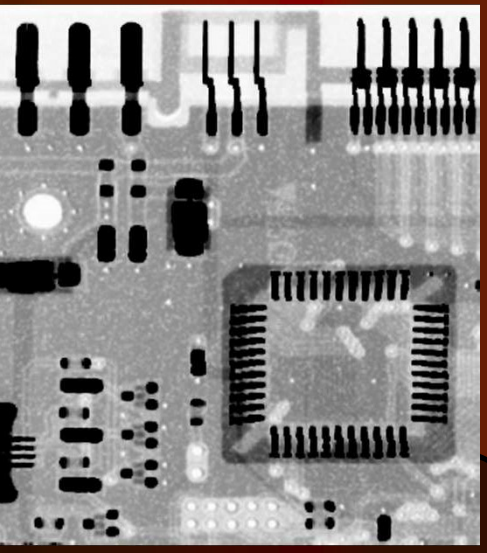
Harmonic mean filter

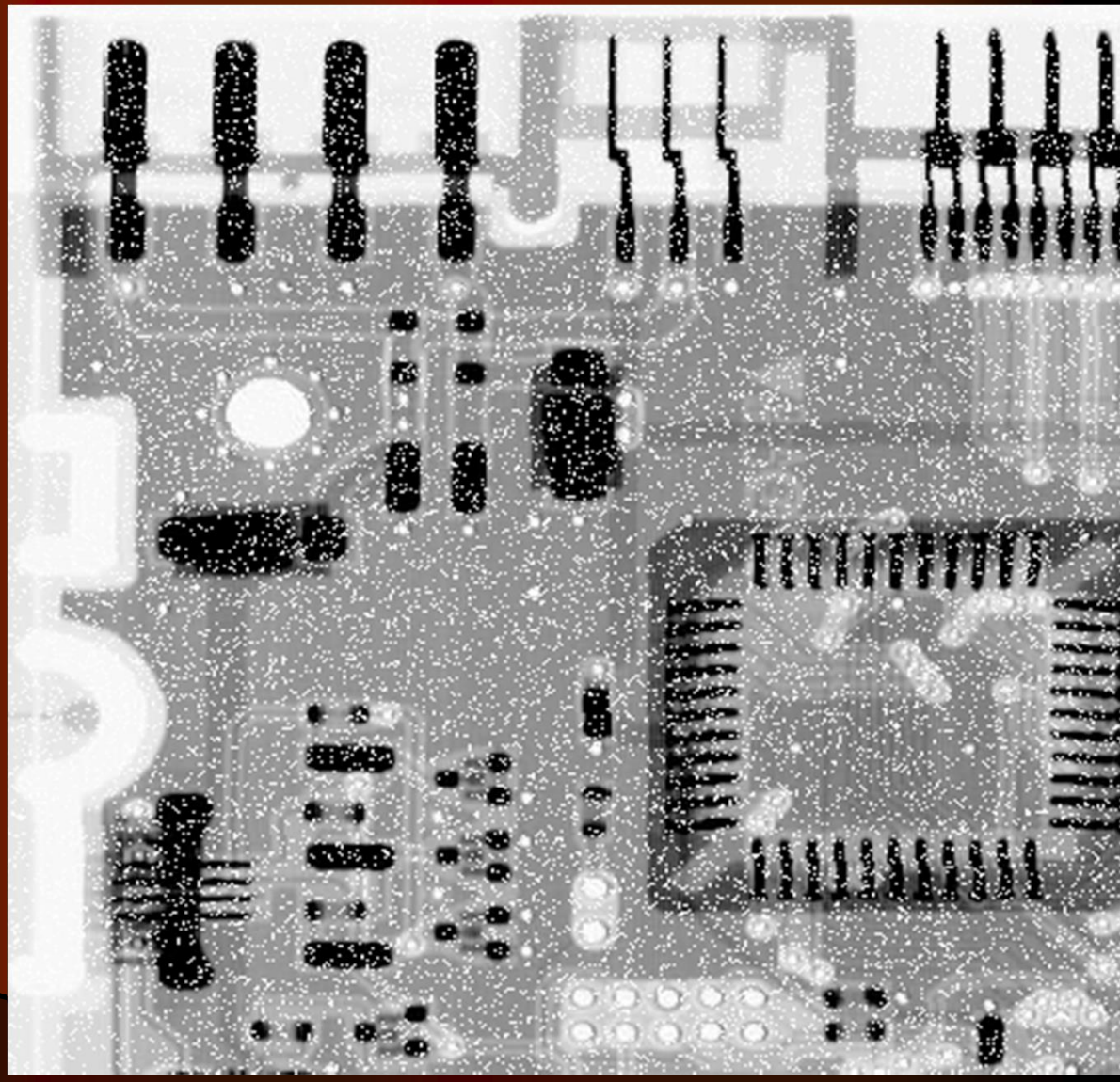
$$\hat{f}(x,y) = \frac{mn}{\sum_{(s,t)\in S_{xy}} \frac{1}{g(s,t)}}$$

It works well for salt noise, but fails for pepper noise.
It does well also with other types of noise like Gaussian noise.

Spatial Filtering: Harmonic Mean Filter







Spatial Filtering: Contra Harmonic Mean Filter

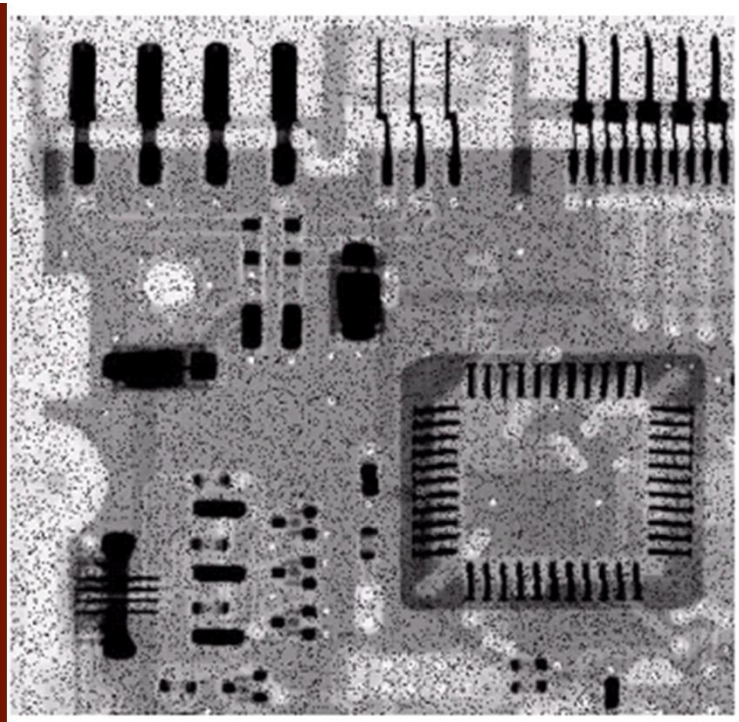
Contraharmonic mean filter

$$\hat{f}(x,y) = \frac{\sum_{(s,t)\in S_{xy}} g(s,t)^{Q+1}}{\sum_{(s,t)\in S_{xy}} g(s,t)^{Q}}$$

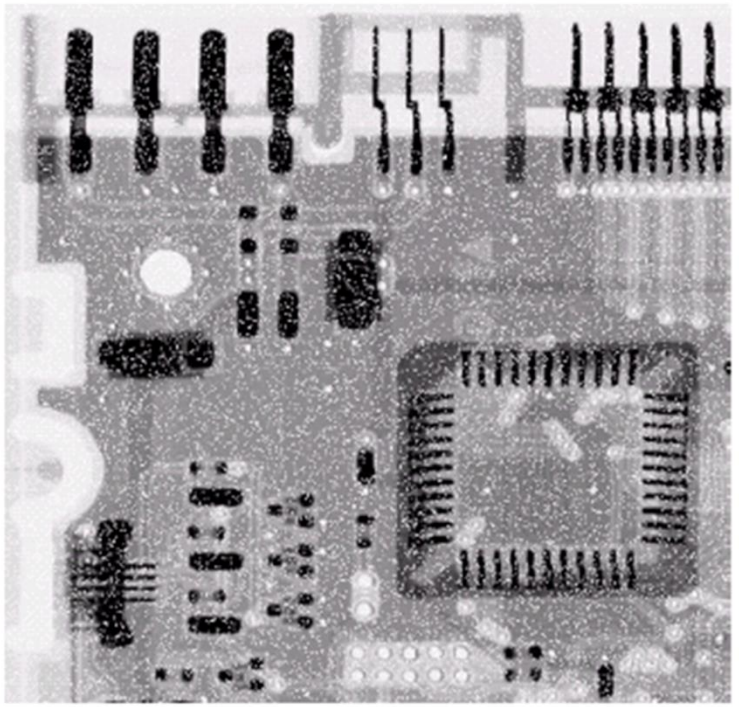
Q is the order of the filter.

It is well suited for reducing the effects of salt-and-pepper noise. Q>0 for pepper noise and Q<0 for salt noise.

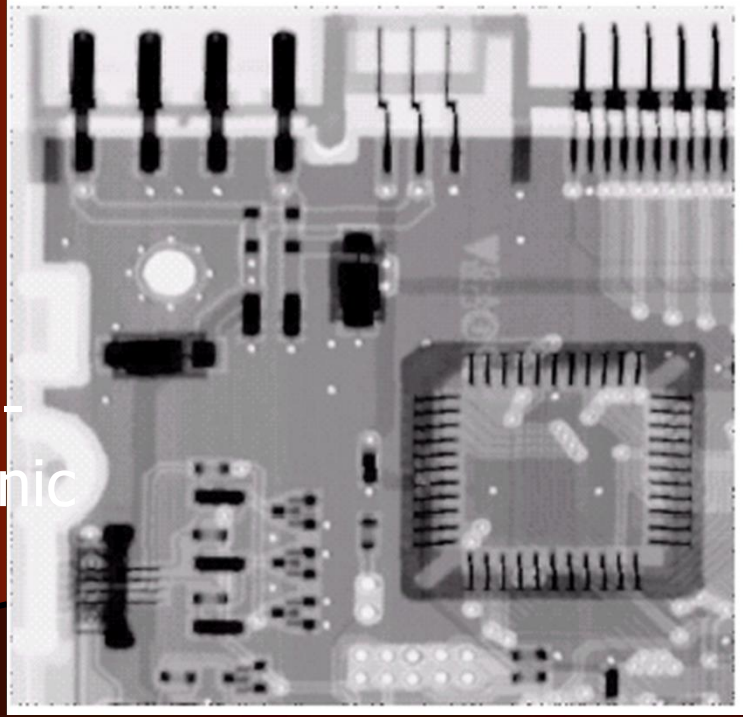
Pepper Noise



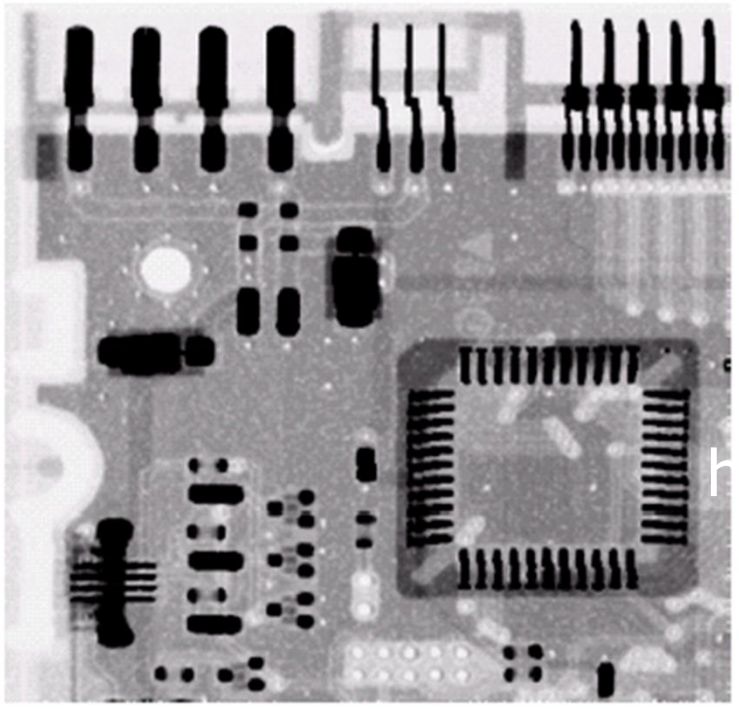
Salt Noise



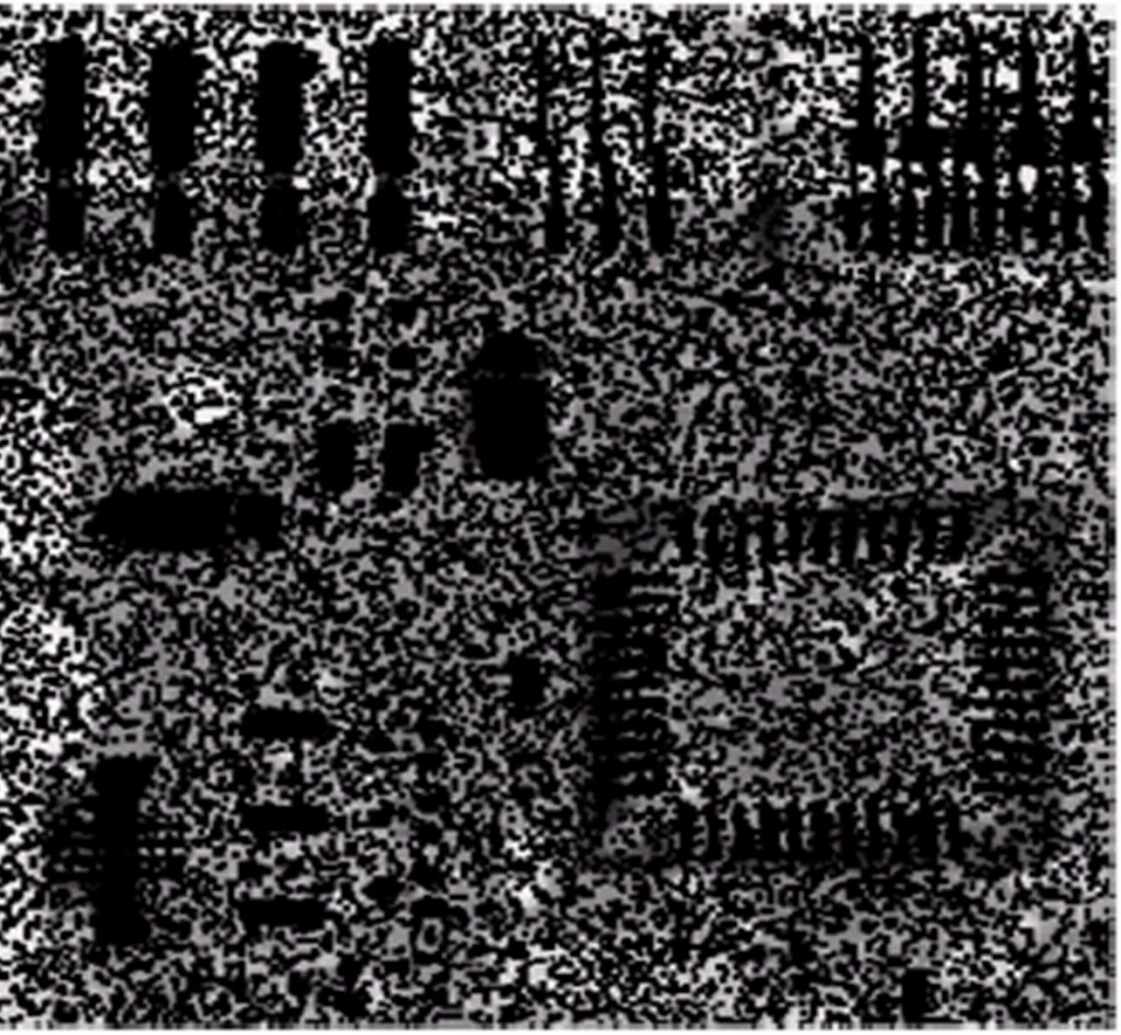
Contra-harmonic Q=1.5



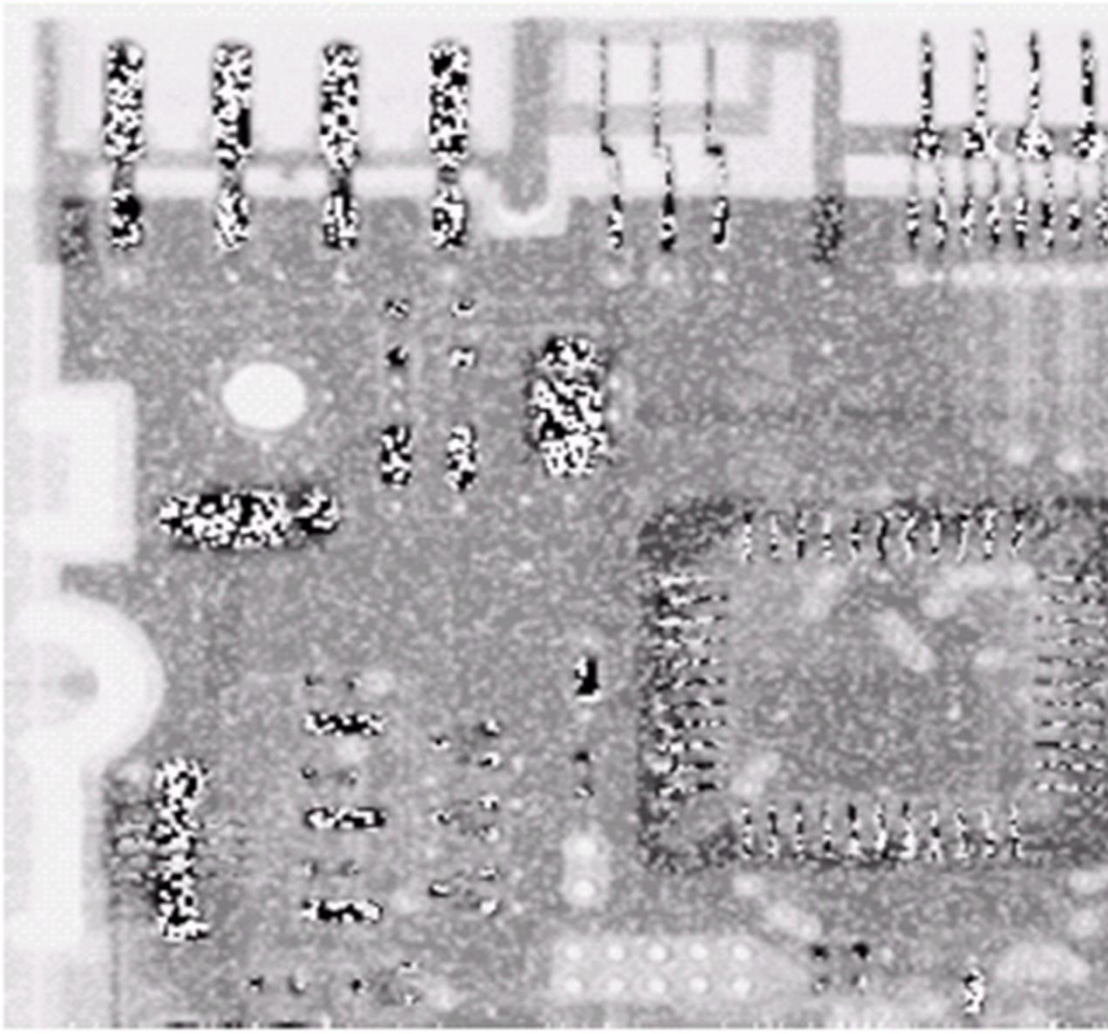
Contra-harmonic Q=-1.5



Wrong sign in contra-harmonic filtering



Q=-1.5



Q=1.5

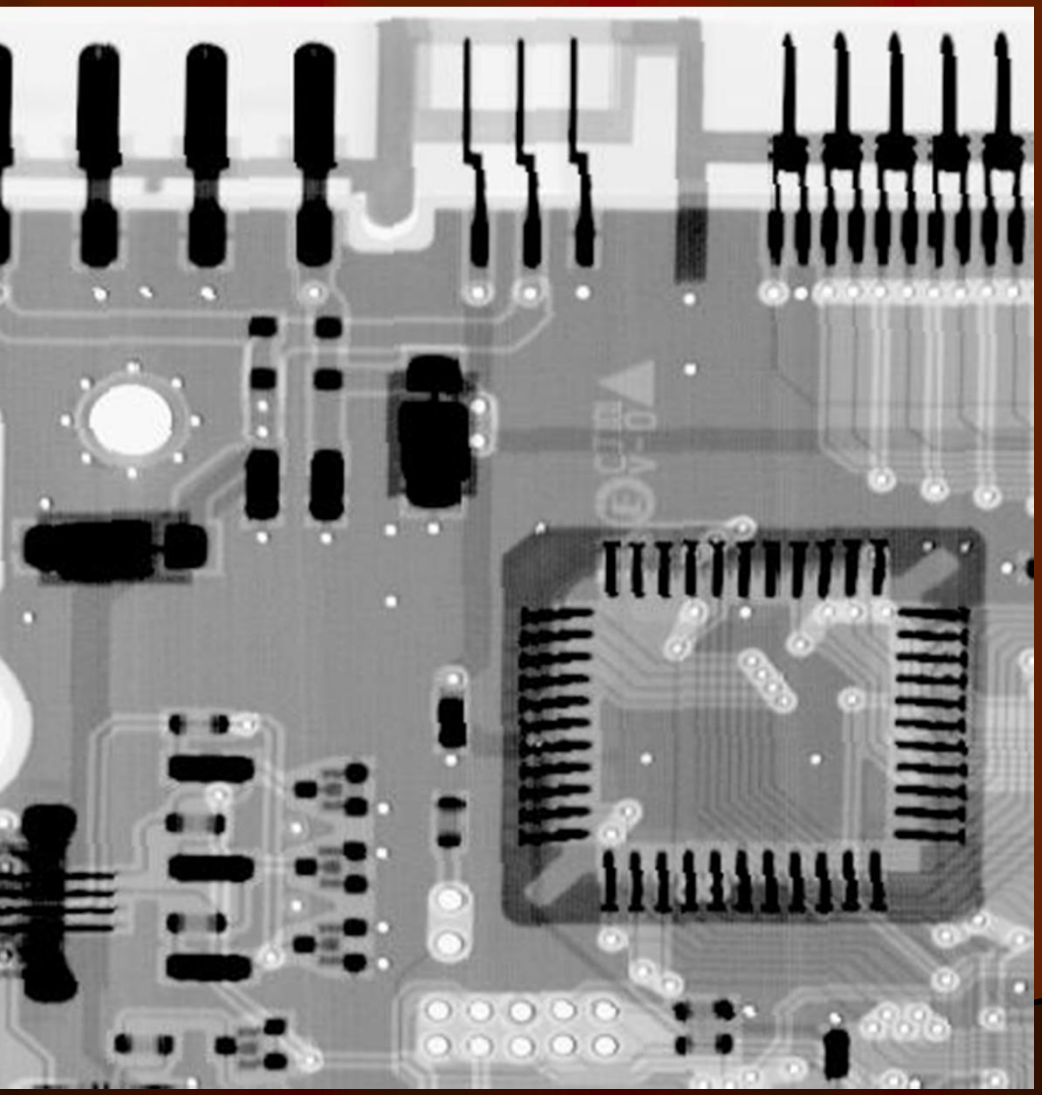
Spatial Filtering: Geometric Mean Filter

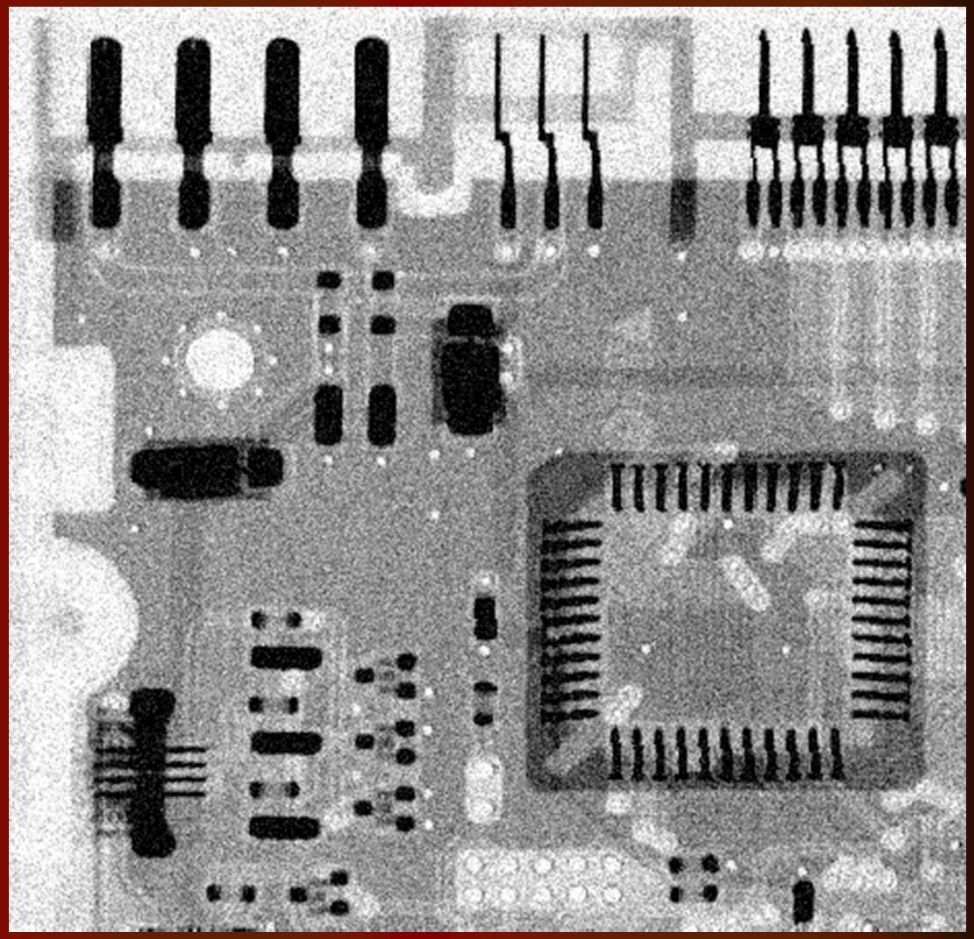
Geometric mean filter

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

Generally, a geometric mean filter achieves smoothing comparable to the arithmetic mean filter, but it tends to lose less image detail in the process

Spatial Filtering: Geometric Mean Filter





Order Statistics Filter

- Order statistics filter are spatial filters
- Response of filter is based on the ordering the values of the pixels contained in the image area encompassed by filter.
- Order- Statistics filter are:

  Median Filter

  Max Filter

  Min Filter

  Mid-Point Filter

  Alpha-Trimmed Mean Filter

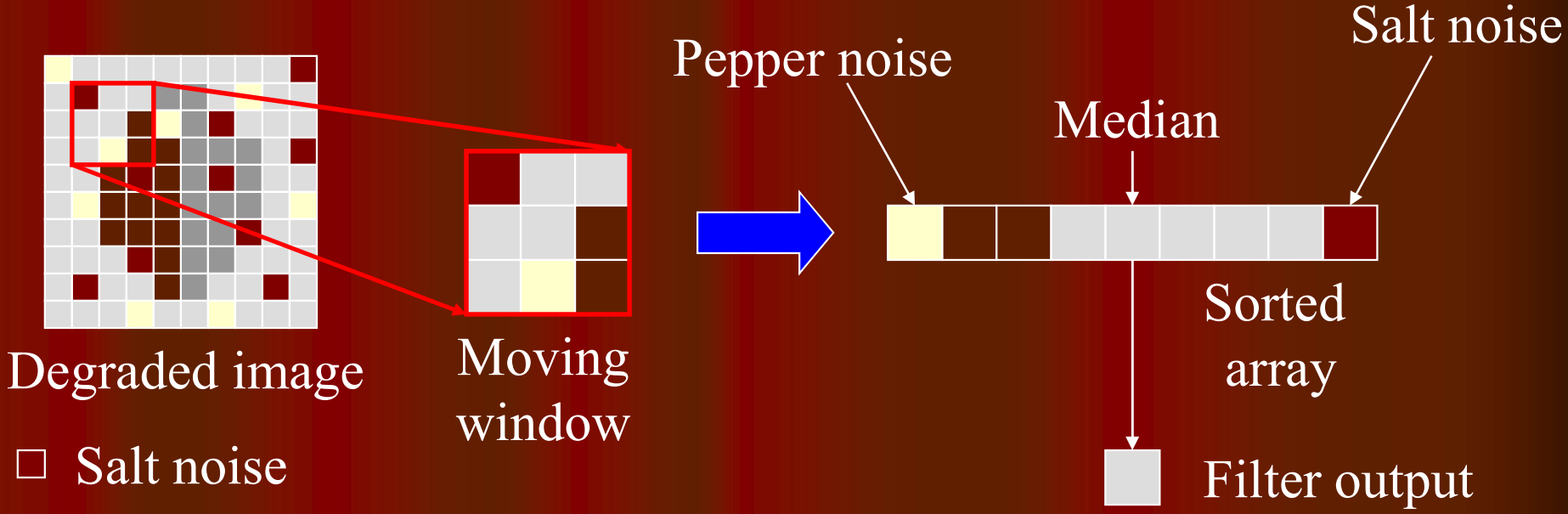
Order Statistics Filter Median Filter

Median filter

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

Order Statistics Filter Median Filter

A median filter is good for removing impulse, isolated noise

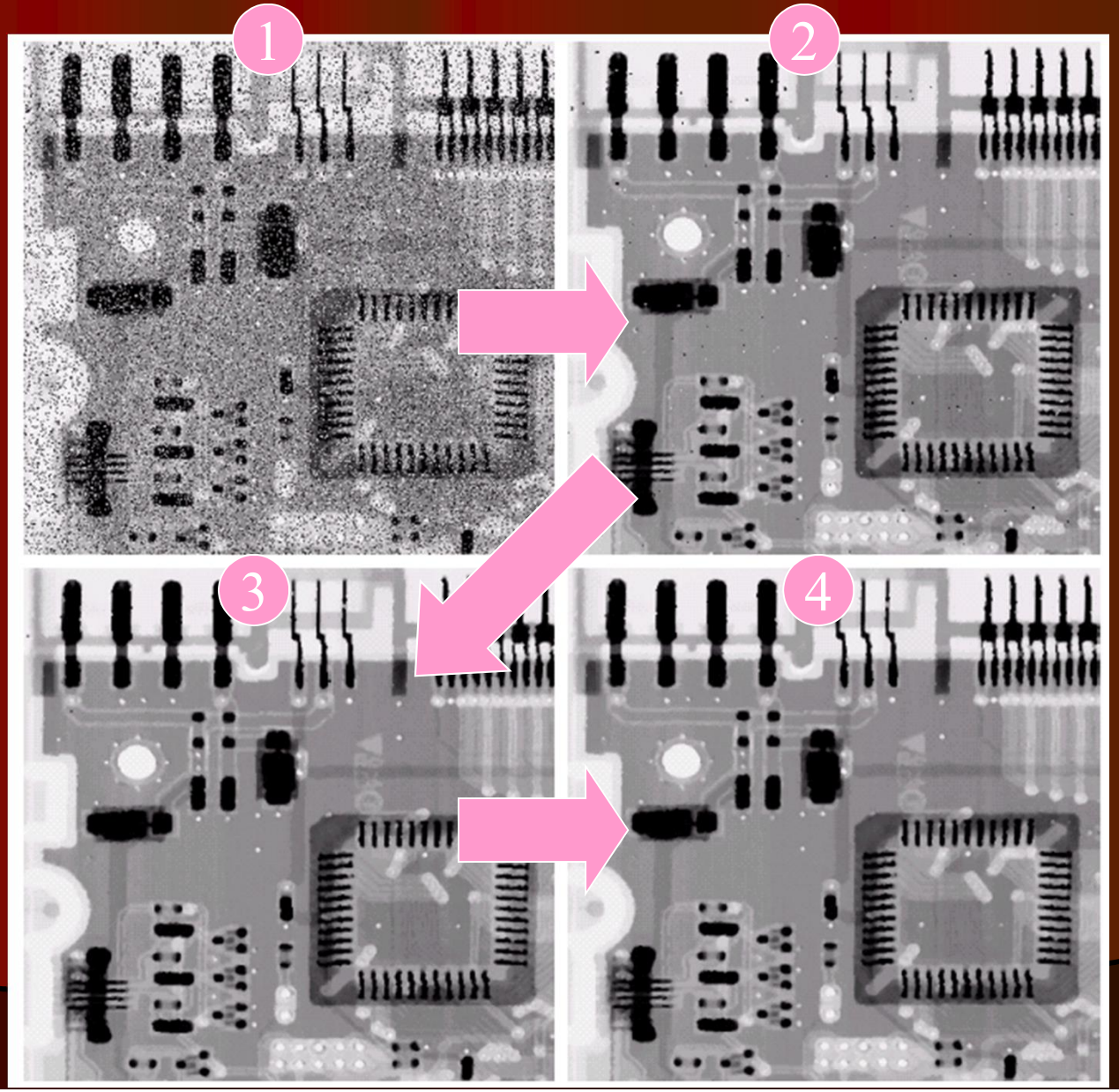


Normally, impulse noise has high magnitude and is isolated. When we sort pixels in the moving window, noise pixels are usually at the ends of the array.

Therefore, it's rare that the noise pixel will be a median value.

Order Statistics Filter Median Filter

Image corrupted by salt-and-pepper noise with $p_a = p_b = 0.1$



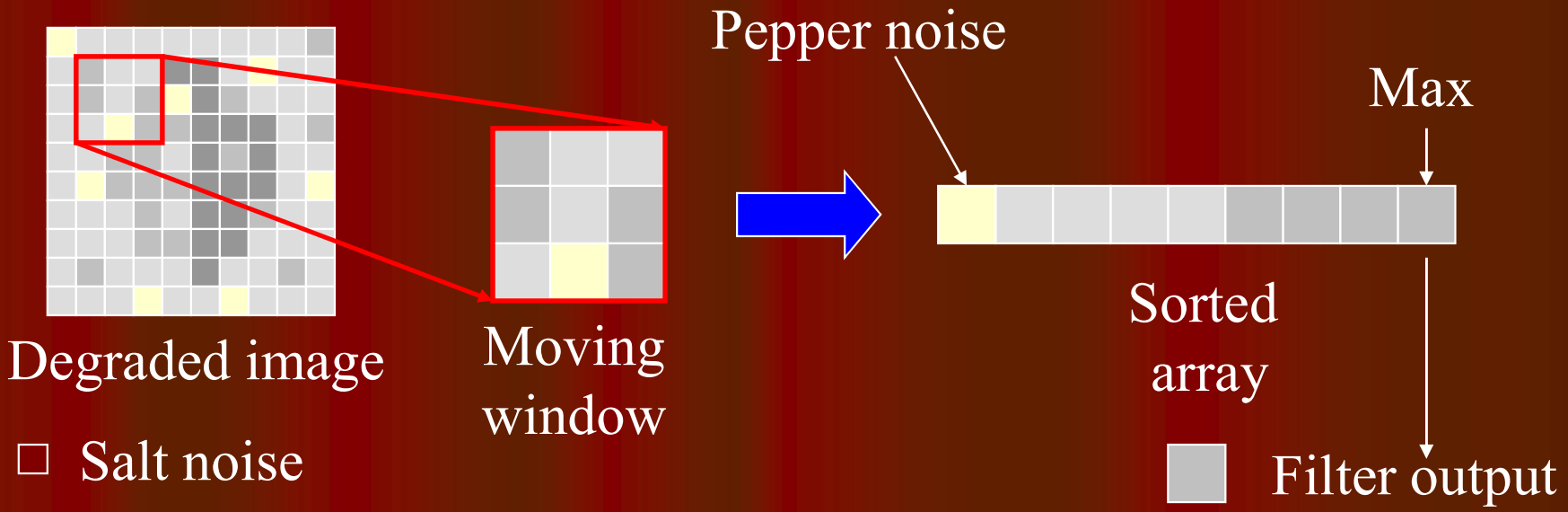
Order Statistics Filter Max Filter

Max filter

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

Order Statistics Filter Max Filter

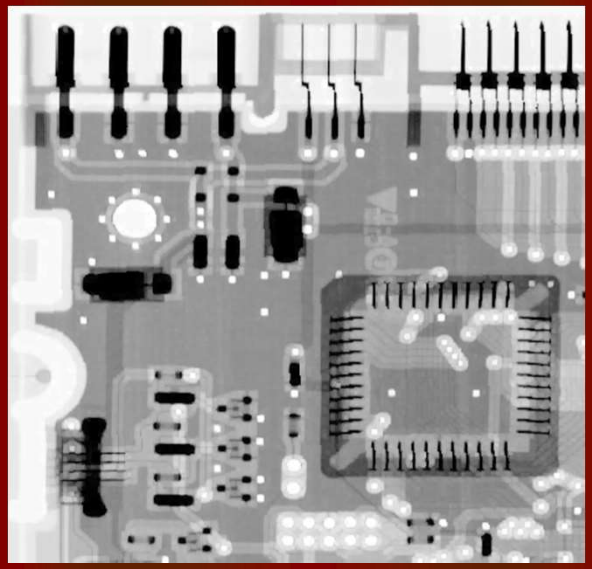
A max filter is good for removing impulse, isolated noise

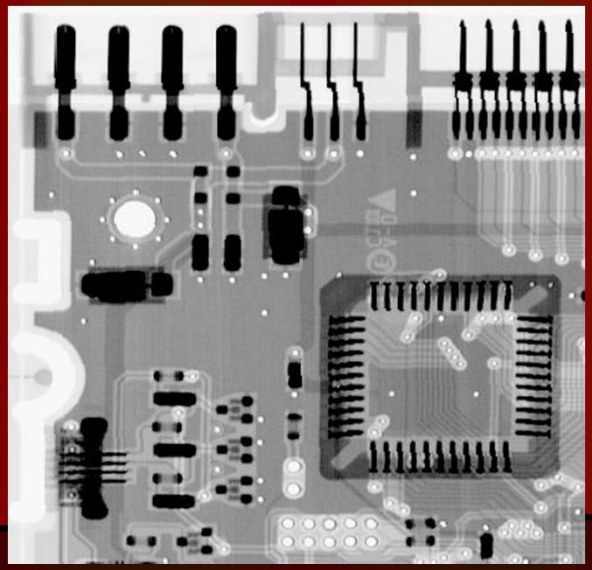


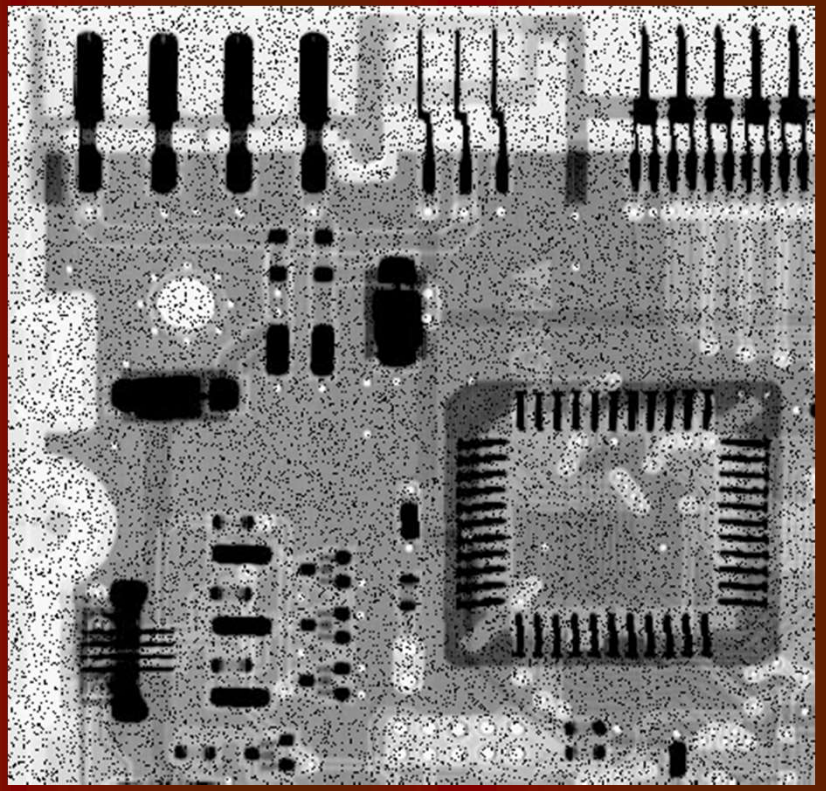
Normally, impulse noise has high magnitude and is isolated. When we sort pixels in the moving window, noise pixels are usually at the Left end of the array.

Therefore, it's rare that the noise pixel will be a median value.

Spatial Filtering: Max Filter







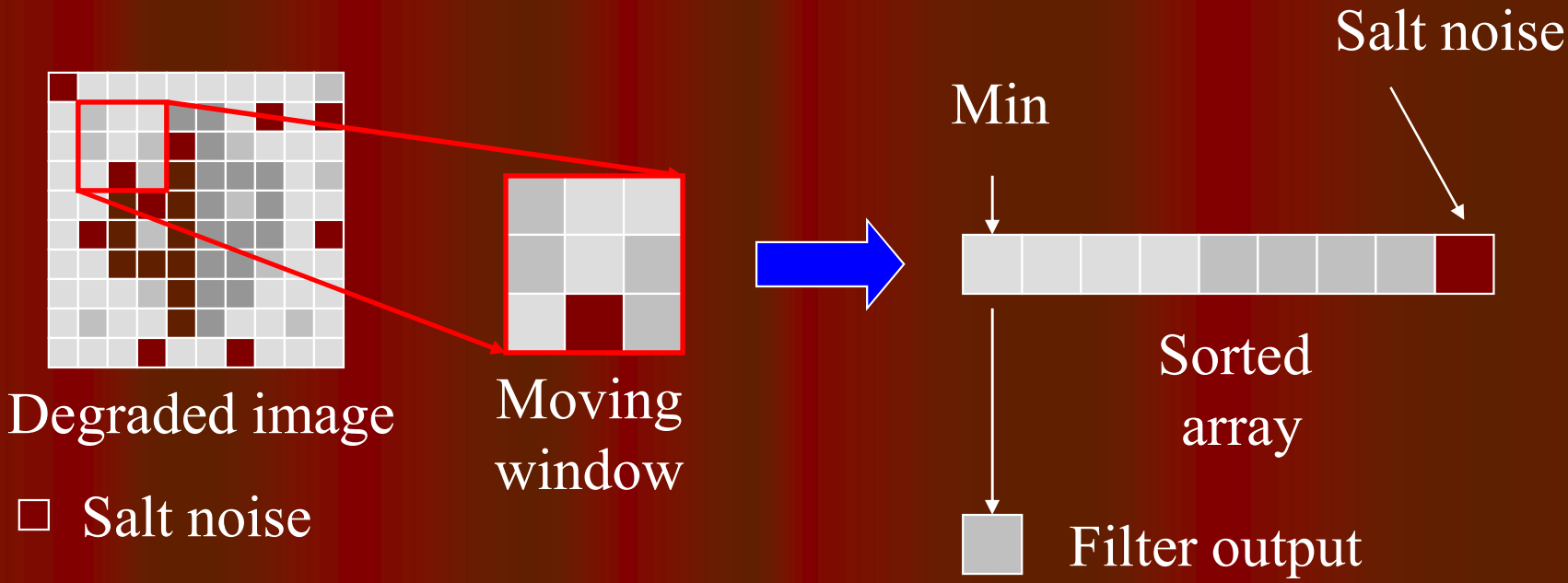
Order Statistics Filter Min Filter

Min filter

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

Order Statistics Filter Min Filter

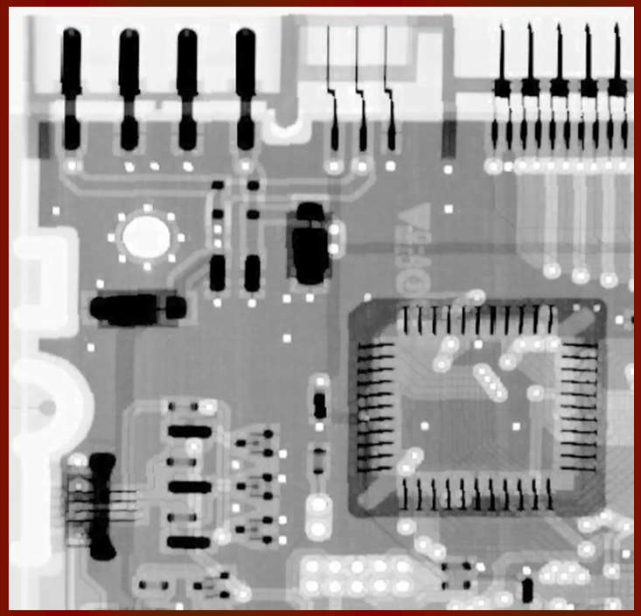
A min filter is good for removing impulse, isolated noise

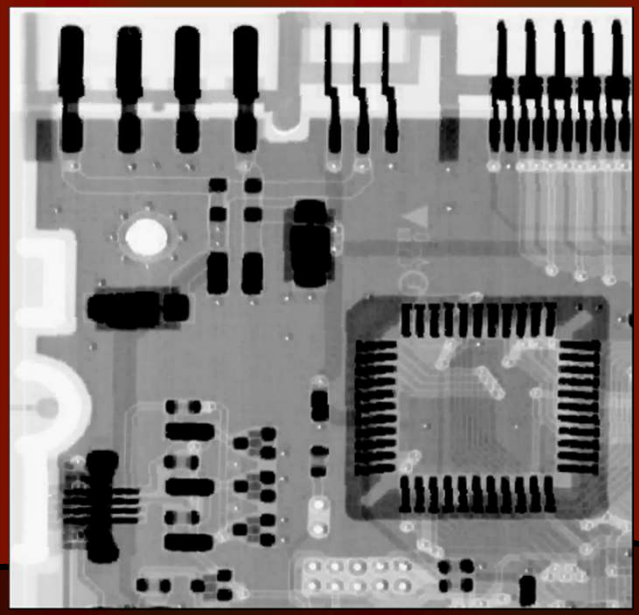


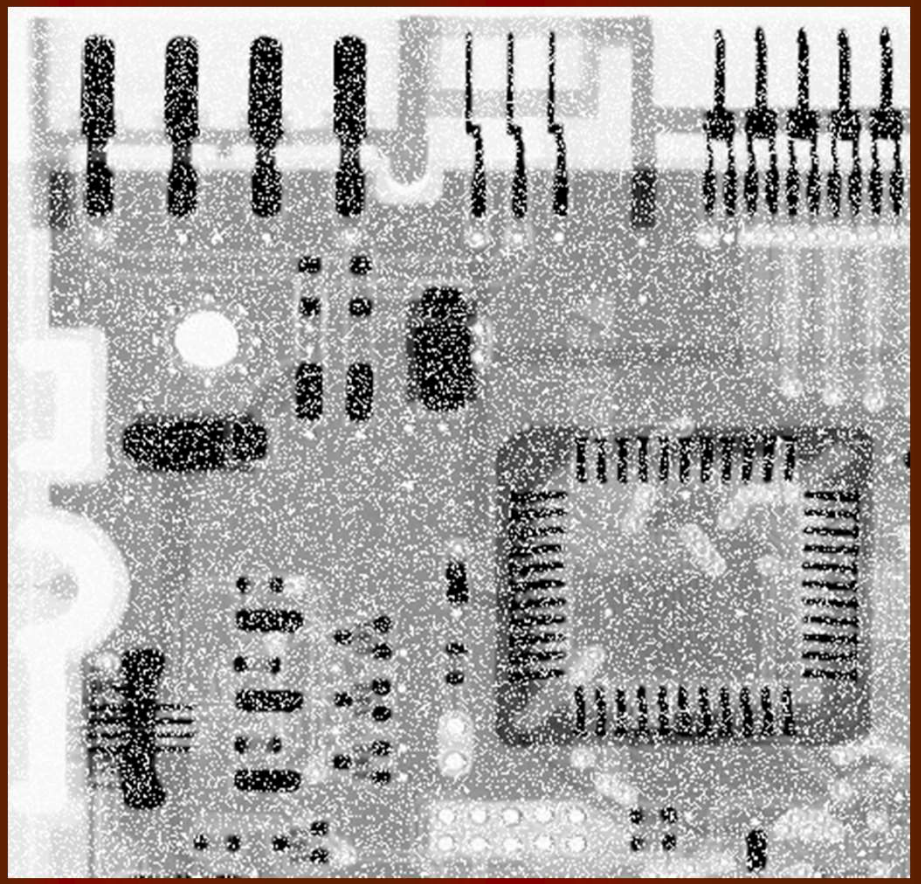
Normally, impulse noise has high magnitude and is isolated. When we sort pixels in the moving window, noise pixels are usually at the right end of the array.

Therefore, it's rare that the noise pixel will be a median value.

Spatial Filtering: Min Filter







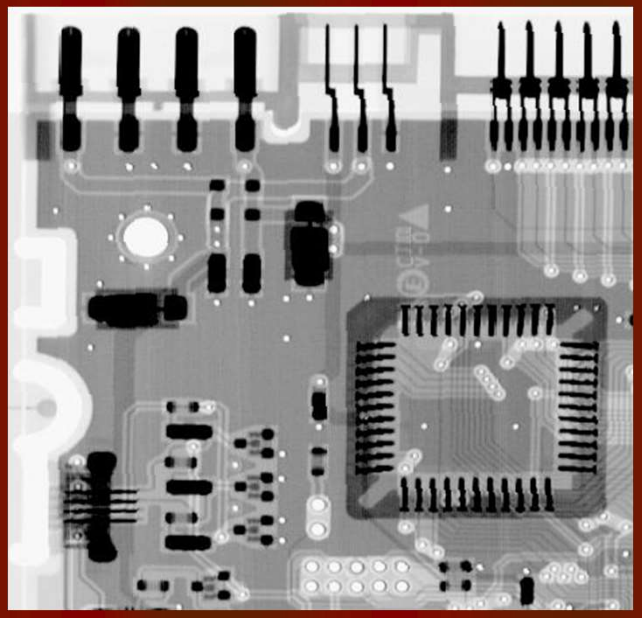
Spatial Filtering: Mid Point Filter

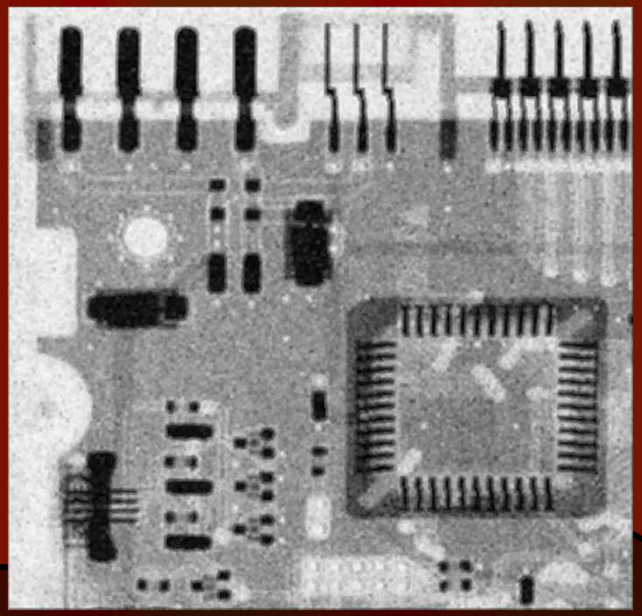
Formula:

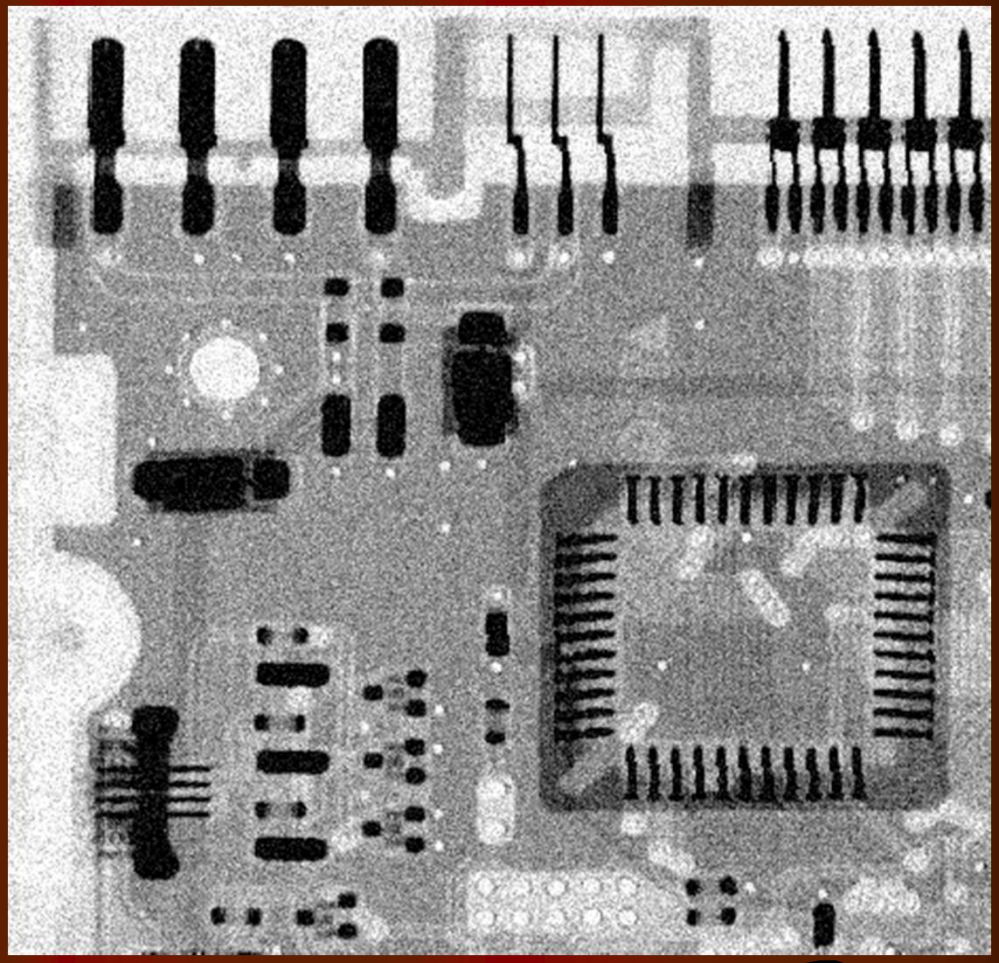
Midpoint filter

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

Spatial Filtering: Mid Point Filter







Alpha Trimmed Mean Filter

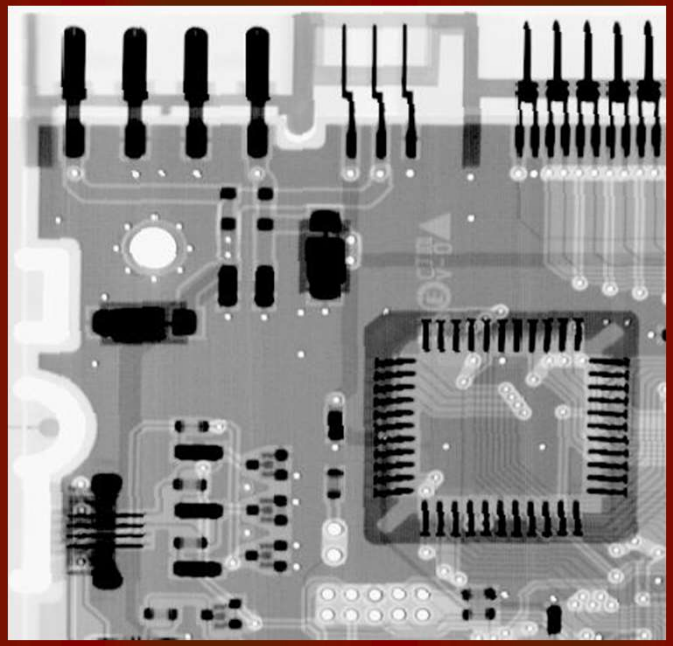
Formula:

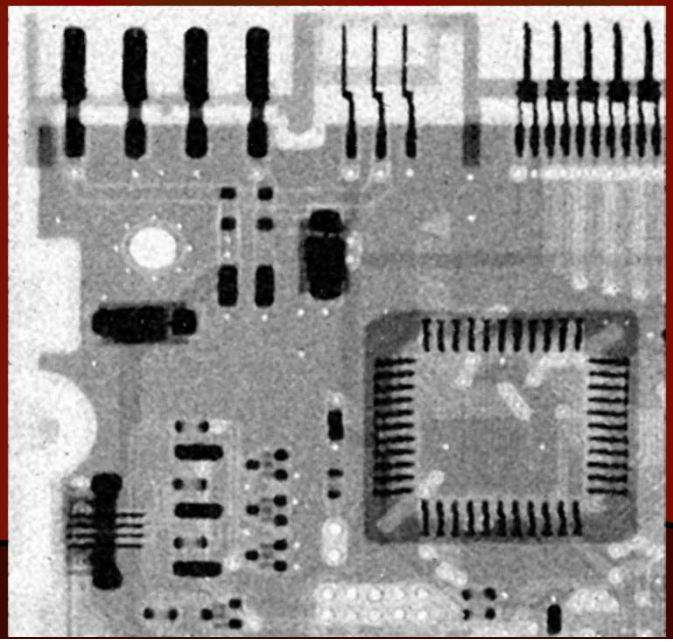
$$\hat{f}(x,y) = \frac{1}{mn-d} \sum_{(s,t)\in S_{xy}} g_r(s,t)$$

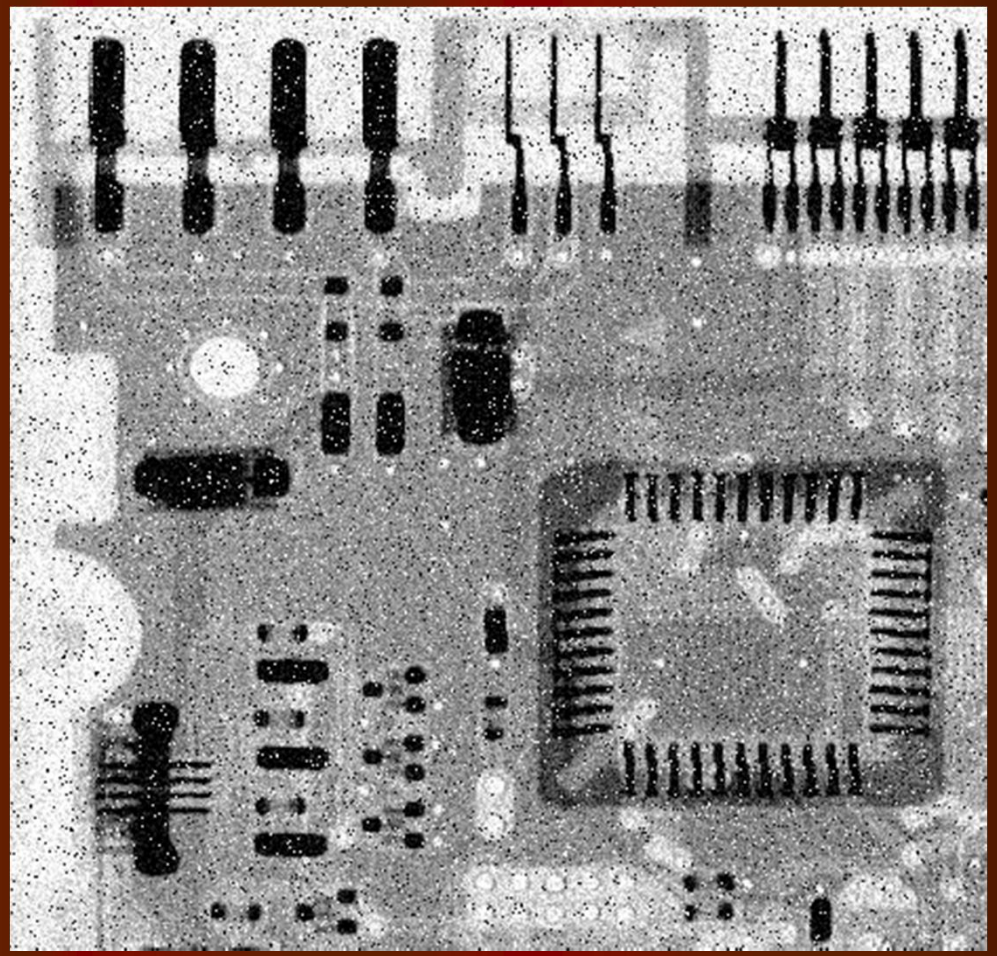
where $g_r(s,t)$ represent the remaining mn-d pixels after removing the $d/2$ highest and $d/2$ lowest values of $g(s,t)$.

This filter is useful in situations involving multiple types of noise such as a combination of salt-and-pepper and Gaussian noise.

Alpha Trimmed Mean Filter



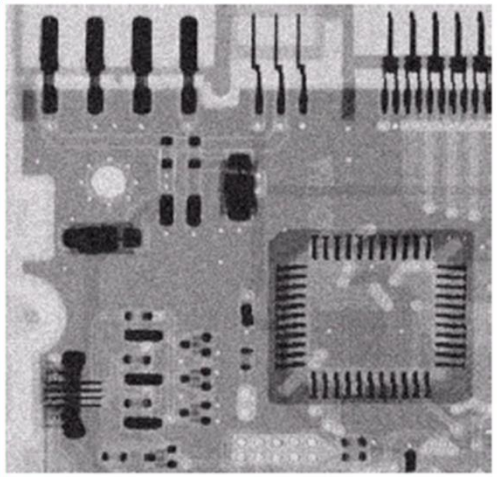


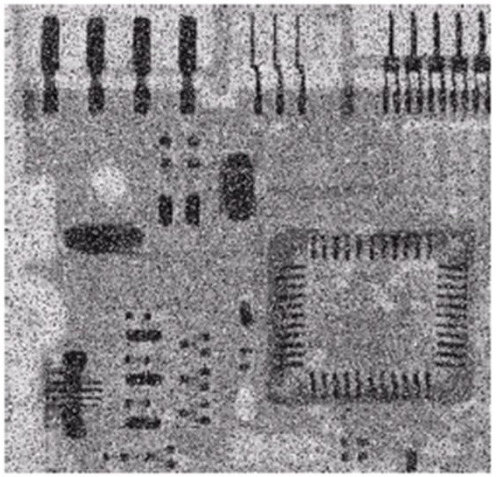


Uniform noise

$\mu=0$

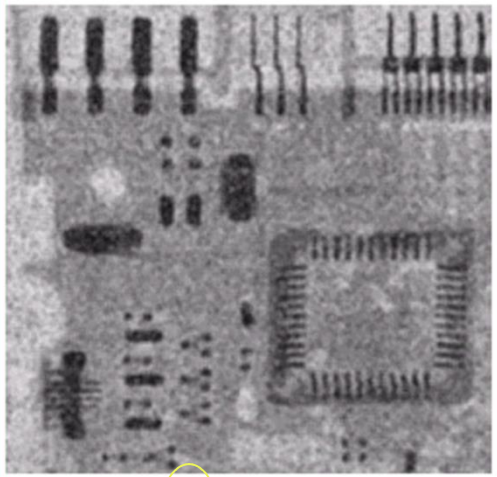
$\sigma^2=800$

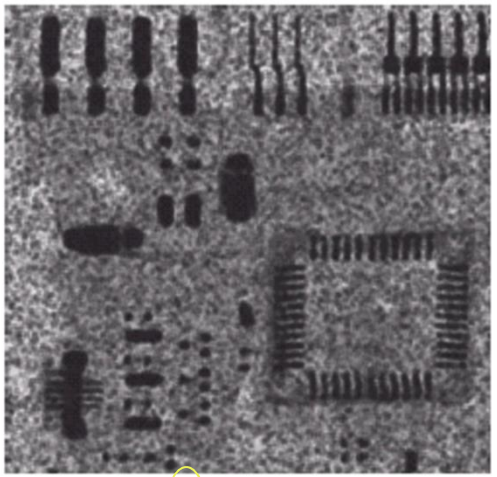




Left +
Bipolar Noise
$P_a = 0.1$
$P_b = 0.1$

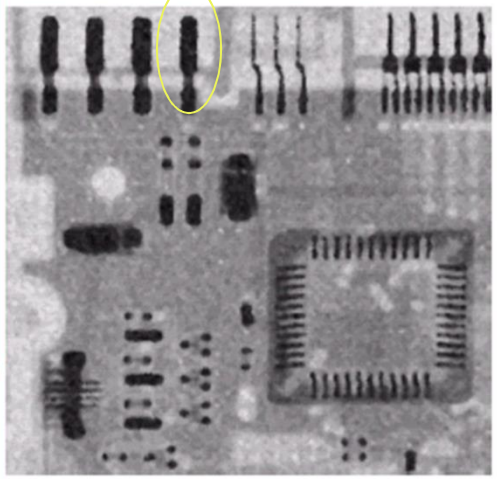
5x5
Arith. Mean
filter

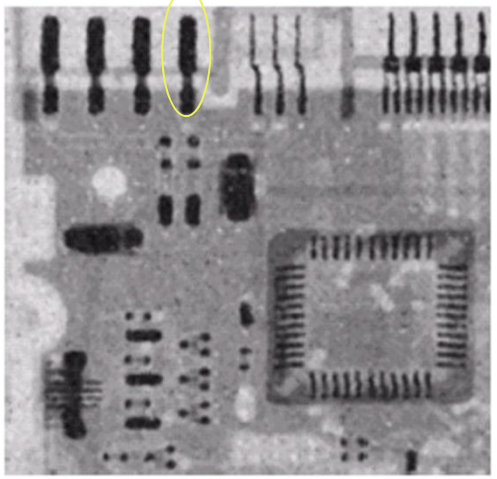




5x5
Geometric
mean

5x5
Median
filter





5x5
Alpha-trim.
Filter
d=5

Spatial Filtering: Adaptive Filters

Adaptive filters

The behavior changes based on statistical characteristics of the image inside the filter region defined by the mxn rectangular window.

The performance is superior to that of the filters discussed

Adaptive Filters:
Adaptive, Local Noise Reduction Filters

S_{xy}: local region

The response of the filter at the center point (x,y) of S_{xy} is based on four quantities:

(a) $g(x, y)$, the value of the noisy image at (x, y);

(b) σ_η^2, the variance of the noise corrupting $f(x, y)$ to form $g(x, y)$;

(c) m_L, the local mean of the pixels in S_{xy};

(d) σ_L^2, the local variance of the pixels in S_{xy}.

Adaptive Filters:
Adaptive, Local Noise Reduction Filters

The behavior of the filter:

(a) if σ_η^2 is zero, the filter should return simply the value of $g(x, y)$.

(b) if the local variance is high relative to σ_η^2, the filter should return a value close to $g(x, y)$;

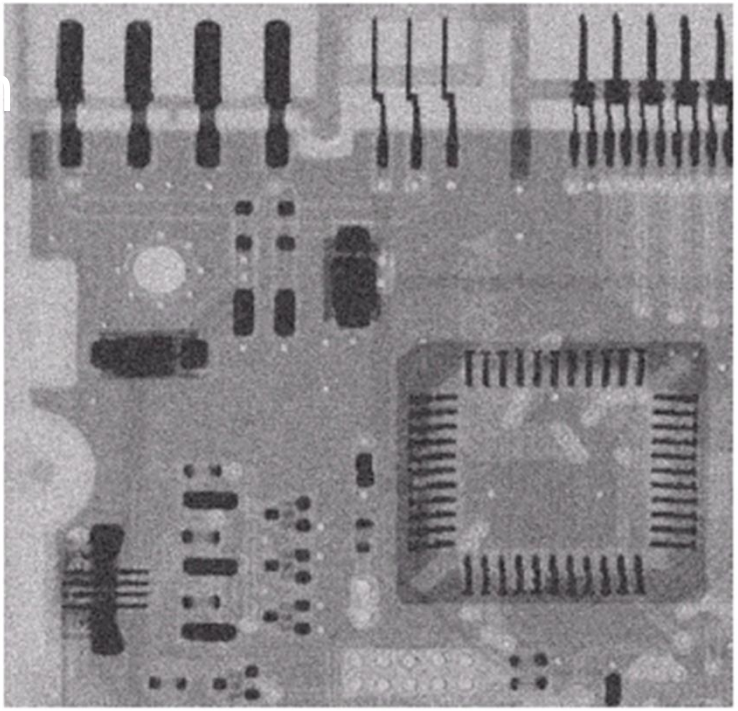
(c) if the two variances are equal, the filter returns the arithmetic mean value of the pixels in S_{xy}.

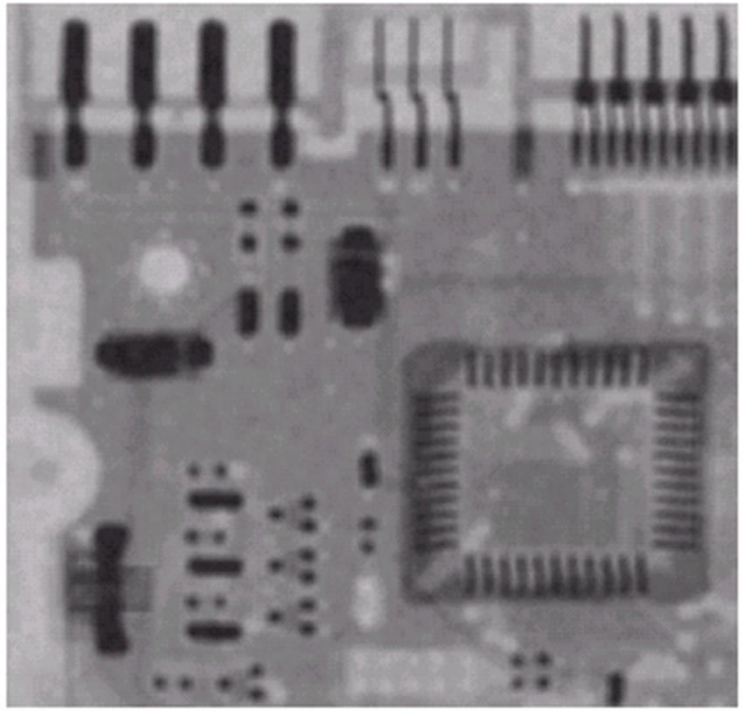
Adaptive Filters:
Adaptive, Local Noise Reduction Filters

An adaptive expression for obtaining $\hat{f}(x,y)$ based on the assumptions:

$$\hat{f}(x,y) = g(x,y) - \frac{\sigma_\eta^2}{\sigma_L^2}\left[g(x,y) - m_L\right]$$

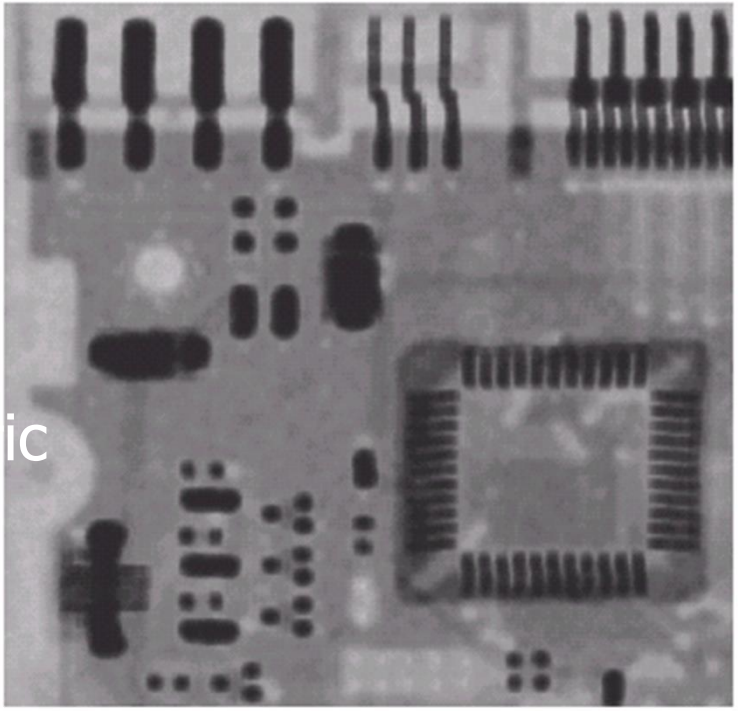
Gaussian noise
$\mu=0$
$\sigma^2=1000$

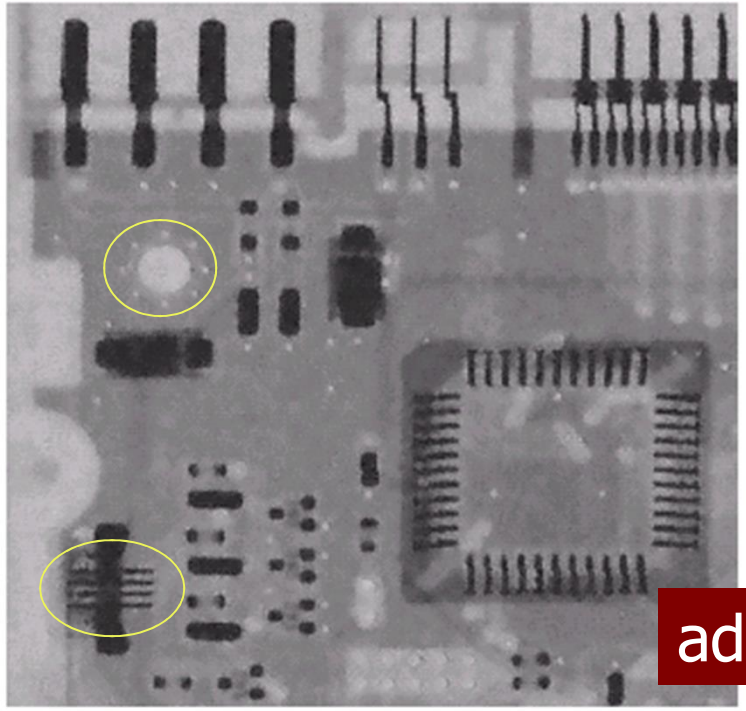




Arith. mean 7x7

Geometric mean 7x7





adaptive

Adaptive Filters:
Adaptive Median Filters

The notation:

$z_{\min}$ = minimum intensity value in S_{xy}

$z_{\max}$ = maximum intensity value in S_{xy}

z_{med} = median intensity value in S_{xy}

z_{xy} = intensity value at coordinates (x, y)

$S_{\max}$ = maximum allowed size of S_{xy}

Adaptive Filters:
Adaptive Median Filters

The adaptive median-filtering works in two stages:

Stage A:

$A1 = z_{med} - z_{min}$; $A2 = z_{med} - z_{max}$

if A1>0 and A2<0, go to stage B

Else increase the window size

if window size $\leq S_{max}$, repeat stage A; Else output z_{med}

Stage B:

$B1 = z_{xy} - z_{min}$; $B2 = z_{xy} - z_{max}$

if B1>0 and B2<0, output z_{xy}; Else output z_{med}

Adaptive Filters:
Adaptive Median Filters

The adaptive median-filtering works in two stages:

Stage A:

$A1 = z_{med} - z_{min}$; $A2 = z_{med} - z_{max}$

if A1>0 and A2<0, go to stage B

Else increase the window size

if window size $\leq S_{max}$, repeat stage A; Else output z_{med}

The median filter output is an impulse or not

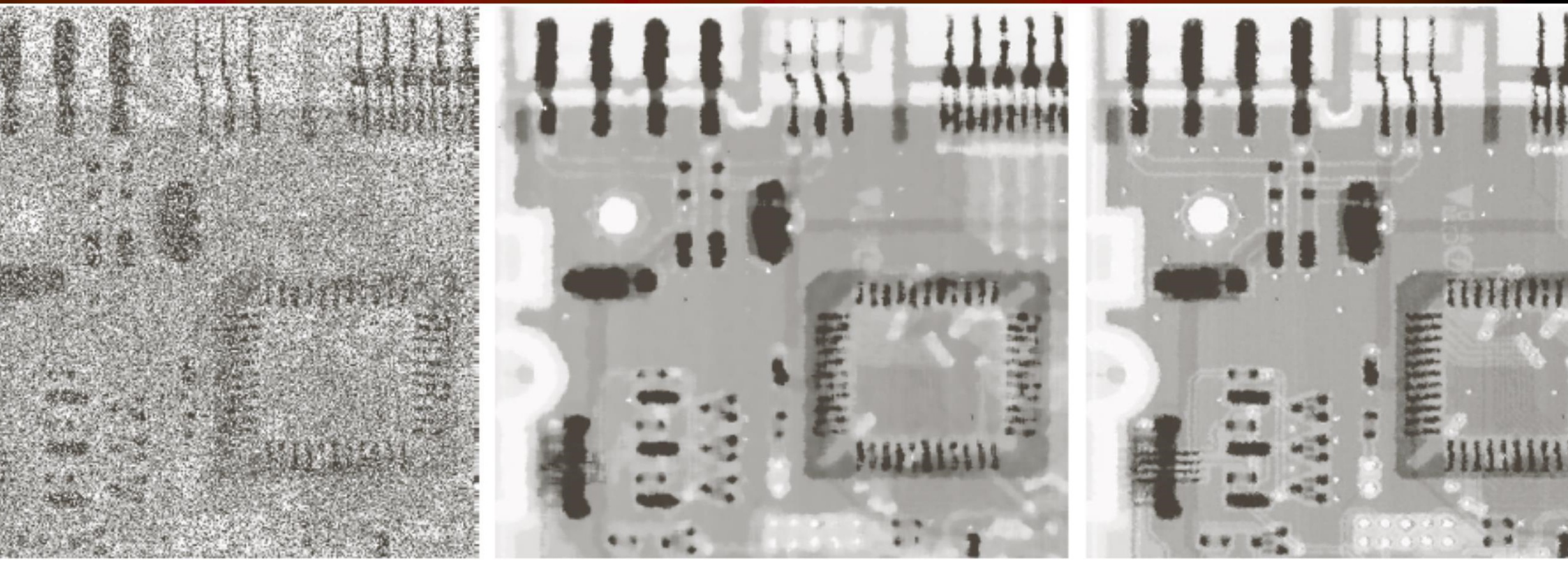
Stage B:

$B1 = z_{xy} - z_{min}$; $B2 = z_{xy} - z_{max}$

if B1>0 and B2<0, output z_{xy}; Else output z_{med}

The processed point is an impulse or not

Adaptive Filters:
Adaptive Median Filters



c

E 5.14 (a) Image corrupted by salt-and-pepper noise with probabilities $P_a = P_b = 0.25$. (b) Res
ng with a 7×7 median filter. (c) Result of adaptive median filtering with $S_{\max} = 7$.

Periodic Interference/Noise

- Periodic noise or interference occurs in images due to electrical or electromechanical interference during image acquisition.
- It is an example of spatially dependent noise.

- This type of noise can be very effectively removed using frequency domain filtering.

Bandreject filters

- Bandreject filters remove (or attenuate) a band of frequencies, around some frequency, say D_0 .

Periodic Noise Reduction by Frequency Domain Filtering

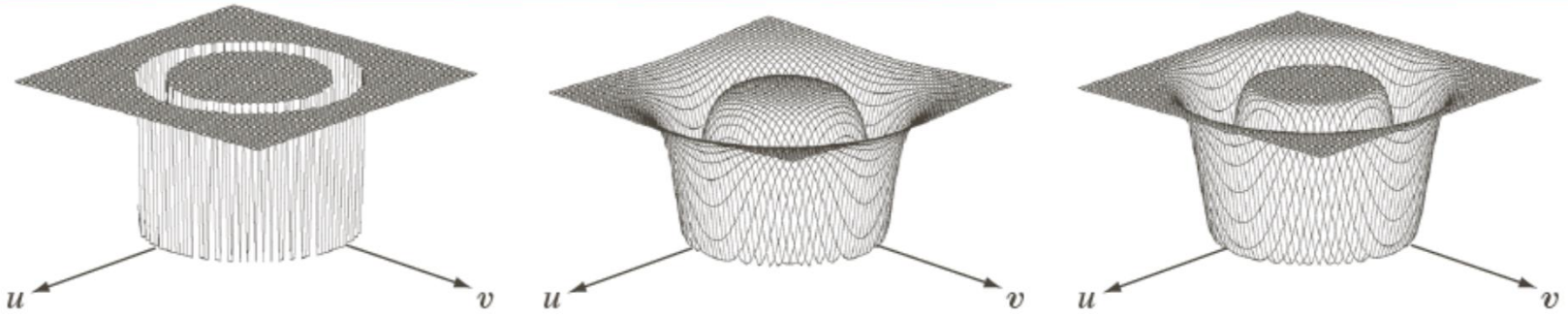
The basic idea

Periodic noise appears as concentrated bursts of energy in the Fourier transform, at locations corresponding to the frequencies of the periodic interference

Approach

A selective filter is used to isolate the noise

Perspective Plots of Bandreject Filters



a b c

FIGURE 5.15 From left to right, perspective plots of ideal, Butterworth (of order 1), and Gaussian bandreject filters.

- An ideal bandreject filter is given by:

$$H(u,v)=\begin{cases}1 & if \quad D(u,v)<D_0-W/2\\ 0 & if \quad D_0-W/2<D(u,v)<D_0+W/2\\ 1 & if \quad D(u,v)>D_0+W/2\end{cases}$$

where

$$D(u,v)=\sqrt{u^2+v^2}$$

- W is usually referred to the width of the (stop) band and D_0 as the center frequency.

- A Butterworth bandreject filter of order n is given by

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

- A Gaussian bandreject filter is given by

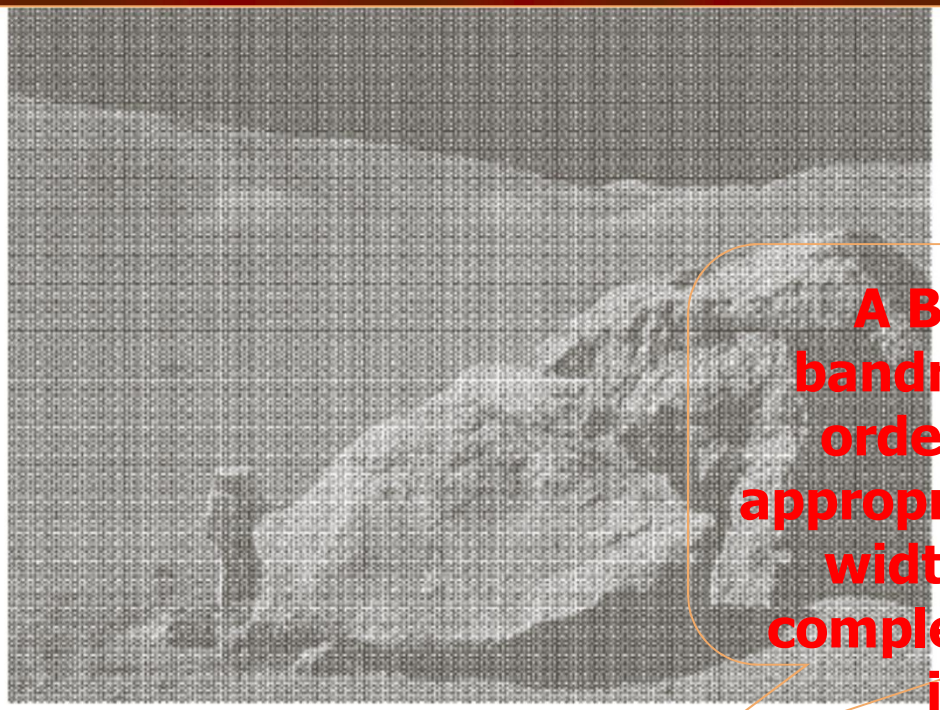
$$H(u,v) = 1 - \exp\left(-\frac{1}{2}\left[\frac{D^2(u,v) - D_0^2}{D(u,v)W}\right]\right)$$

Example

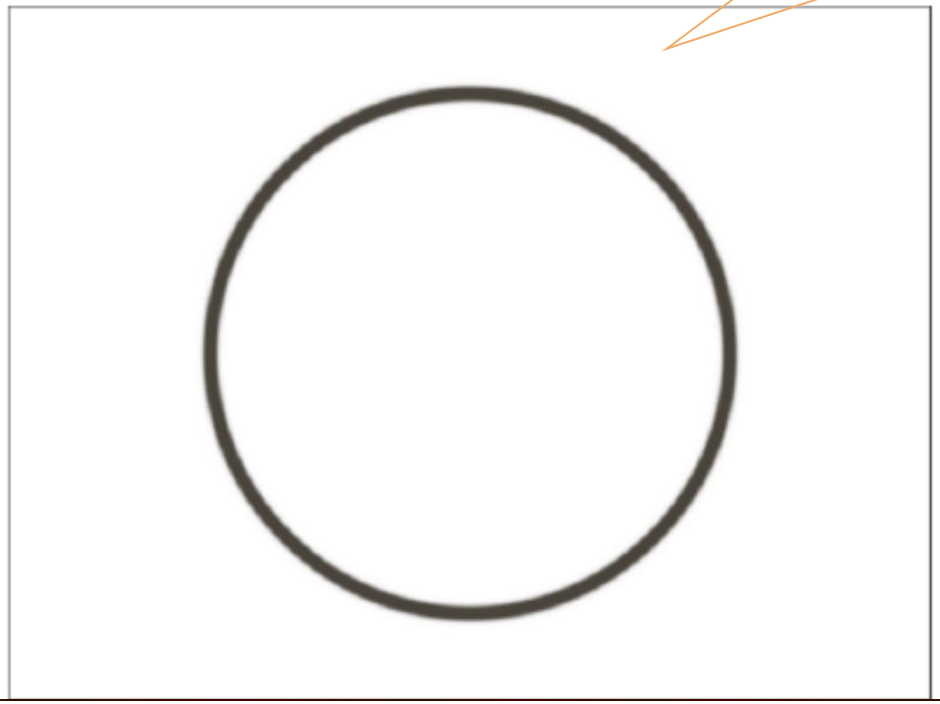
- Bandreject filters are ideally suited for filtering out periodic interference.
- Recall that the Fourier transform of a pure sine or cosine function is just a pair of impulses.
- Therefore the interference is "localized" in the spectral domain and one can easily identify this region and filter it out.

b
d

URE 5.16
Image
rupted by
usoidal noise.
Spectrum of (a).
Butterworth
dreject filter
ite represents
(d) Result of
ering.
riginal image
rtesy of
SA.)



A Butterworth bandreject filter of order 4, with the appropriate radius and width to enclose completely the noise impulses



Bandpass filters

- Bandpass filters are the exact opposite of bandreject filters. They pass a band of frequencies, around some frequency, say D_0 (rejecting the rest).

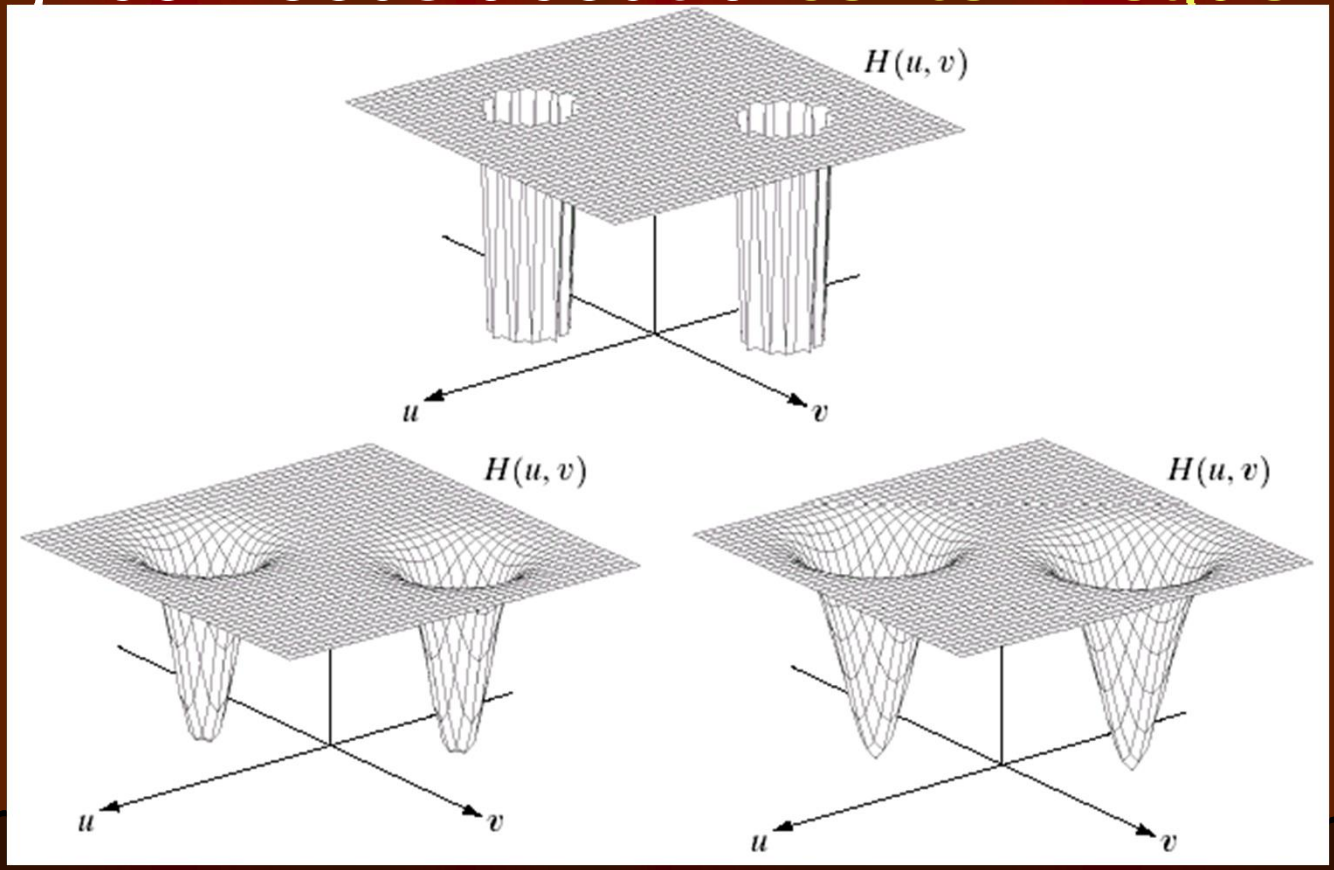
- $H_{bp}(u, v) = 1 - H_{br}(u, v)$

- Bandpass filter is usually used to isolate components of an image that correspond to a band of frequencies.

- It can also be used to isolate noise interference, so that more detailed analysis of the interference can be performed, independent of the image.

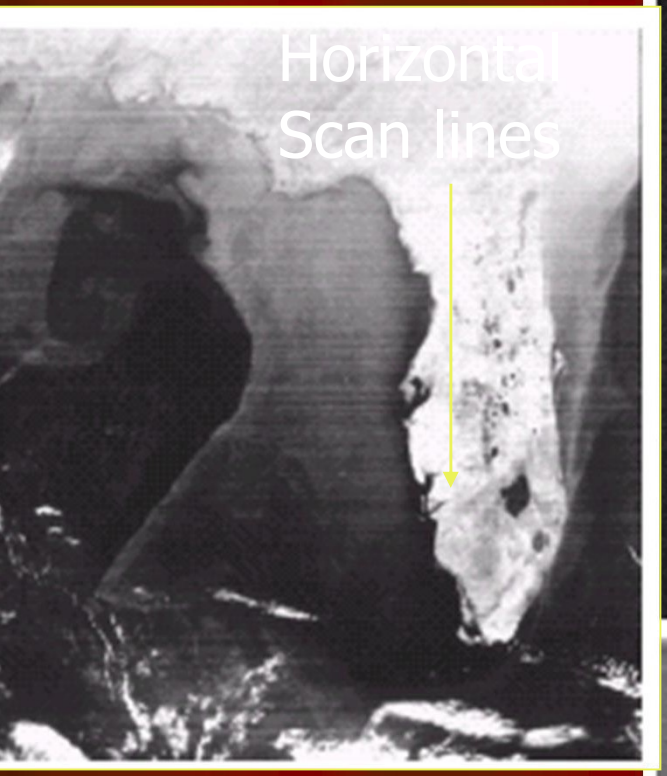
Notch filters

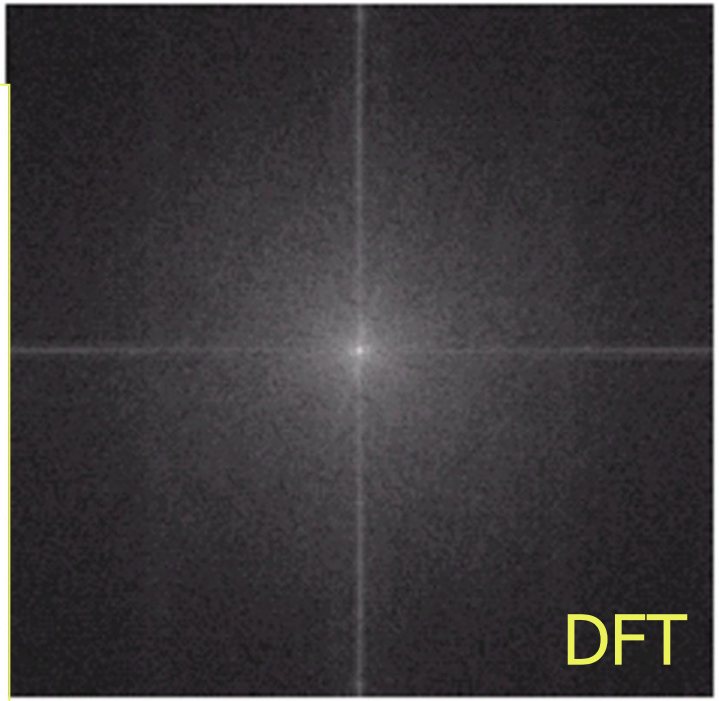
- Reject(or pass) frequencies in predefined neighborhoods about a center frequency



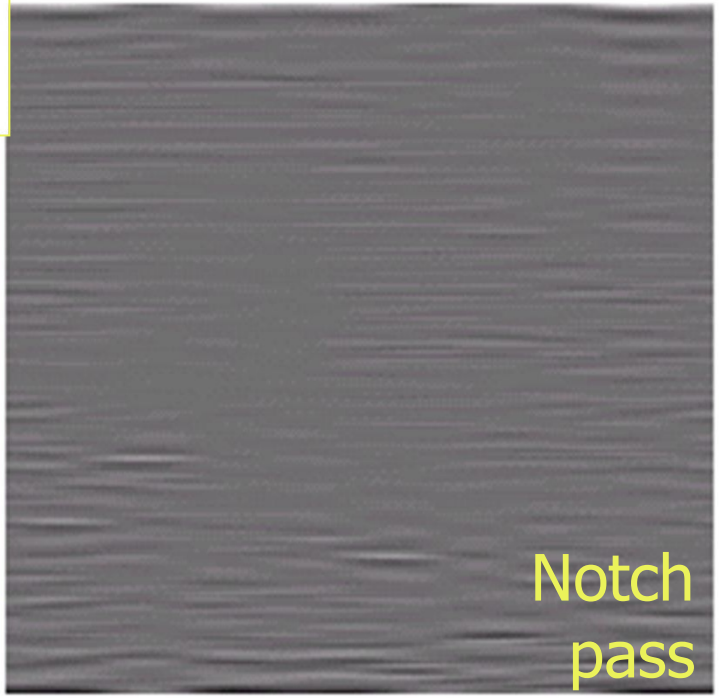
Butterworth

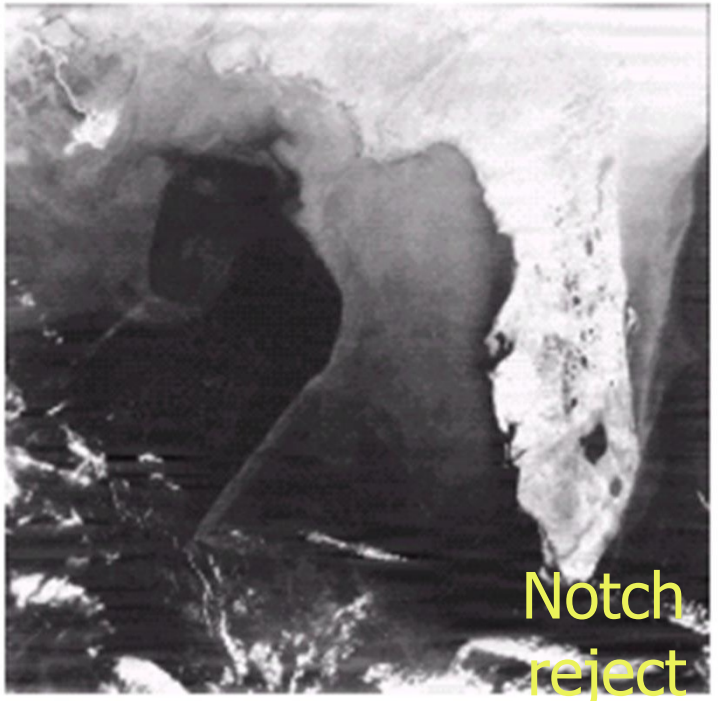
Gaussian


Horizontal
Scan lines


DFT


Notch
pass


Notch
pass


Notch
reject