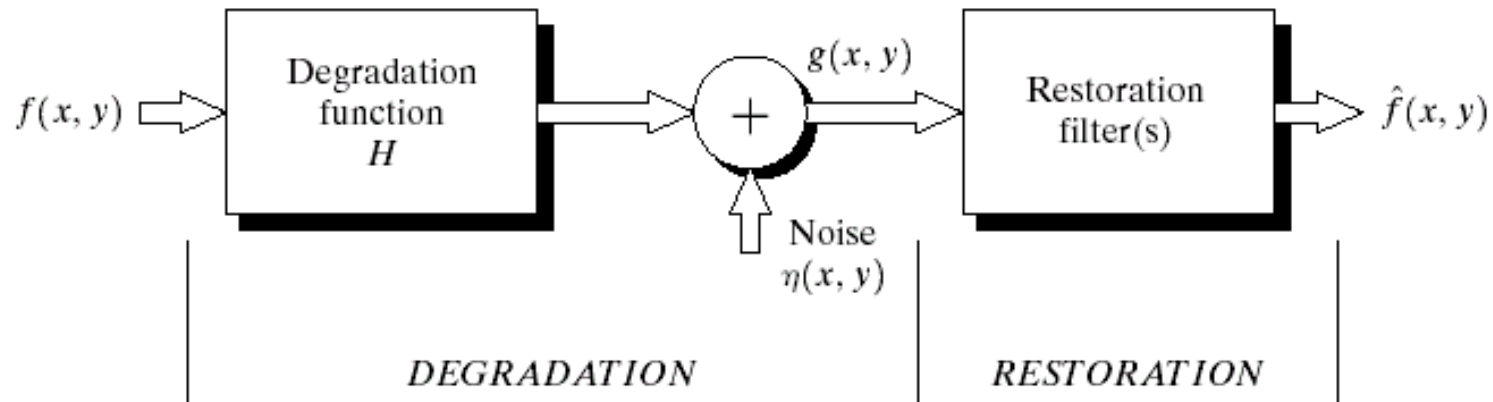

Digital Image Processing

UNIT 5:

Image Restoration

Concept of Image Restoration

*Image restoration is to restore a degraded image back to the original image while **image enhancement** is to manipulate the image so that it is suitable for a specific application.*



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.

Noise Models

Noise cannot be predicted but can be approximately described in statistical way using the probability density function (PDF)

Gaussian noise:

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

Rayleigh noise

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

Erlang (Gamma) noise

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

Noise Models (cont.)

Exponential noise

$$p(z) = ae^{-az}$$

Uniform noise

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

Impulse (salt & pepper) noise

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

PDF: Statistical Way to Describe Noise

PDF tells how much each z value occurs.

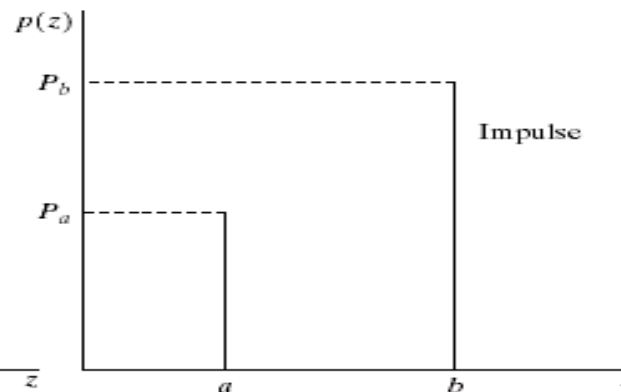
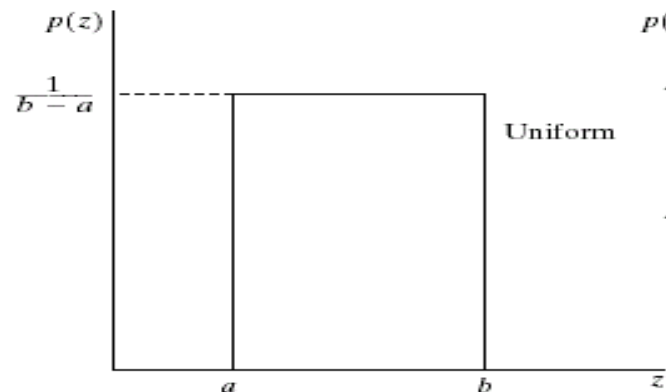
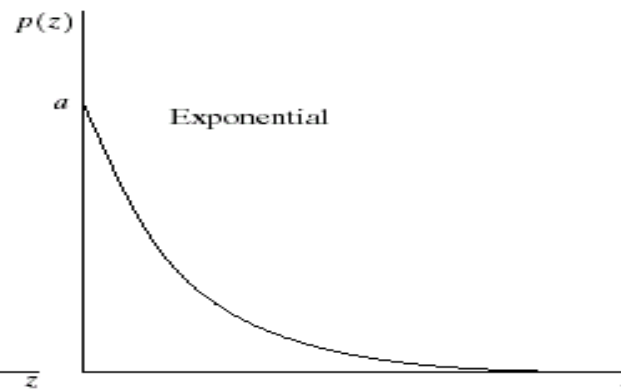
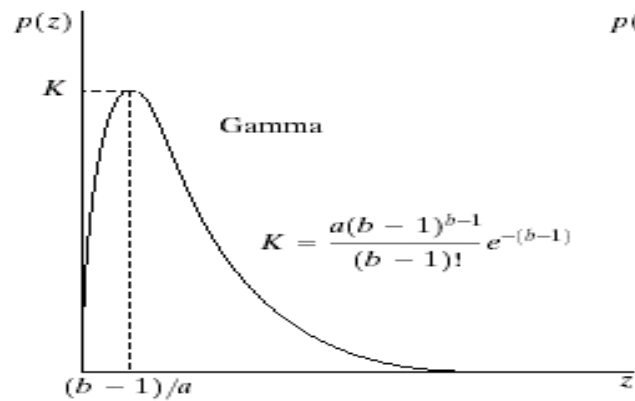
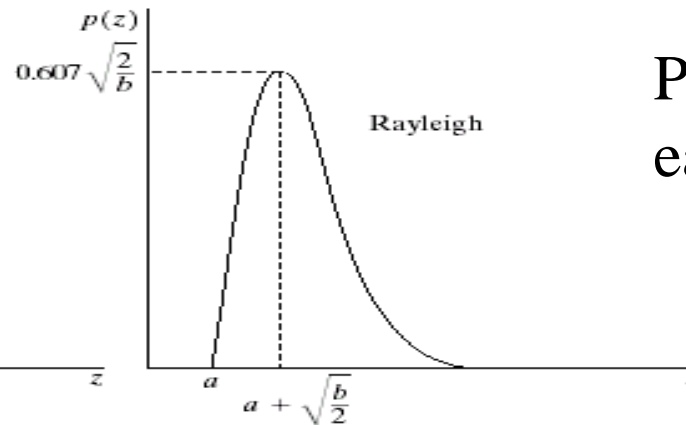
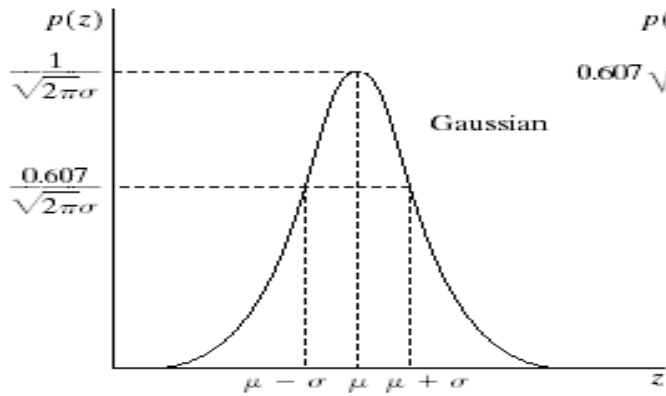
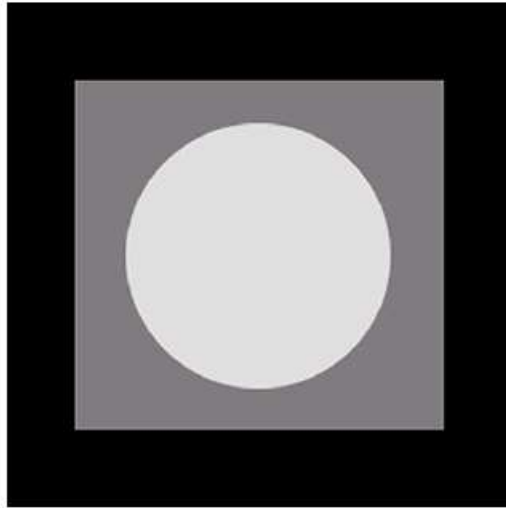


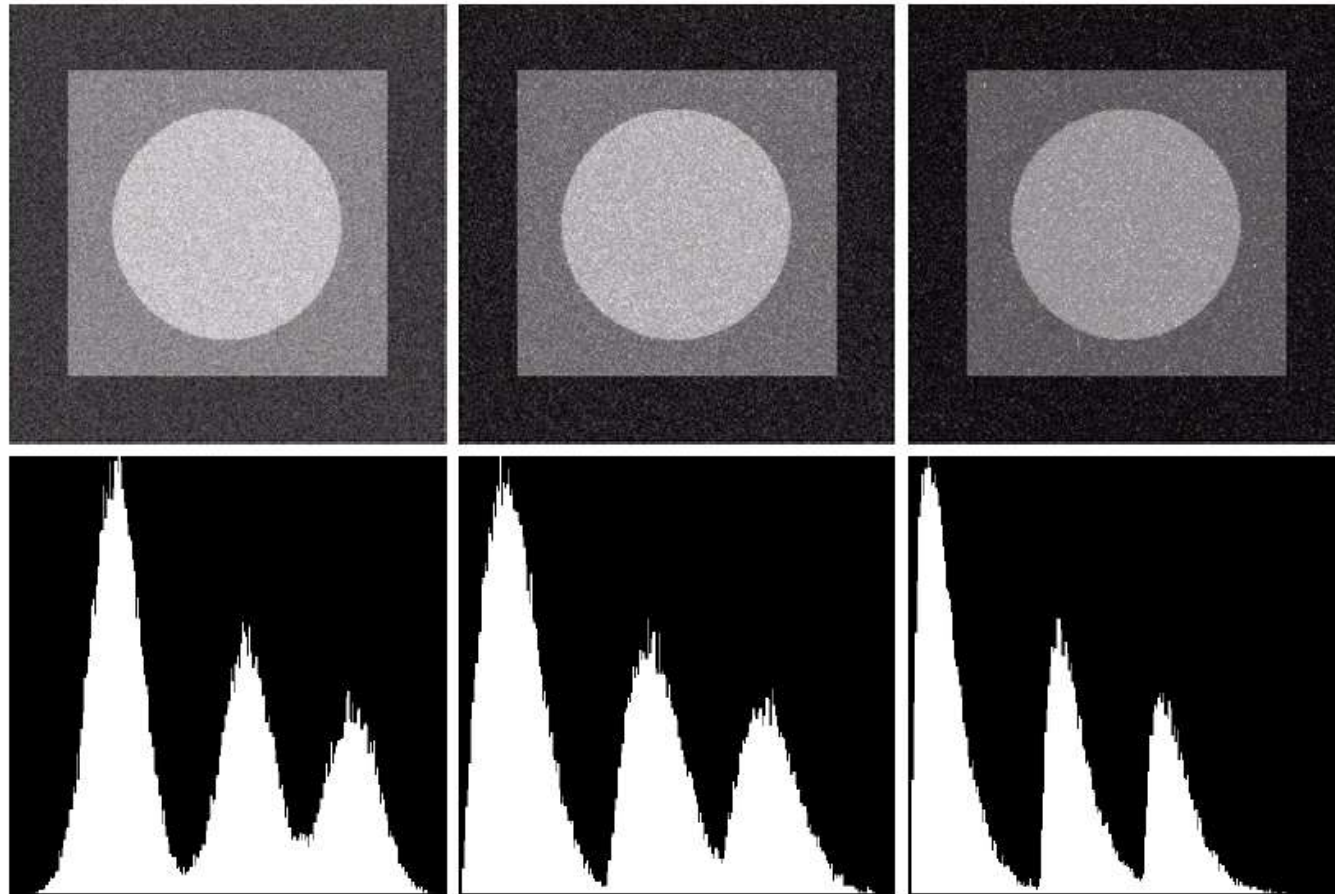
Image Degradation with Additive Noise



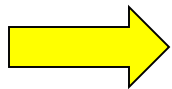
Original image

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

Degraded images



Histogram

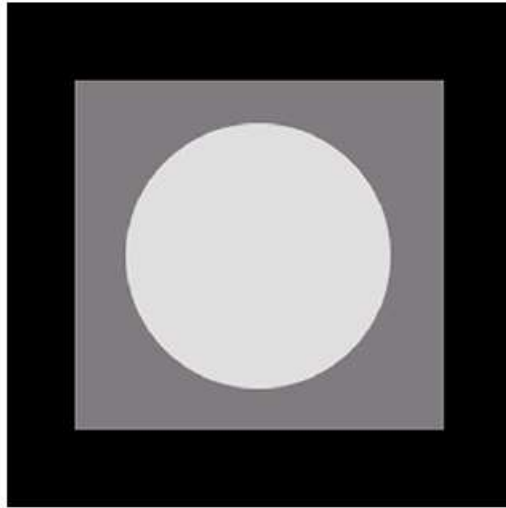


Gaussian

Rayleigh

Gamma

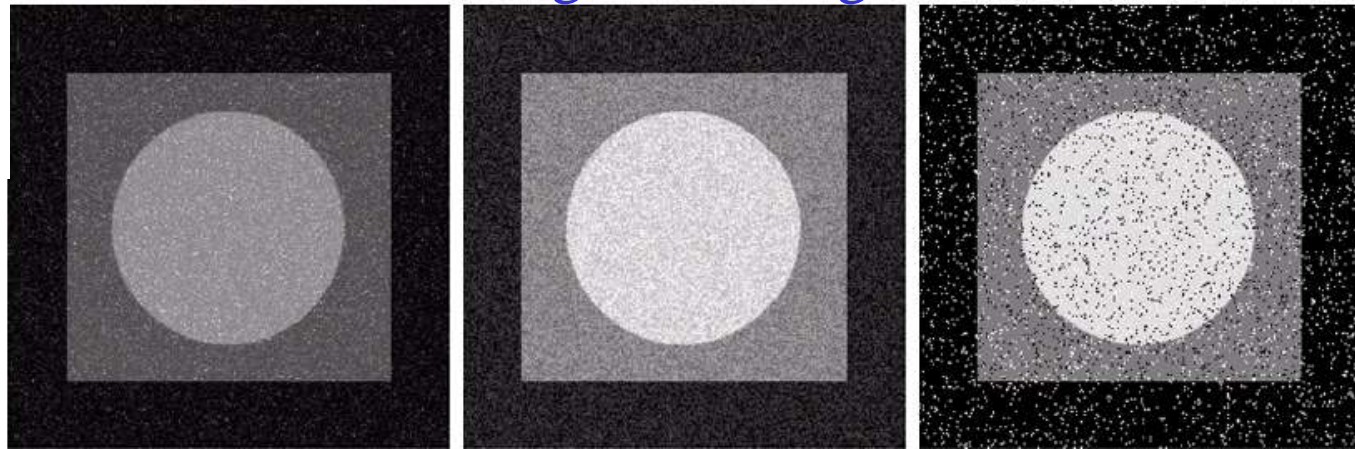
Image Degradation with Additive Noise (cont.)



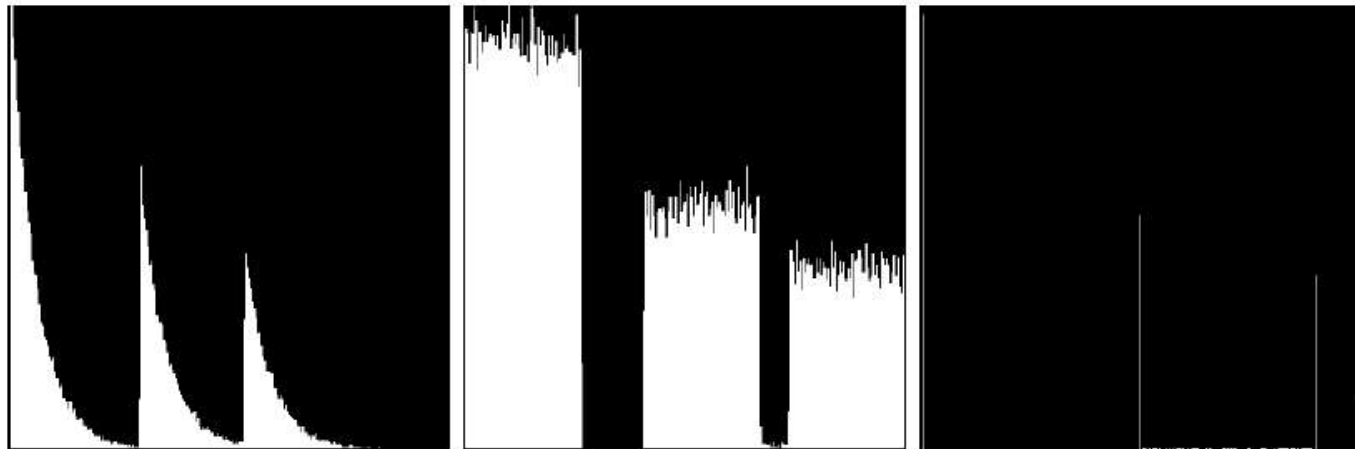
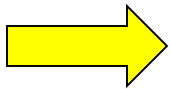
Original image

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

Degraded images



Histogram



Exponential

Uniform

Salt & Pepper

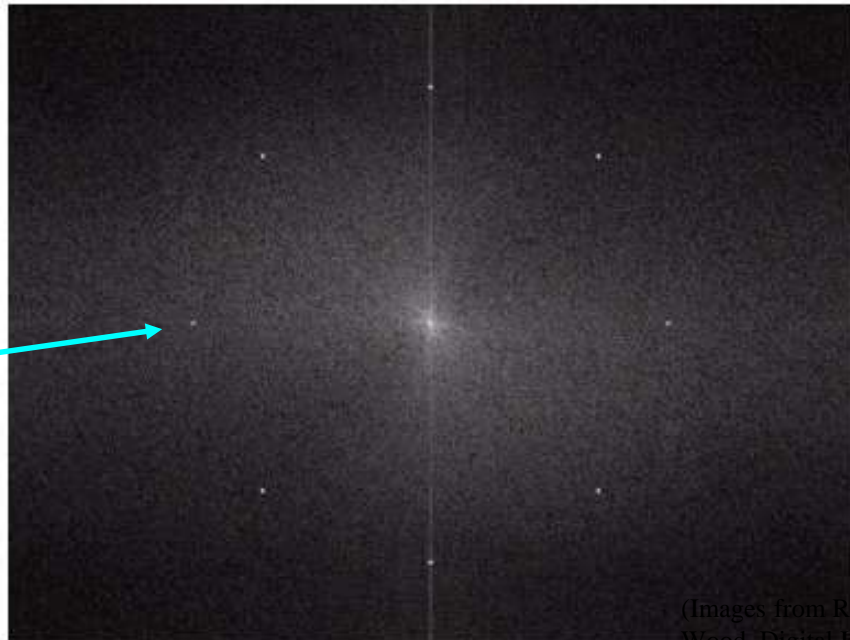
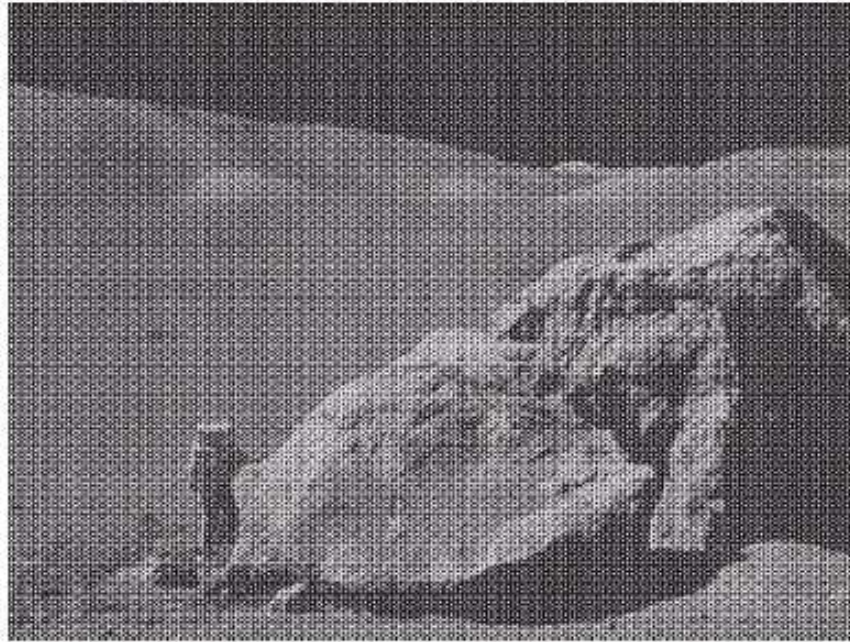
Periodic Noise

a
b

FIGURE 5.5

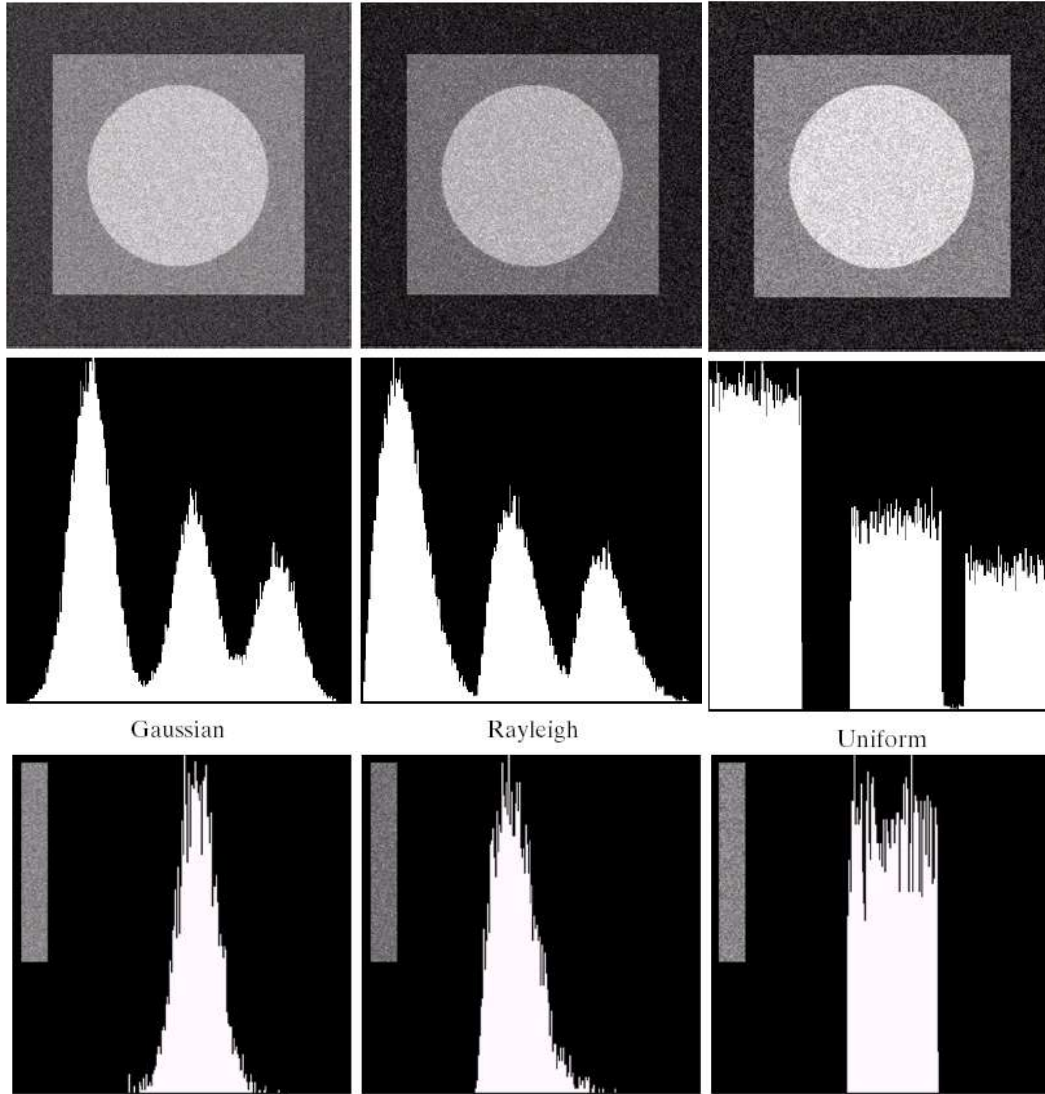
(a) Image corrupted by sinusoidal noise.

(b) Spectrum (each pair of conjugate impulses corresponds to one sine wave). (Original image courtesy of NASA.)



Periodic noise
looks like **dots**
In the frequency
domain

Estimation of Noise

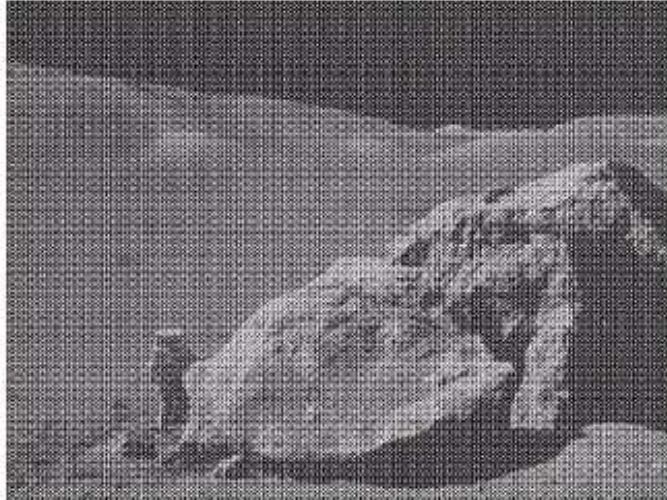


We cannot use the image histogram to estimate noise PDF.

It is better to use the histogram of one area of an image that has constant intensity to estimate noise PDF.

Periodic Noise Reduction by Freq. Domain Filtering

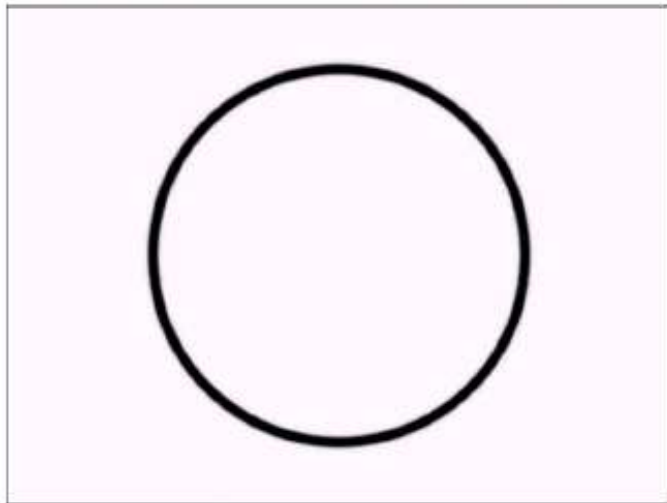
Degraded image



DFT



Periodic noise can be reduced by setting frequency components corresponding to noise to zero.



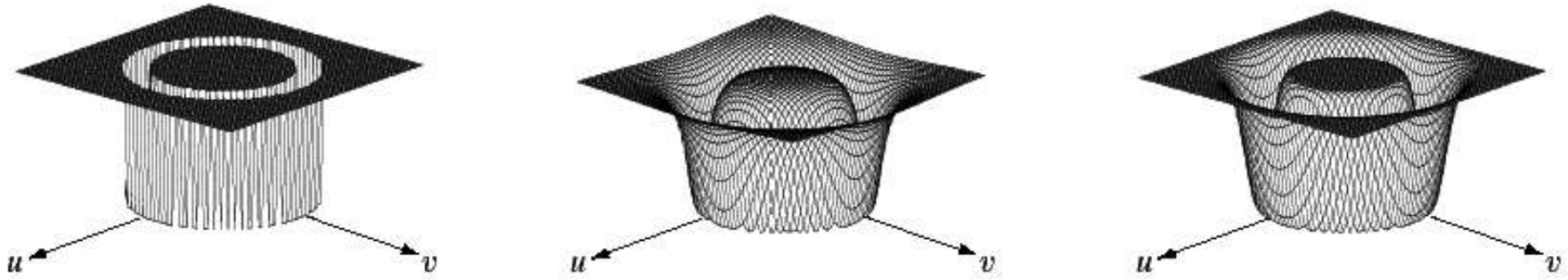
Band reject filter



Restored image

Band Reject Filters

Use to eliminate frequency components in some bands

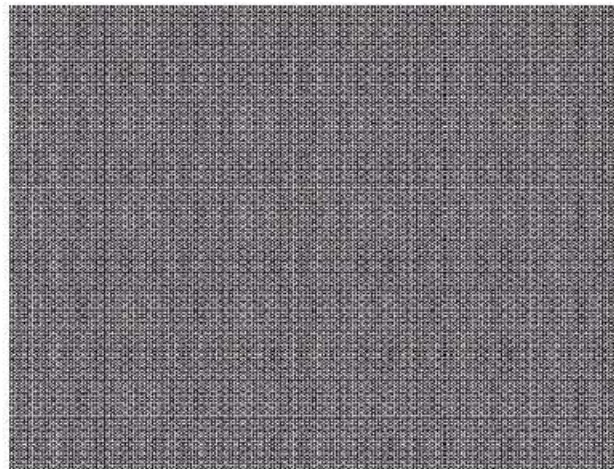


a b c

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

FIGURE 5.17

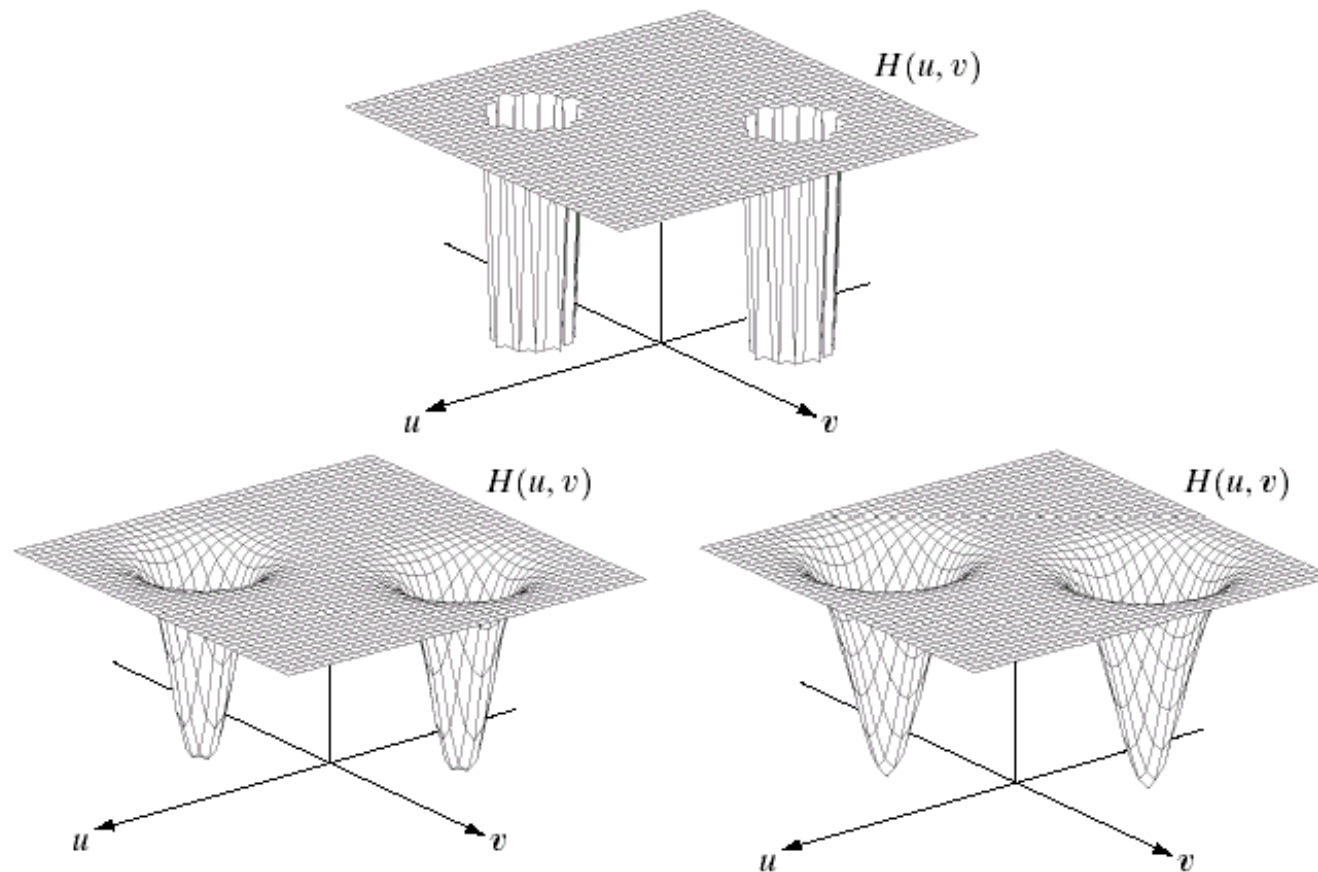
Noise pattern of the image in Fig. 5.16(a) obtained by bandpass filtering.



Periodic noise from the previous slide that is Filtered out.

Notch Reject Filters

A notch reject filter is used to eliminate some frequency components.



a
b c

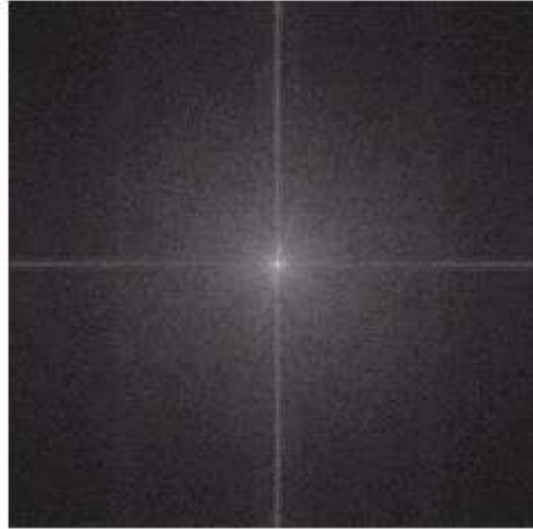
FIGURE 5.18 Perspective plots of (a) ideal, (b) Butterworth (of order 2), and (c) Gaussian notch (reject) filters.

Notch Reject Filter:

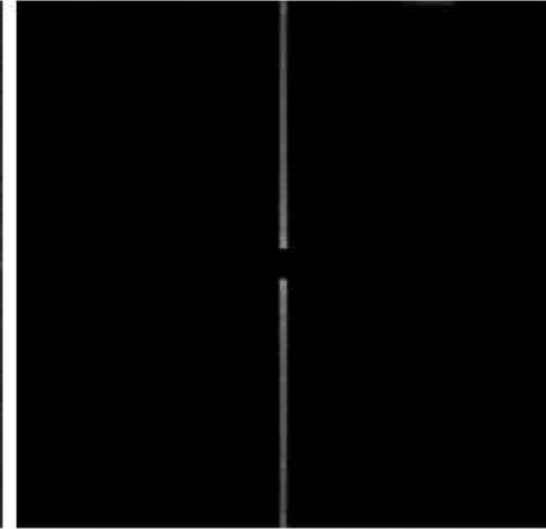
Degraded image



DFT



Notch filter
(freq. Domain)



Noise



Restored image

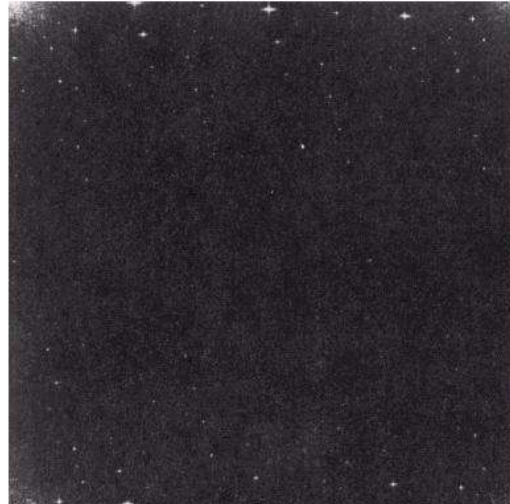
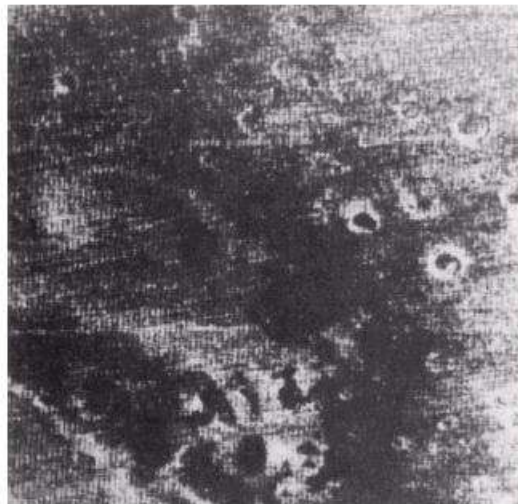
(Images from Rafael C. Gonzalez and Richard E. Wood, Digital Image Processing, 2nd Edition.)

Example: Image Degraded by Periodic Noise

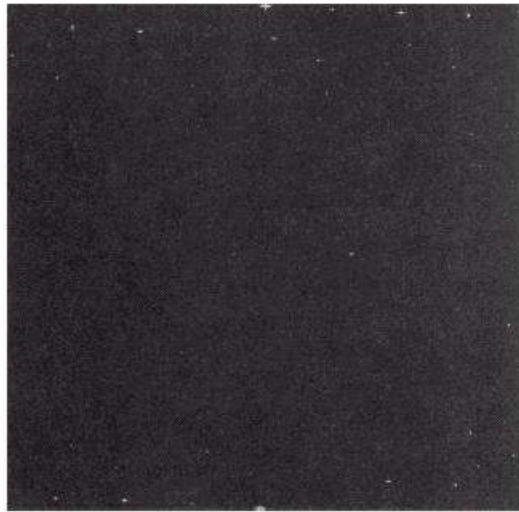
Degraded image

a b

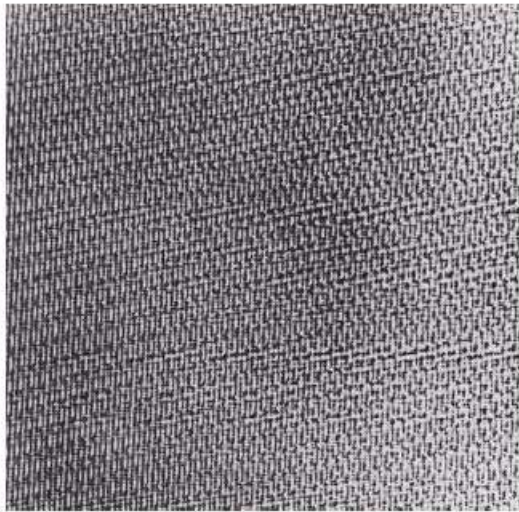
FIGURE 5.20
(a) Image of the
Martian terrain
taken by
Mariner 6.
(b) Fourier
spectrum showing
periodic
interference.
(Courtesy of
NASA.)



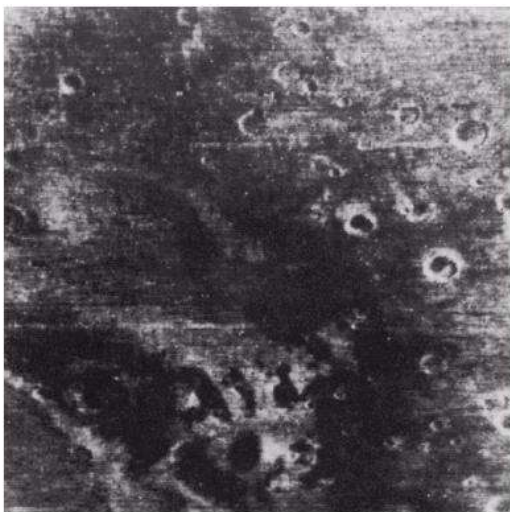
DFT
(no shift)



DFT of noise



Noise



Restored image

(Images from Rafael C. Gonzalez and Richard E. Wood, Digital Image Processing, 2nd Edition.

Mean Filters

Degradation model:

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


To remove this part

Arithmetic mean filter or moving average filter (from Chapter 3)

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

Geometric mean filter

mn = size of moving window

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

Geometric Mean Filter: Example

Original
image

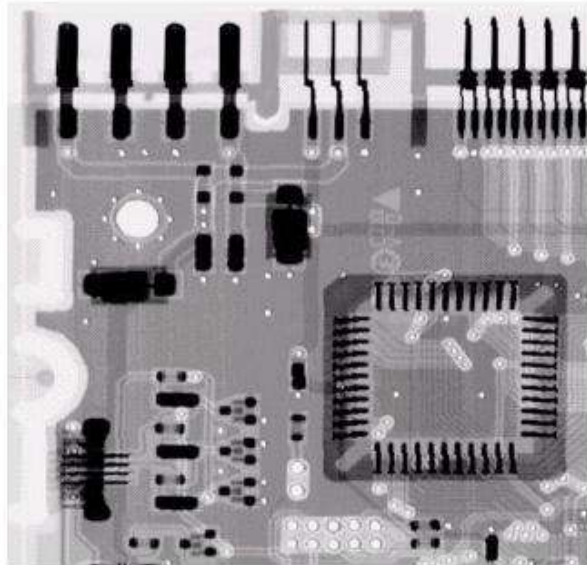


Image
corrupted
by AWGN

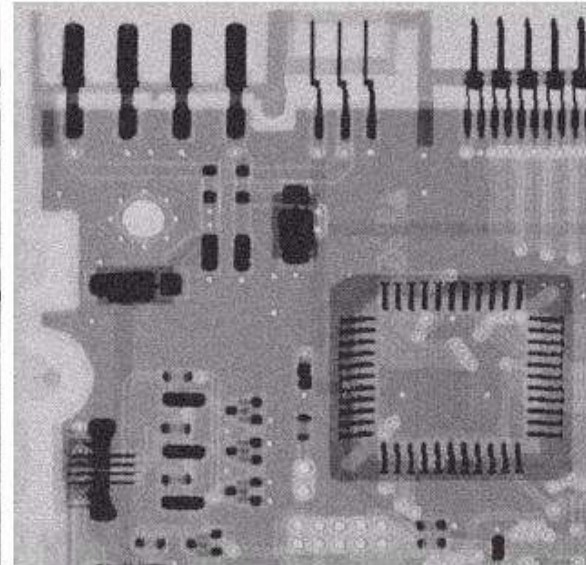


Image
obtained
using a 3x3
arithmetic
mean filter

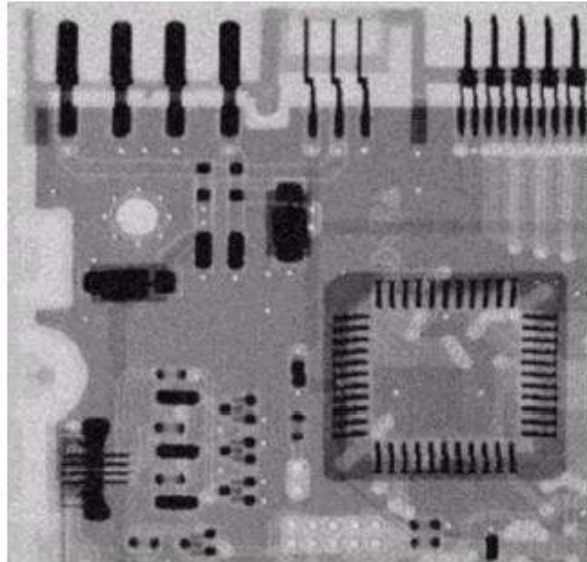
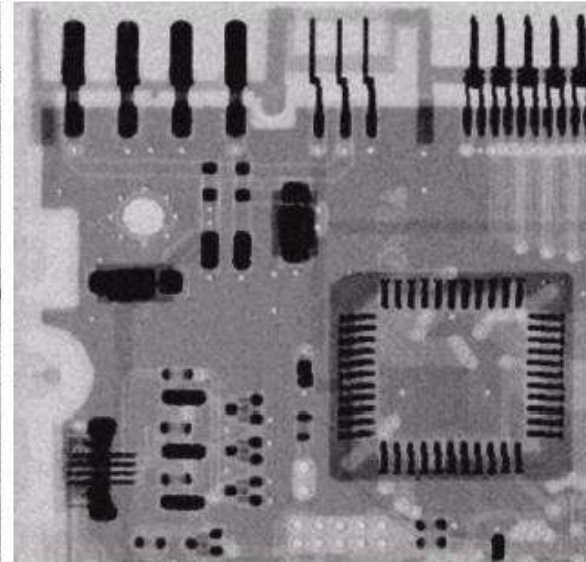


Image
obtained
using a 3x3
geometric
mean filter



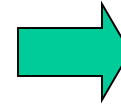
AWGN: Additive White Gaussian Noise

(Images from Rafael C. Gonzalez and Richard E. Wood, Digital Image Processing, 2nd Edition.

Harmonic and Contraharmonic Filters

Harmonic mean filter

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

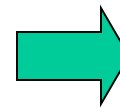


Works well for salt noise
but fails for pepper noise

Contraharmonic mean filter

mn = size of moving window

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



Positive Q is suitable for
eliminating pepper noise.
Negative Q is suitable for
eliminating salt noise.

Q = the filter order

For $Q = 0$, the filter reduces to an arithmetic mean filter.

For $Q = -1$, the filter reduces to a harmonic mean filter.

Contra-harmonic Filters: Example

Image
corrupted
by pepper
noise with
prob. = 0.1

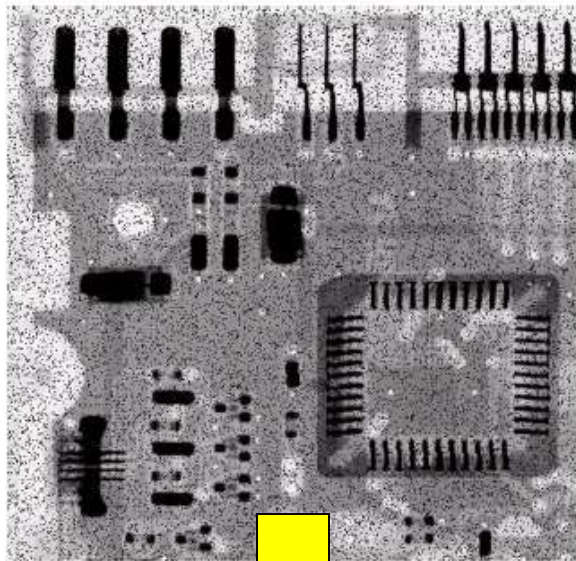


Image
corrupted
by salt
noise with
prob. = 0.1

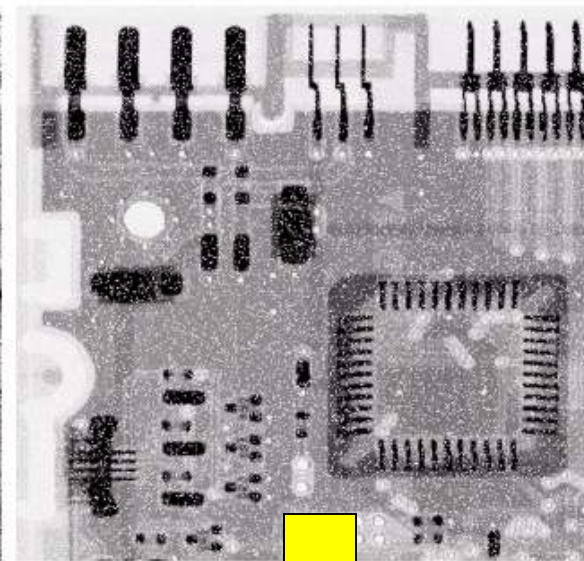


Image
obtained
using a 3x3
contra-
harmonic
mean filter
With $Q = 1.5$

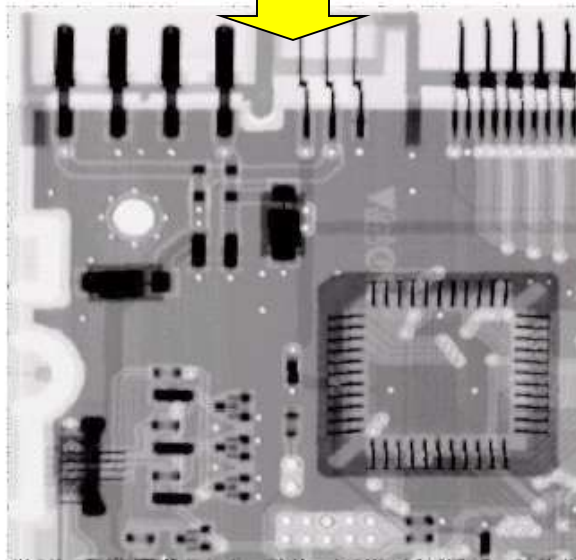
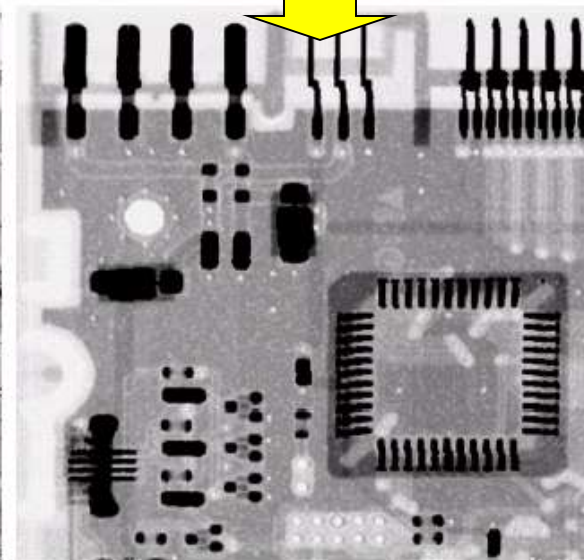


Image
obtained
using a 3x3
contra-
harmonic
mean filter
With $Q = -1.5$



Contra-harmonic Filters: Incorrect Use Example

Image corrupted by pepper noise with prob. = 0.1

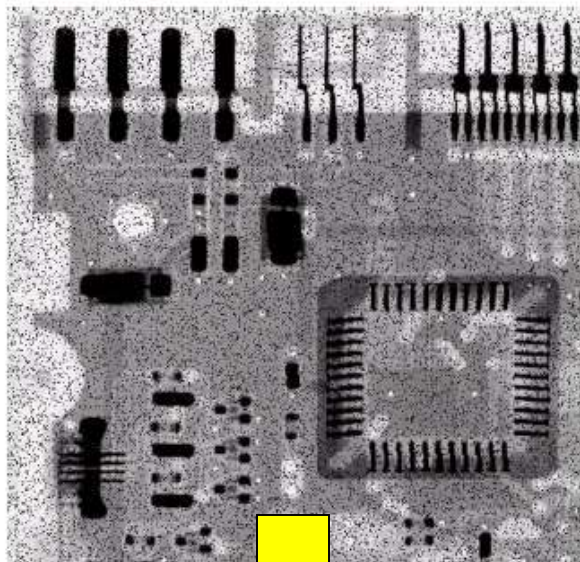


Image obtained using a 3x3 contra-harmonic mean filter With $Q=-1.5$

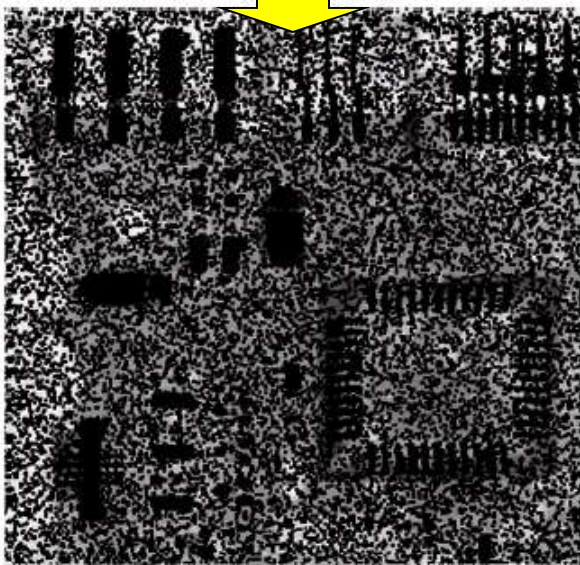


Image corrupted by salt noise with prob. = 0.1

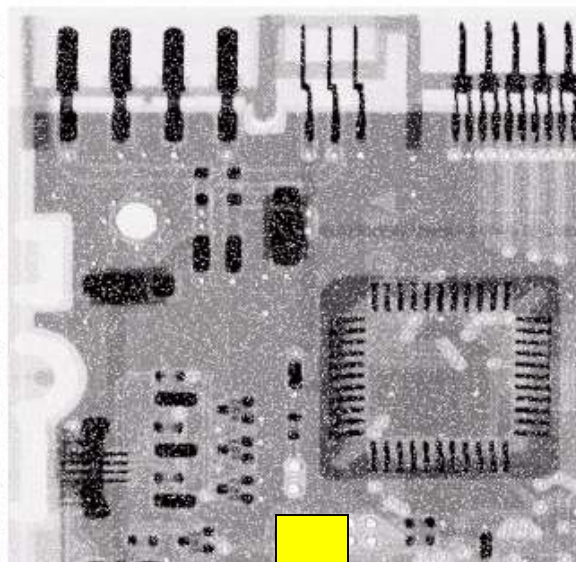
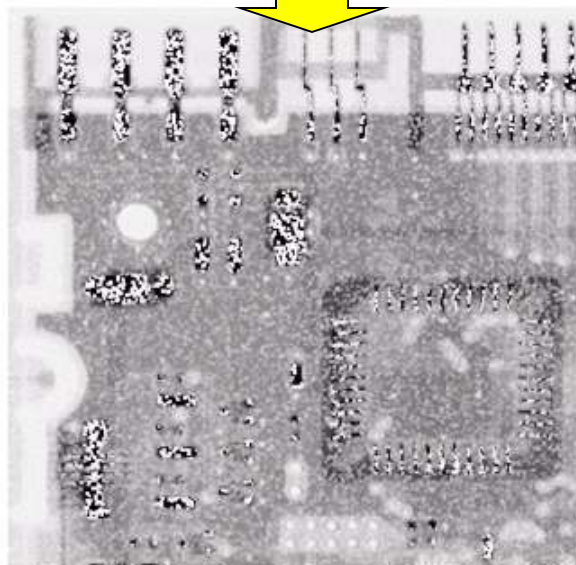
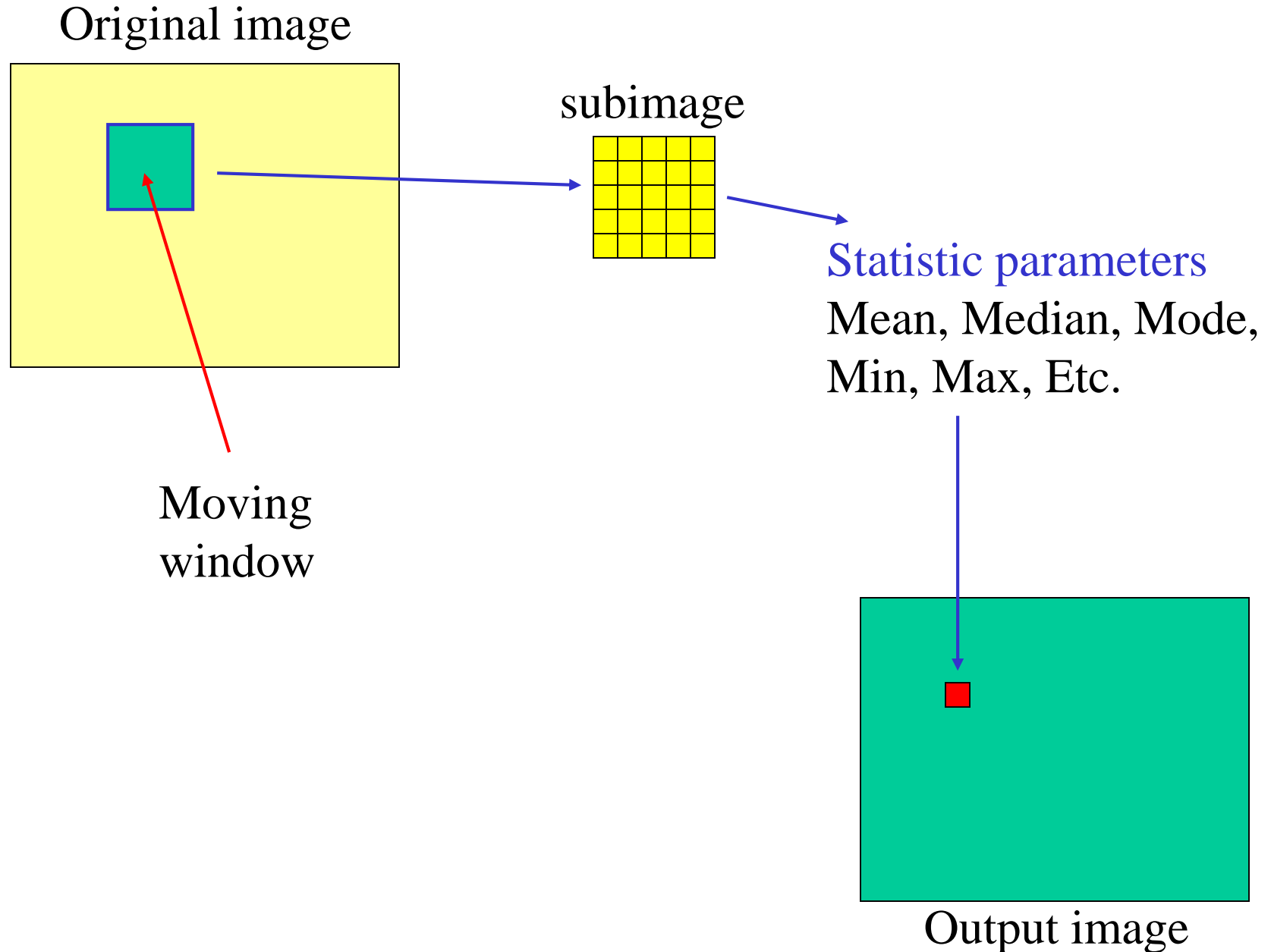


Image obtained using a 3x3 contra-harmonic mean filter With $Q=1.5$



Order-Statistic Filters: Revisit



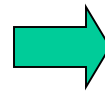
Order-Statistics Filters

Median filter

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

Max filter

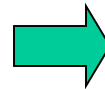
$$\hat{f}(x, y) = \max_{(s,t) \in S_{xy}} \{g(s, t)\}$$



Reduce “dark” noise
(pepper noise)

Min filter

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



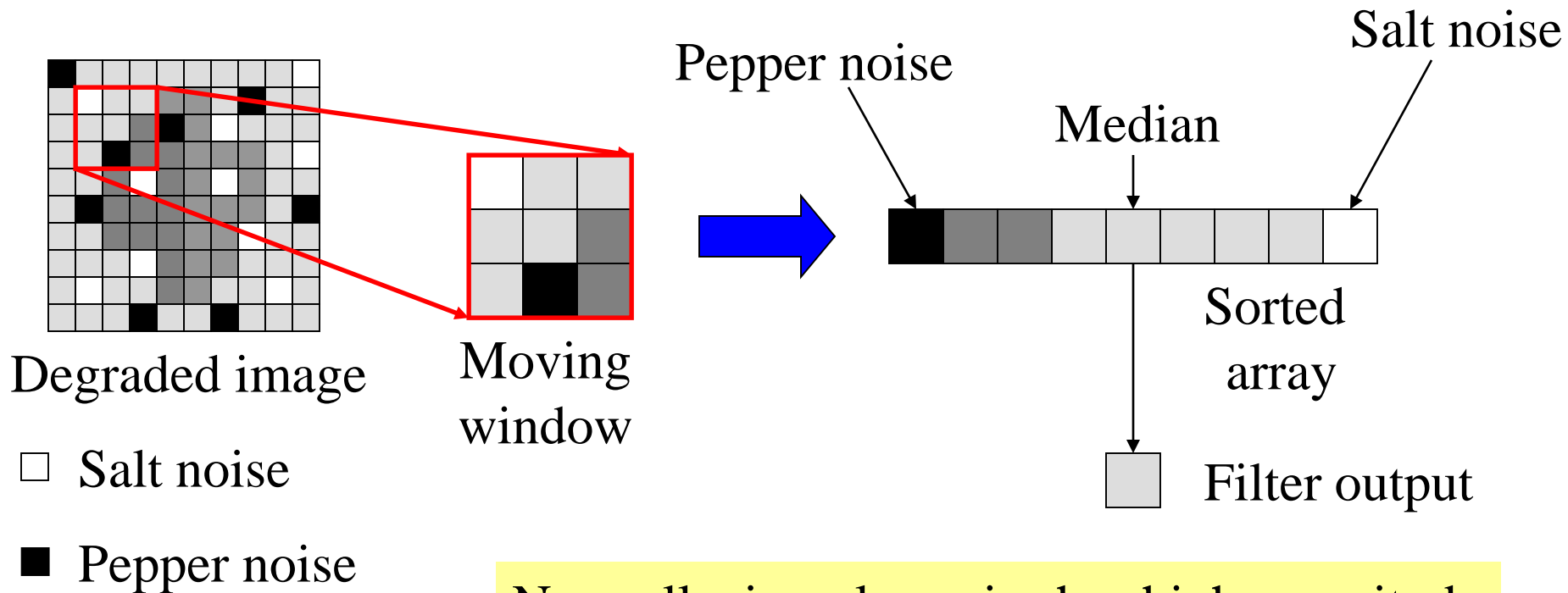
Reduce “bright” noise
(salt noise)

Midpoint filter

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

Median Filter : How it works

A median filter is good for removing impulse, isolated noise

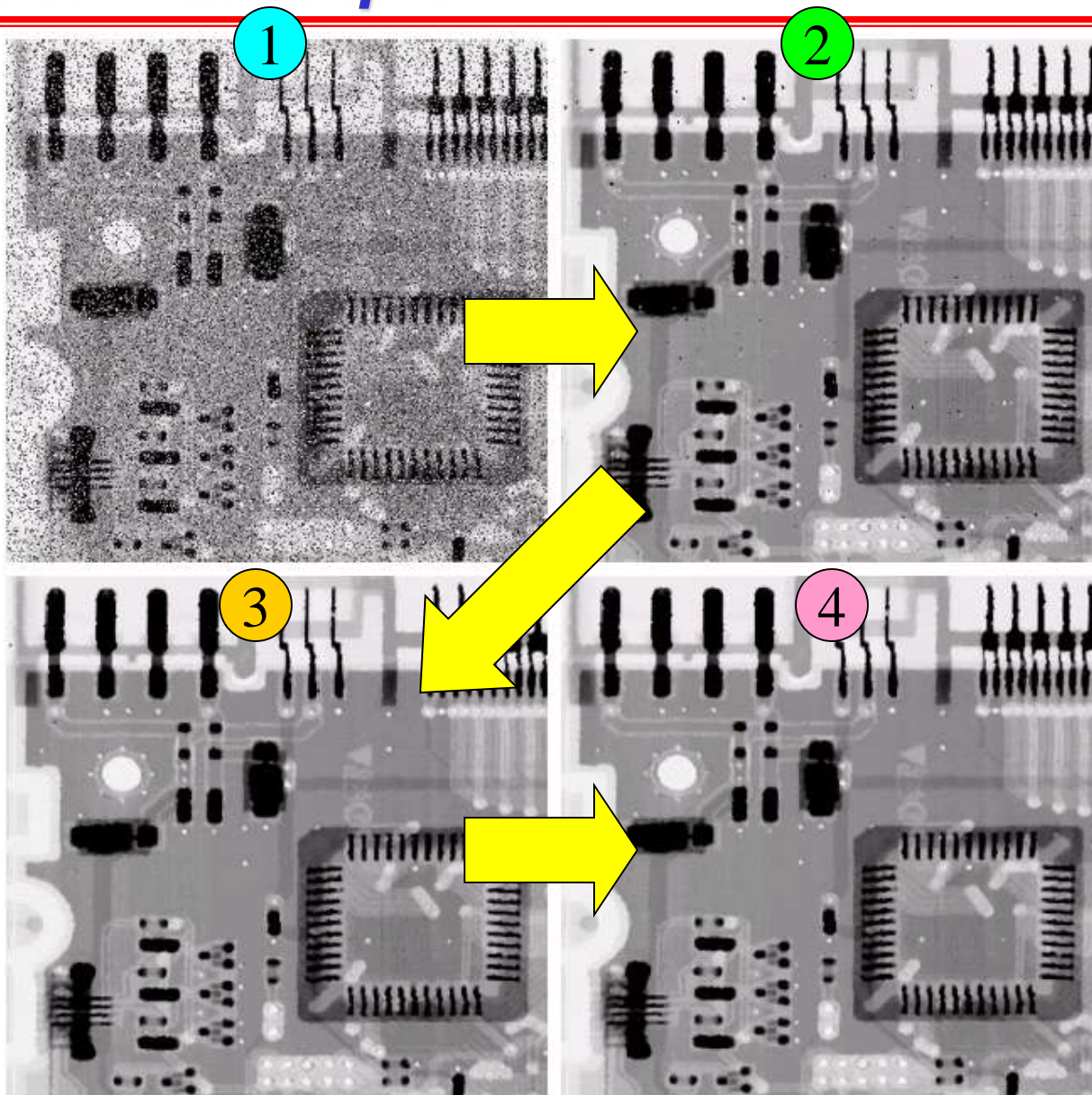


Normally, impulse noise has high magnitude and is isolated. When we sort pixels in the moving window, noise pixels are usually at the ends of the array.

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

Median Filter : Example

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



Images obtained using a 3x3 median filter

(Images from Rafael C. Gonzalez and Richard E. Wood, Digital Image Processing, 2nd Edition.

Max and Min Filters: Example

Image corrupted by pepper noise with prob. = 0.1

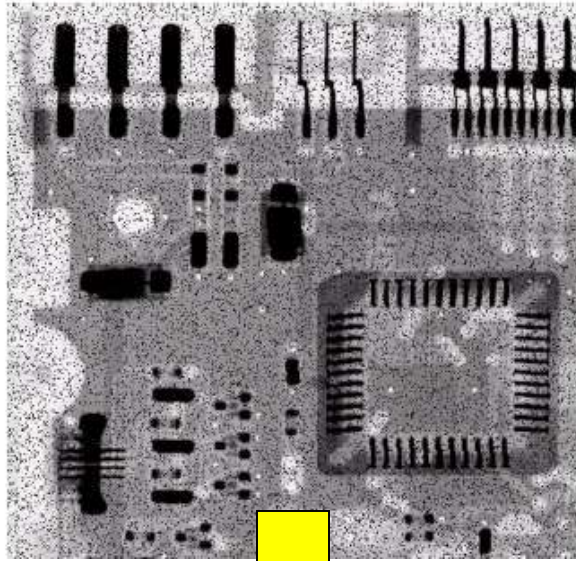


Image obtained using a 3x3 max filter

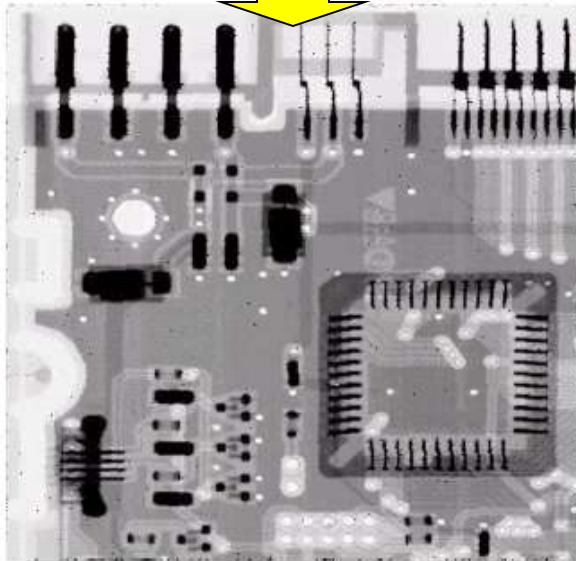


Image corrupted by salt noise with prob. = 0.1

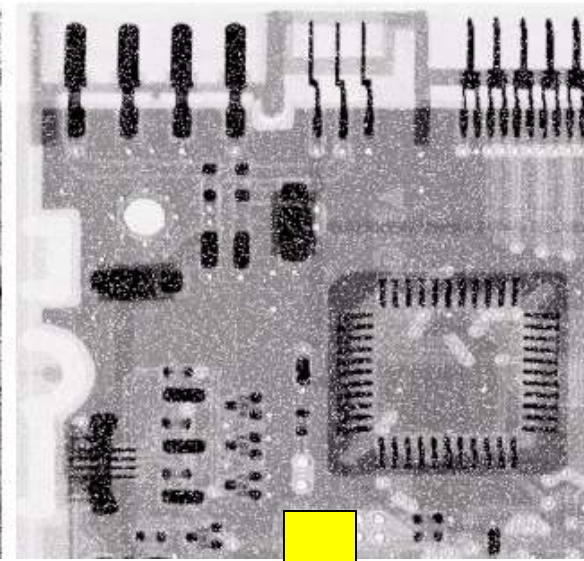
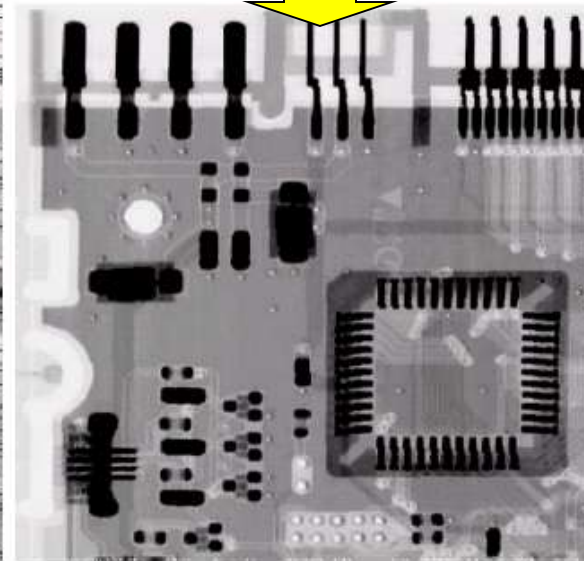


Image obtained using a 3x3 min filter



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: Example

Image corrupted by additive uniform noise

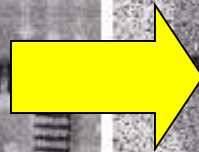
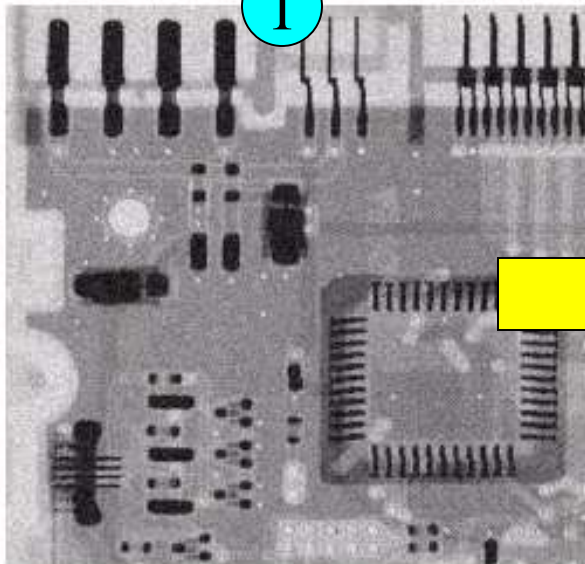


Image additionally corrupted by additive salt-and-pepper noise

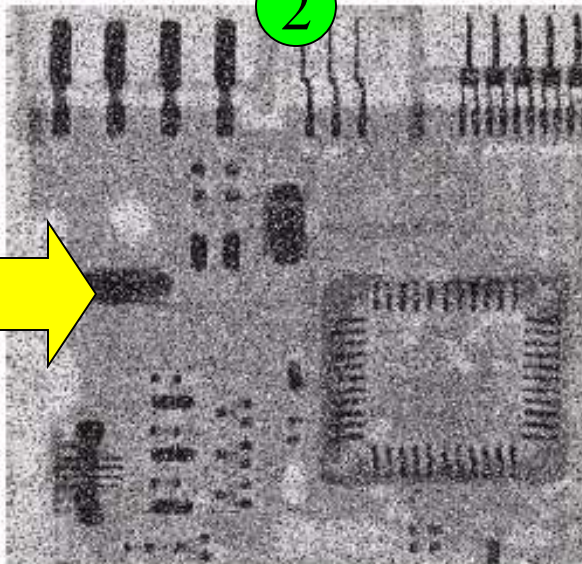


Image 2 obtained using a 5x5 arithmetic mean filter

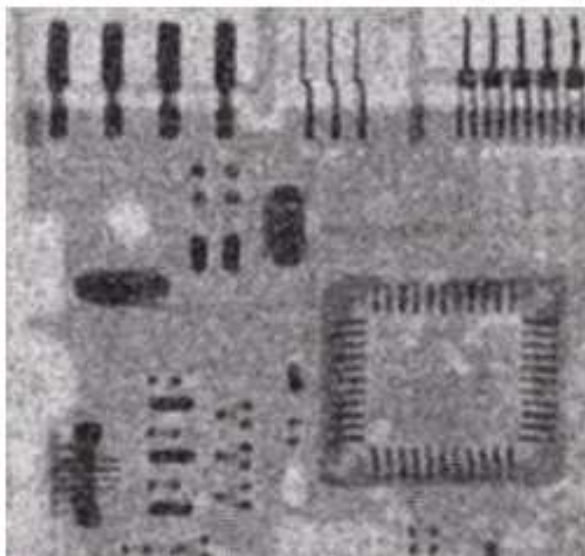
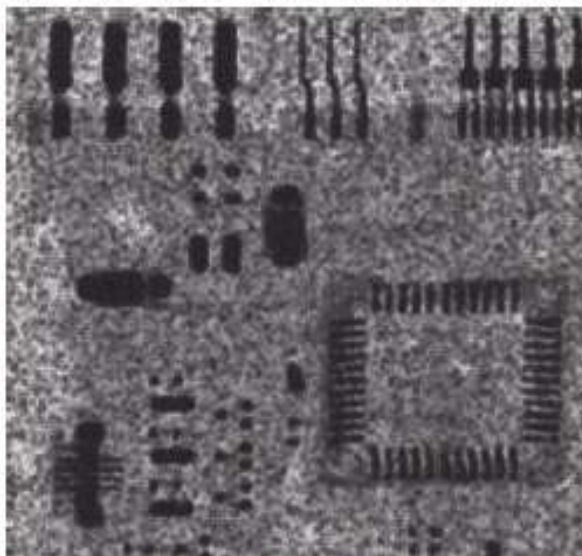


Image 2 obtained using a 5x5 geometric mean filter



Alpha-trimmed Mean Filter: Example (cont.)

Image corrupted by additive uniform noise

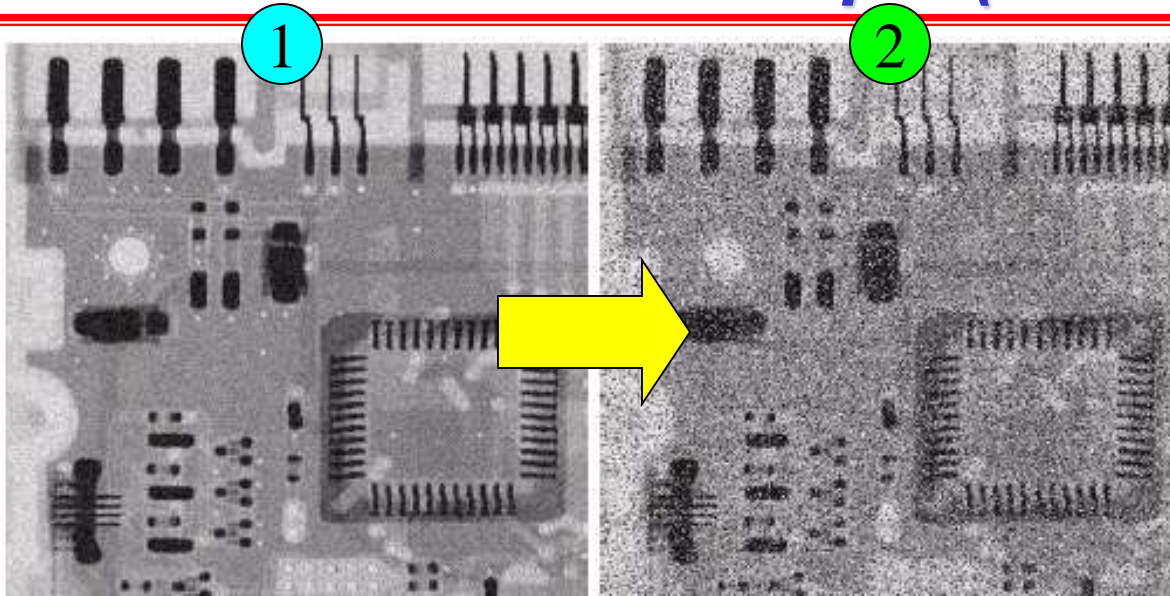


Image additionally corrupted by additive salt-and-pepper noise

Image 2 obtained using a 5x5 median filter

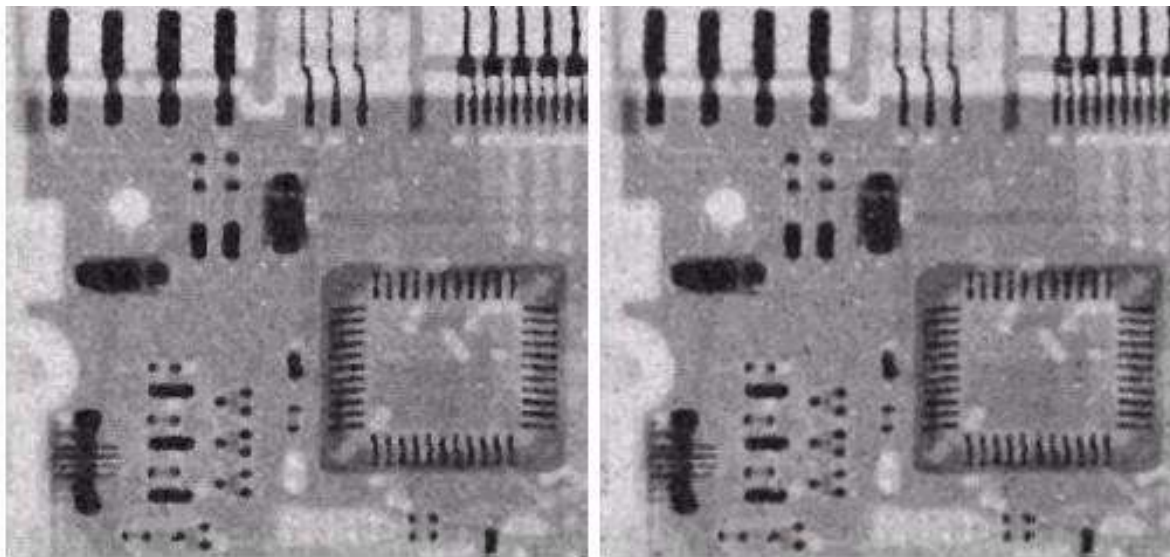


Image 2 obtained using a 5x5 alpha-trimmed mean filter with $d = 5$

Alpha-trimmed Mean Filter: Example (cont.)

Image
obtained
using a **5x5
arithmetic
mean filter**

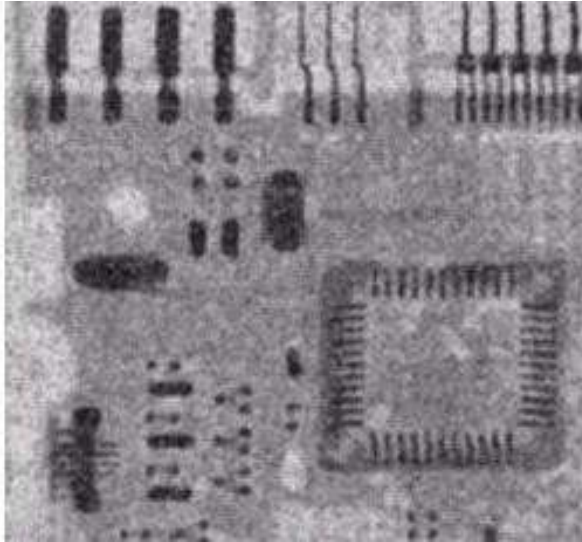


Image
obtained
using a **5x5
geometric
mean filter**

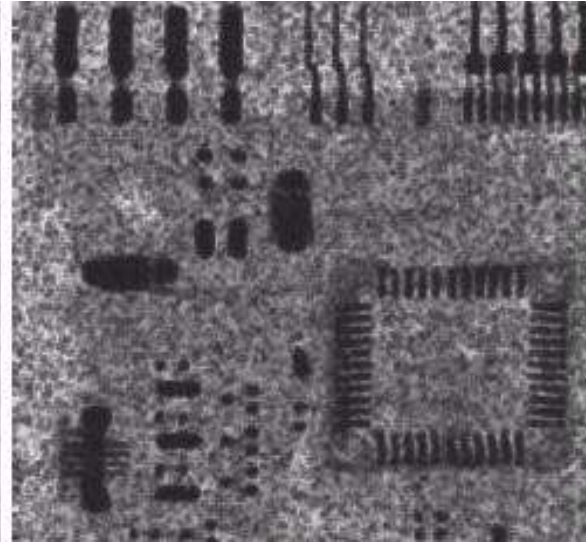


Image
obtained
using a **5x5
median filter**

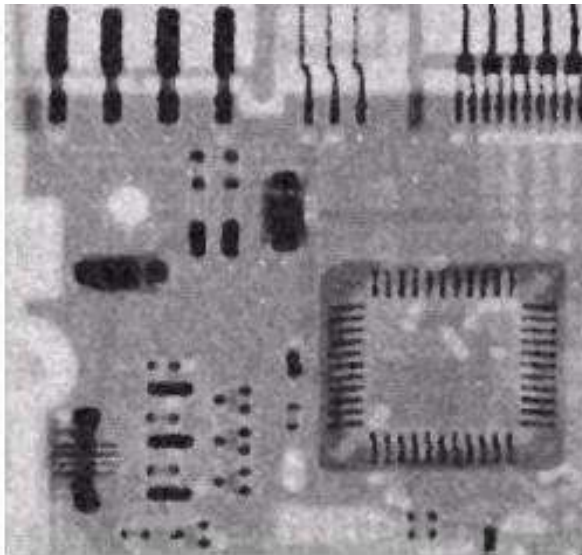
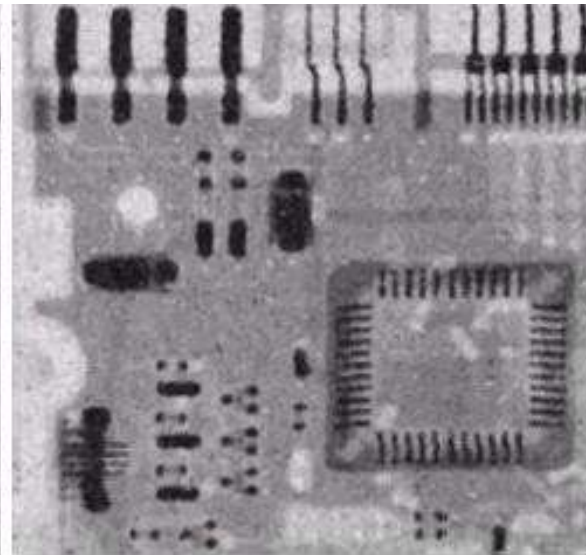


Image
obtained
using a **5x5
alpha-
trimmed
mean filter
with $d = 5$**



Adaptive Filter

General concept:

- Filter behavior depends on statistical characteristics of local areas inside $m \times n$ moving window
- More complex but **superior performance** compared with “fixed” filters

Statistical characteristics:

Local mean:(measure avg. intensity)

$$m_L = \frac{1}{mn} \sum_{(s,t) \in S_{xy}} g(s,t)$$

Noise variance:

$$\sigma_\eta^2$$

Local variance:(measure of contrast in that region)

$$\sigma_L^2 = \frac{1}{mn} \sum_{(s,t) \in S_{xy}} (g(s,t) - m_L)^2$$

Adaptive, Local Noise Reduction Filter

Purpose: want to preserve edges

Concept:

1. If σ_η^2 is zero, \rightarrow No noise
the filter should return $g(x,y)$ because $g(x,y) = f(x,y)$
2. If σ_L^2 is high relative to σ_η^2 , \rightarrow Edges (should be preserved),
the filter should return the value close to $g(x,y)$
3. If $\sigma_L^2 = \sigma_\eta^2$, \rightarrow Areas inside objects
the filter should return the arithmetic mean value m_L

Formula:

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

Adaptive Noise Reduction Filter: Example

Image corrupted by **additive Gaussian noise** with zero mean and $\sigma^2=1000$

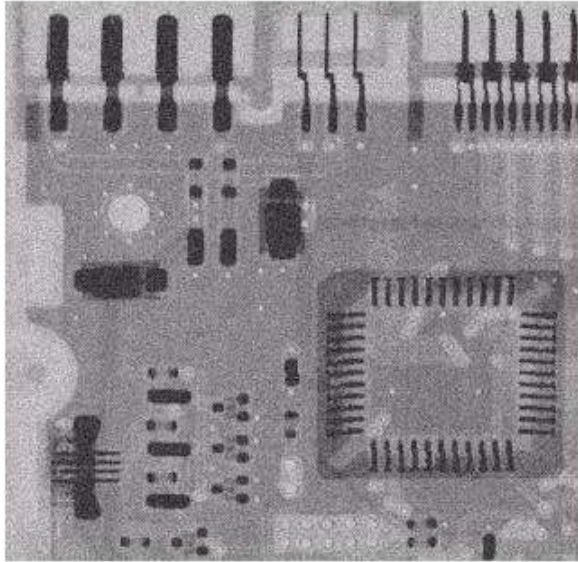


Image obtained using a **7x7 arithmetic mean filter**

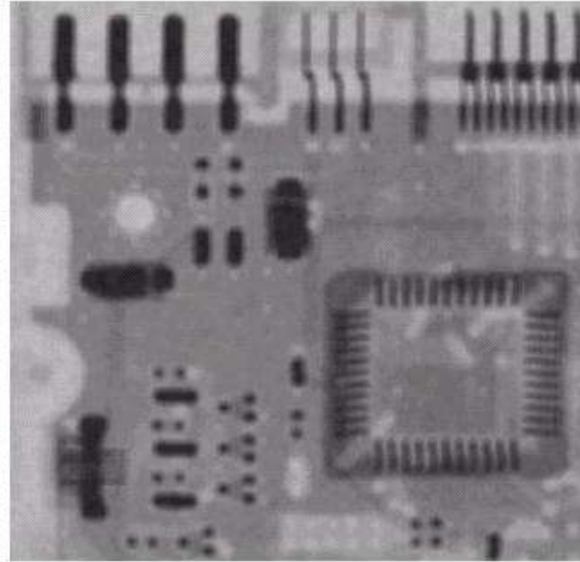


Image obtained using a **7x7 geometric mean filter**

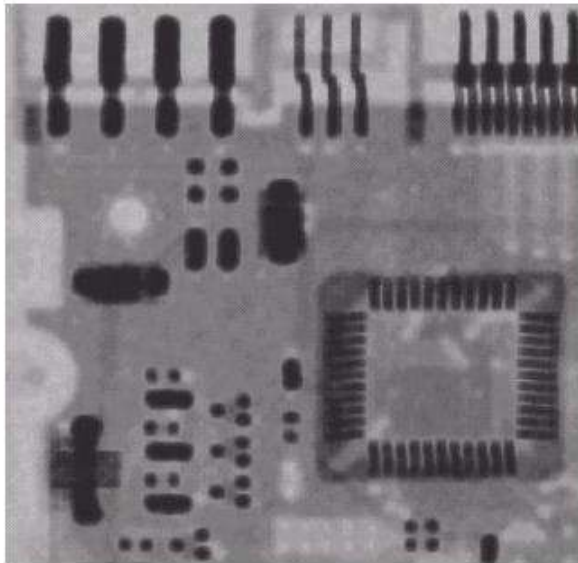
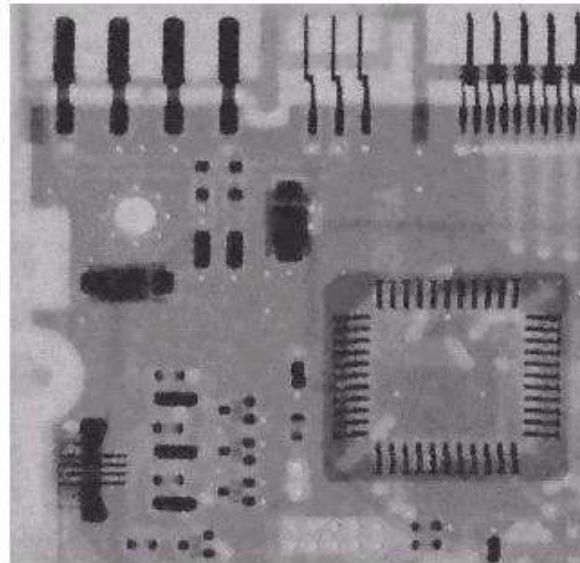


Image obtained using a **7x7 adaptive noise reduction filter**



Adaptive Median Filter

Purpose: want to remove impulse noise while preserving edges

Algorithm: Level A:

- $A1 = z_{\text{median}} - z_{\text{min}}$
- $A2 = z_{\text{median}} - z_{\text{max}}$
- If $A1 > 0$ and $A2 < 0$, goto level B
- Else increase window size
 - If window size $\leq S_{\text{max}}$ repeat level A
 - Else return z_{xy}

Level B:

- $B1 = z_{xy} - z_{\text{min}}$
- $B2 = z_{xy} - z_{\text{max}}$
- If $B1 > 0$ and $B2 < 0$, return z_{xy}
- Else return z_{median}

where

- z_{min} = minimum gray level value in S_{xy}
- z_{max} = maximum gray level value in S_{xy}
- z_{median} = median of gray levels in S_{xy}
- z_{xy} = gray level value at pixel (x,y)
- S_{max} = maximum allowed size of S_{xy}

Adaptive Median Filter: How it works

Level A: $A1 = z_{\text{median}} - z_{\text{min}}$

$A2 = z_{\text{median}} - z_{\text{max}}$

If $A1 > 0$ and $A2 < 0$, goto level B

Else \rightarrow Window is not big enough
increase window size

If window size $\leq S_{\text{max}}$ repeat level A

Else return z_{xy}

Determine
whether z_{median}
is an impulse or not

Level B: $\rightarrow z_{\text{median}}$ is not an impulse

$B1 = z_{\text{xy}} - z_{\text{min}}$

$B2 = z_{\text{xy}} - z_{\text{max}}$

If $B1 > 0$ and $B2 < 0$,

return z_{xy} \rightarrow to preserve original details

Else

return z_{median} \rightarrow to remove impulse

Determine
whether z_{xy}
is an impulse or not
 $\rightarrow z_{\text{xy}}$ is not an impulse

Adaptive Median Filter: Example

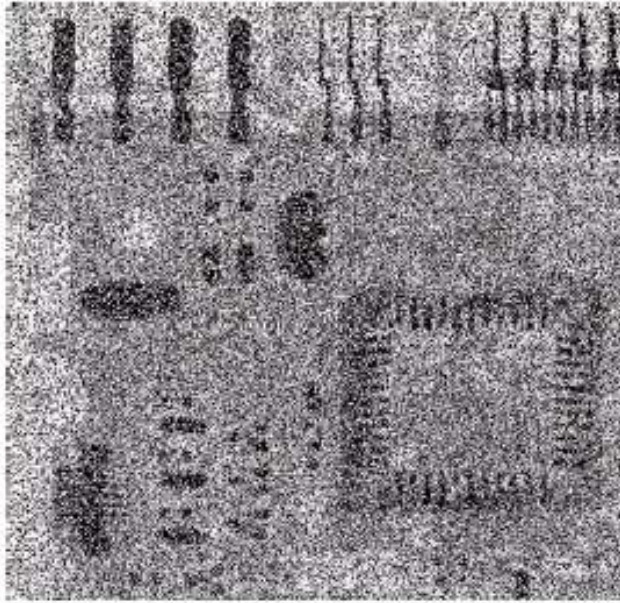


Image corrupted
by salt-and-pepper
noise with
 $p_a = p_b = 0.25$

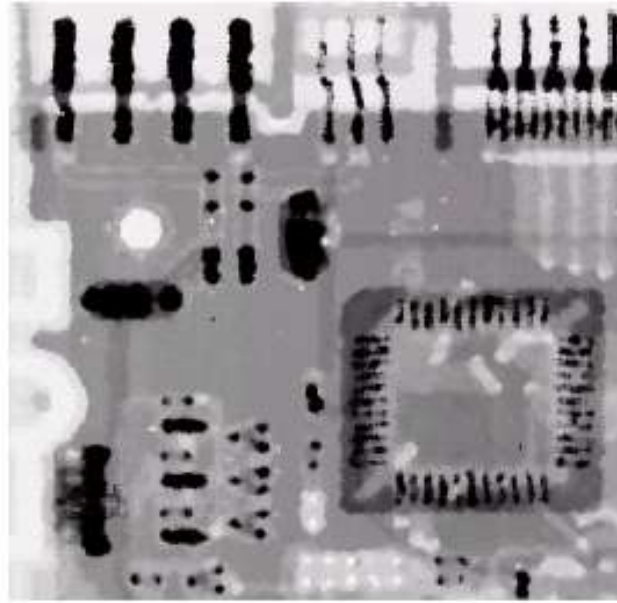


Image obtained
using a **7x7
median filter**

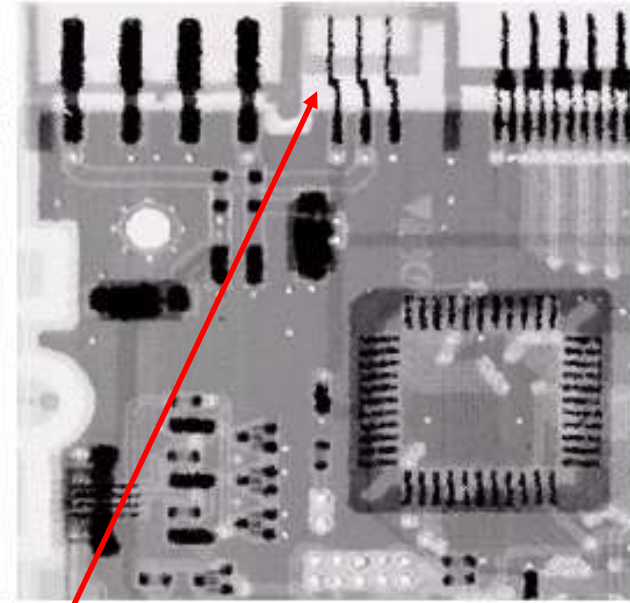


Image obtained
using an **adaptive
median filter** with
 $S_{\max} = 7$

More small details are preserved

Estimation of Degradation Model

Degradation model:

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

or

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

Purpose: to estimate $h(x, y)$ or $H(u, v)$

Why? If we know exactly $h(x, y)$, regardless of noise, we can do deconvolution to get $f(x, y)$ back from $g(x, y)$.

Methods:

1. Estimation by Image Observation
2. Estimation by Experiment
3. Estimation by Modeling

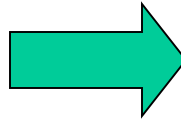
Estimation by Image Observation

Original image (unknown)

$f(x,y)$

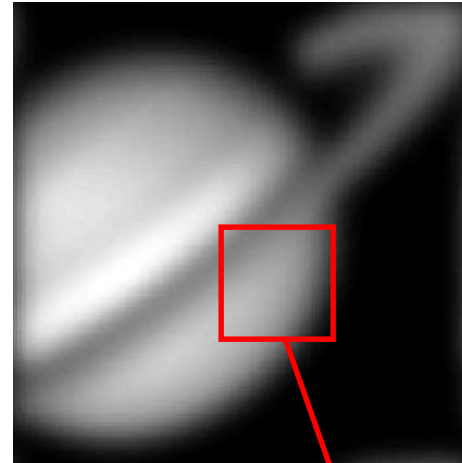


$f(x,y)*h(x,y)$



Degraded image

$g(x,y)$



Observation

Estimated Transfer function

$$H(u,v) \approx H_s(u,v) = \frac{G_s(u,v)}{\hat{F}_s(u,v)}$$

This case is used when we know only $g(x,y)$ and cannot repeat the experiment!

DFT $G_s(u,v)$

DFT $\hat{F}_s(u,v)$

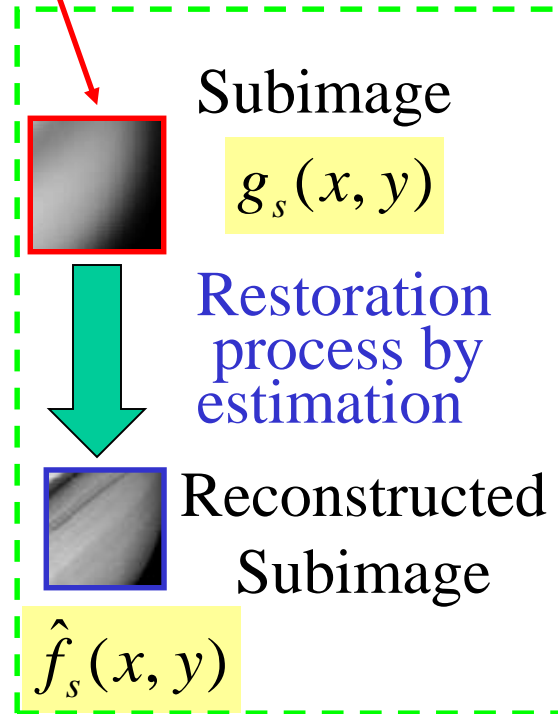
Subimage

$g_s(x,y)$

Restoration process by estimation

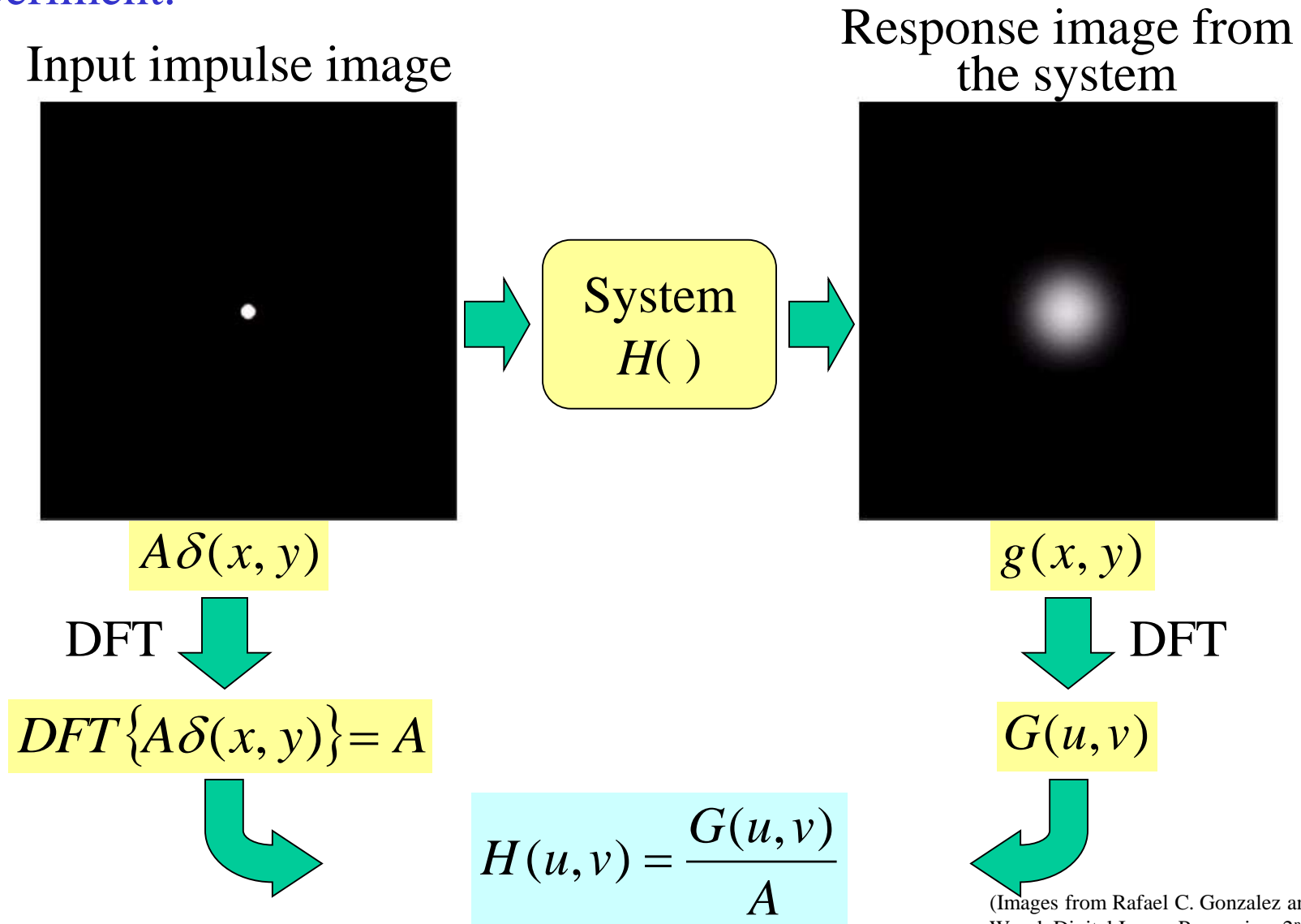
Reconstructed Subimage

$\hat{f}_s(x,y)$



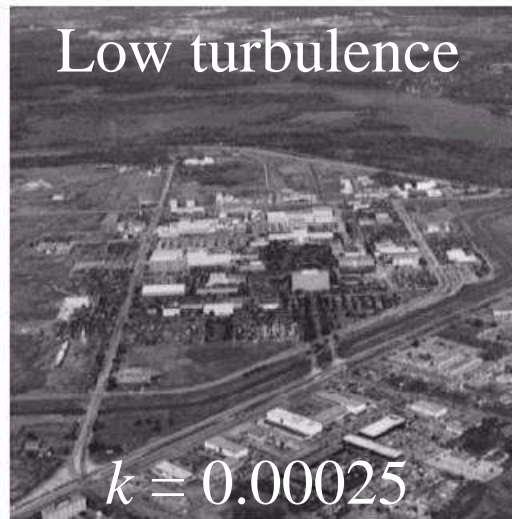
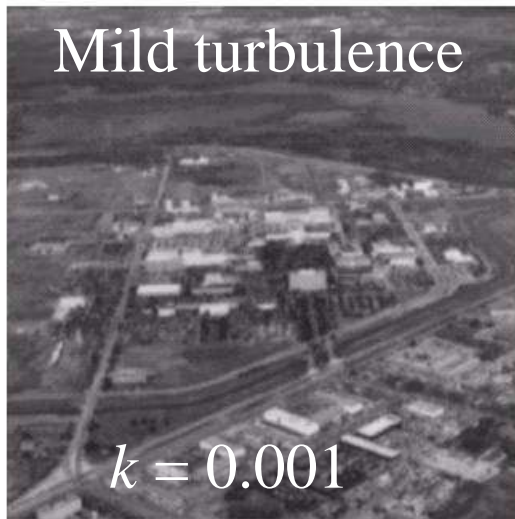
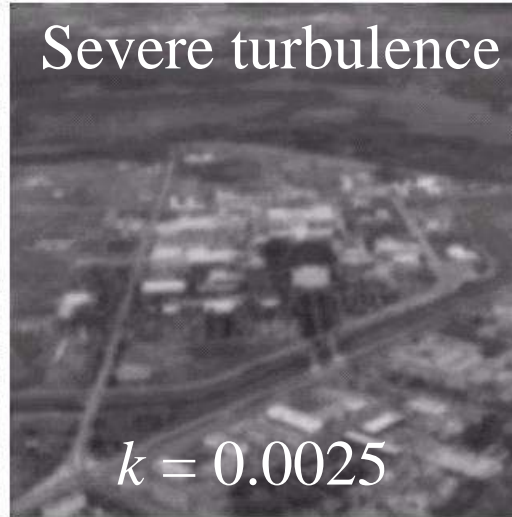
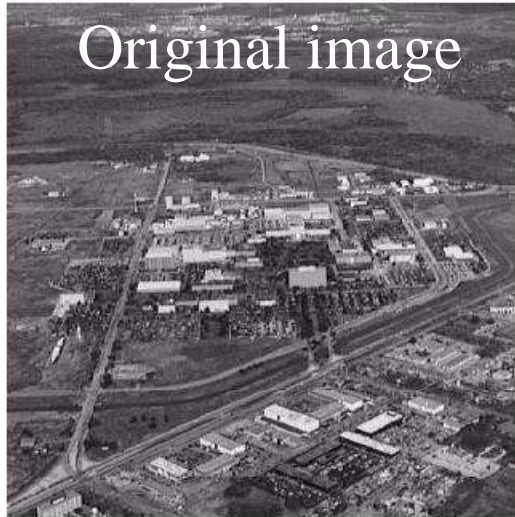
Estimation by Experiment

Used when we have the same equipment set up and can repeat the experiment.



Estimation by Modeling

Used when we know physical mechanism underlying the image formation process that can be expressed mathematically.



Example:

Atmospheric
Turbulence model

$$H(u, v) = e^{-k(u^2 + v^2)^{5/6}}$$

Estimation by Modeling: Motion Blurring

Assume that camera velocity is $(x_0(t), y_0(t))$

The blurred image is obtained by

$$g(x, y) = \int_0^T f(x + x_0(t), y + y_0(t)) dt$$

where T = exposure time.

$$\begin{aligned} G(u, v) &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x, y) e^{-j2\pi(ux+vy)} dx dy \\ &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \left[\int_0^T f(x + x_0(t), y + y_0(t)) dt \right] e^{-j2\pi(ux+vy)} dx dy \\ &= \int_0^T \left[\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x + x_0(t), y + y_0(t)) e^{-j2\pi(ux+vy)} dx dy \right] dt \end{aligned}$$

Estimation by Modeling: Motion Blurring (cont.)

$$\begin{aligned} G(u, v) &= \int_0^T \left[\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x + x_0(t), y + y_0(t)) e^{-j2\pi(ux+vy)} dx dy \right] dt \\ &= \int_0^T \left[F(u, v) e^{-j2\pi(ux_0(t)+vy_0(t))} \right] dt \\ &= F(u, v) \int_0^T e^{-j2\pi(ux_0(t)+vy_0(t))} dt \end{aligned}$$

Then we get, the motion blurring transfer function:

$$H(u, v) = \int_0^T e^{-j2\pi(ux_0(t)+vy_0(t))} dt$$

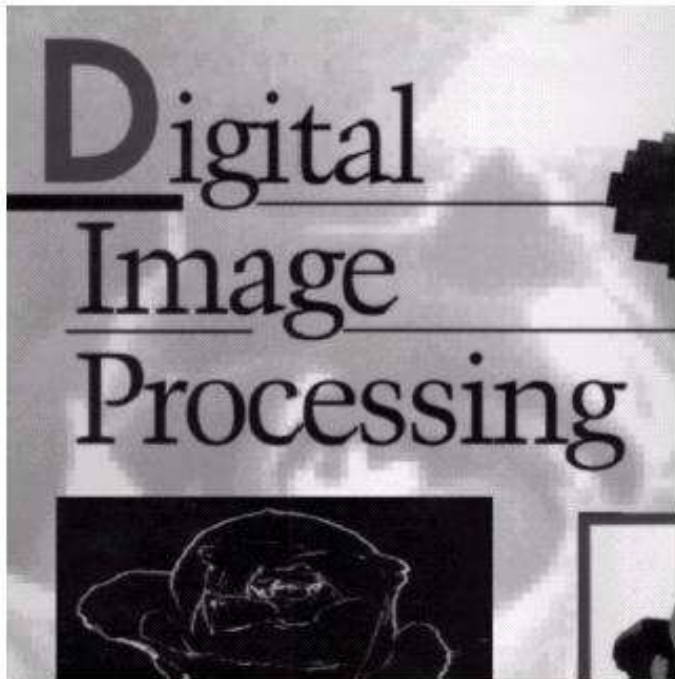
For constant motion $(x_0(t), y_0(t)) = (at, bt)$

$$H(u, v) = \int_0^T e^{-j2\pi(ua+vb)} dt = \frac{T}{\pi(ua+vb)} \sin(\pi(ua+vb)) e^{-j\pi(ua+vb)}$$

Motion Blurring Example

For constant motion

$$H(u, v) = \frac{T}{\pi(ua + vb)} \sin(\pi(ua + vb)) e^{-j\pi(ua + vb)}$$



Original image



Motion blurred image

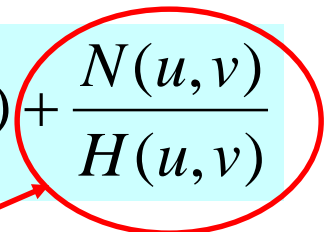
$$a = b = 0.1, T = 1$$

Inverse Filter

From degradation model:

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

after we obtain $H(u, v)$, we can estimate $F(u, v)$ by the inverse filter:

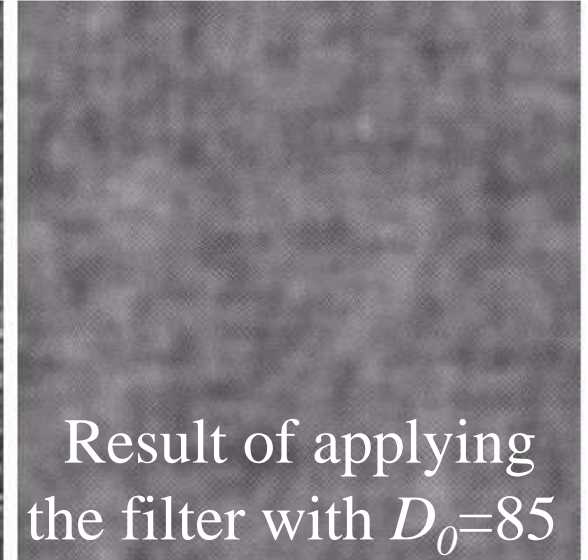
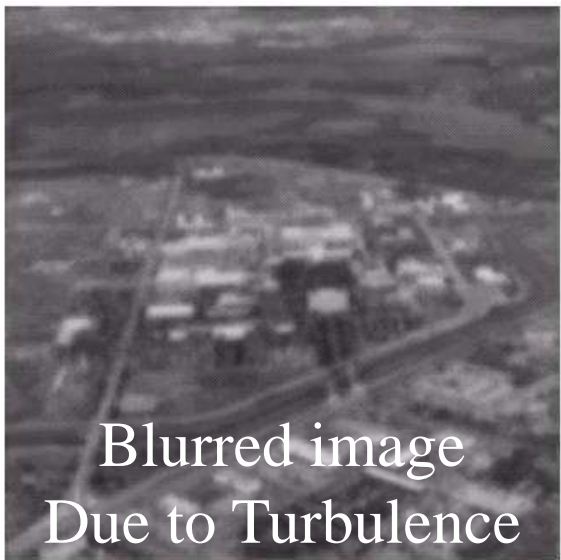
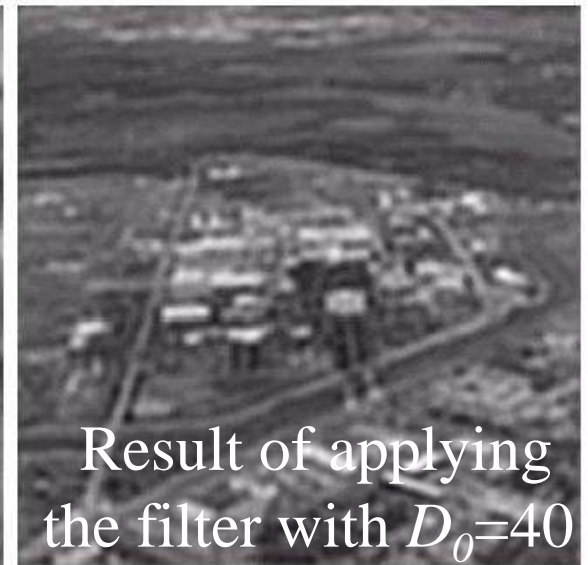
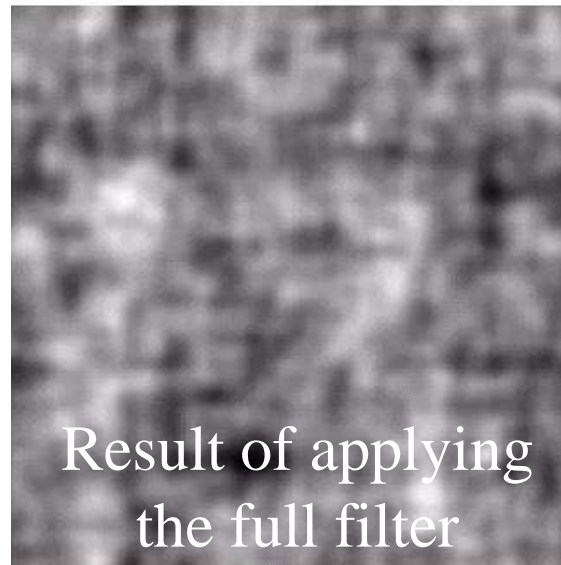
$$\hat{F}(u, v) = \frac{G(u, v)}{H(u, v)} = F(u, v) + \frac{N(u, v)}{H(u, v)}$$


Noise is enhanced
when $H(u, v)$ is small.

To avoid the side effect of enhancing noise, we can apply this formulation to freq. component (u, v) with in a radius D_0 from the center of $H(u, v)$.

In practical, the inverse filter is not
Popularly used.

Inverse Filter: Example



$$H(u, v) = e^{-0.0025(u^2 + v^2)^{5/6}}$$

Wiener Filter: Minimum Mean Square Error Filter

Objective: optimize mean square error: $e^2 = E\{(f - \hat{f})^2\}$

Wiener Filter Formula:

$$\begin{aligned}\hat{F}(u, v) &= \left[\frac{H^*(u, v) S_f(u, v)}{S_f(u, v) |H(u, v)|^2 + S_\eta(u, v)} \right] G(u, v) \\ &= \left[\frac{H^*(u, v)}{|H(u, v)|^2 + S_\eta(u, v) / S_f(u, v)} \right] G(u, v) \\ &= \left[\frac{1}{H(u, v)} \frac{|H(u, v)|^2}{|H(u, v)|^2 + S_\eta(u, v) / S_f(u, v)} \right] G(u, v)\end{aligned}$$

where

$H(u, v)$ = Degradation function

$S_\eta(u, v)$ = Power spectrum of noise

$S_f(u, v)$ = Power spectrum of the undegraded image

Approximation of Wiener Filter

Wiener Filter Formula:

$$\hat{F}(u, v) = \left[\frac{1}{H(u, v)} \frac{|H(u, v)|^2}{|H(u, v)|^2 + S_{\eta}(u, v) / S_f(u, v)} \right] G(u, v)$$

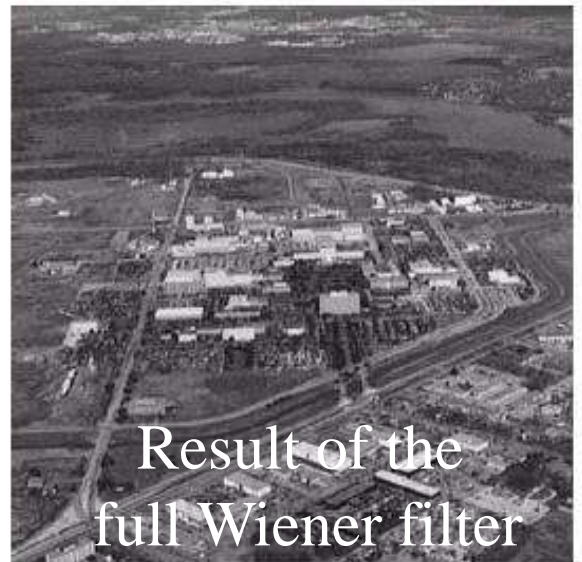
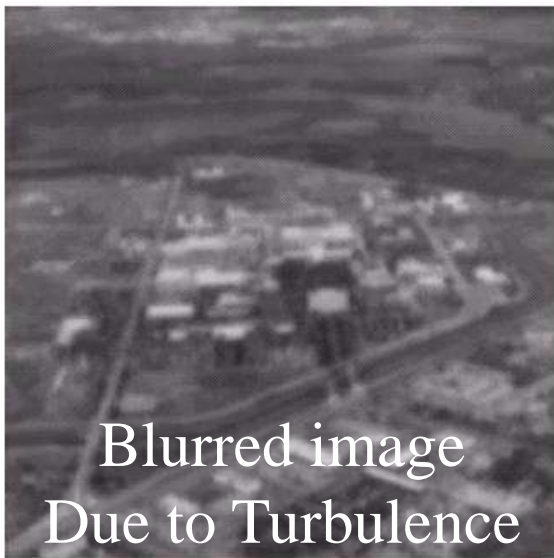
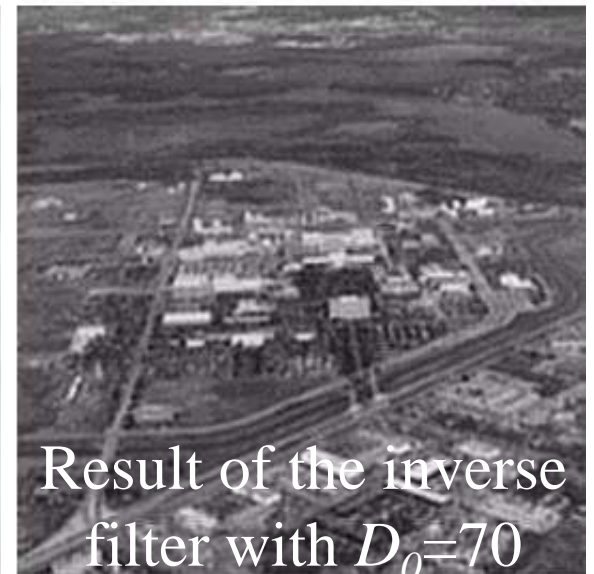
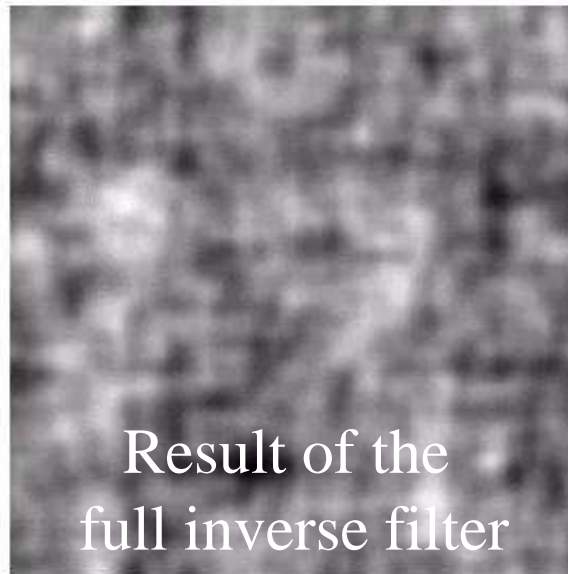
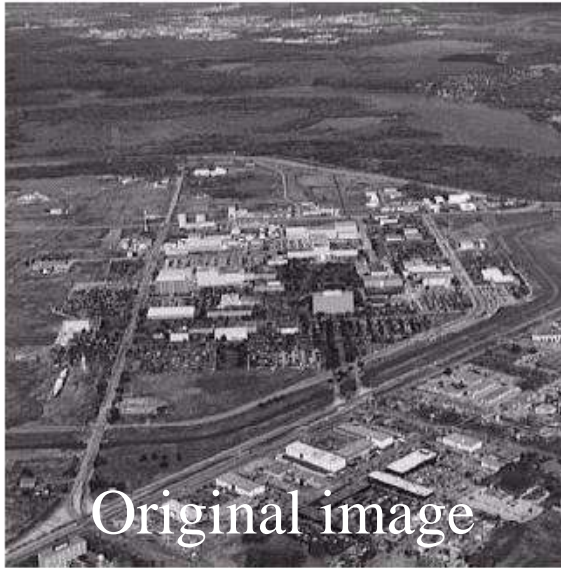
Difficult to estimate

Approximated Formula:

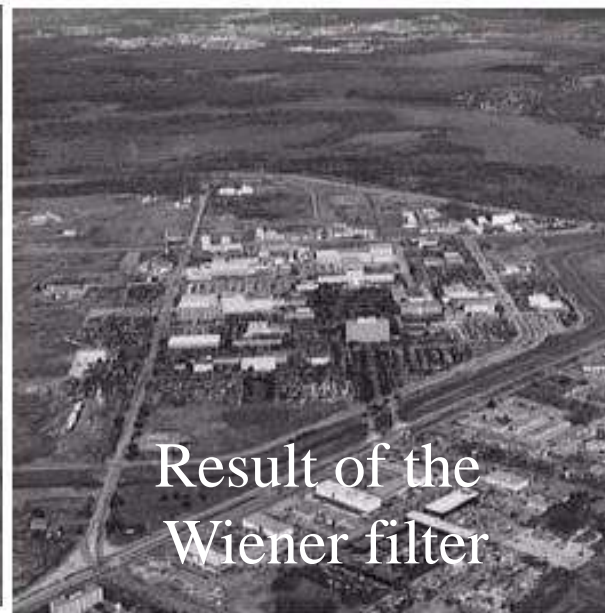
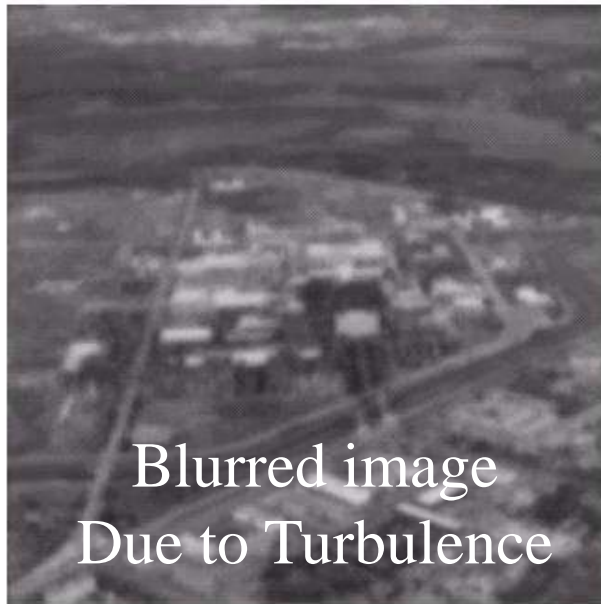
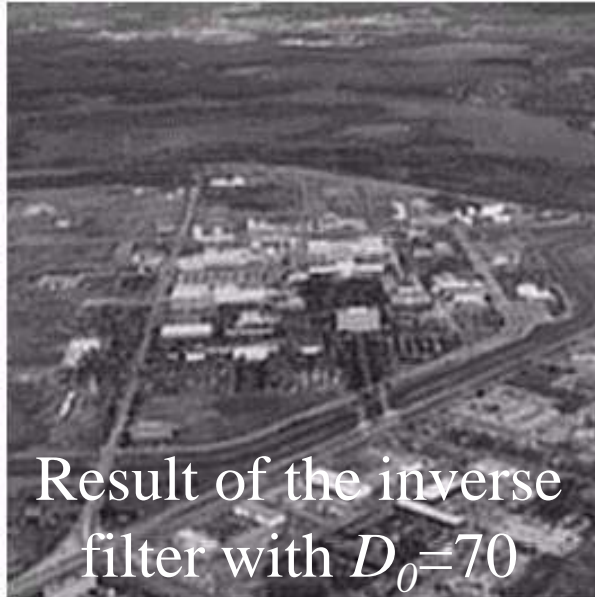
$$\hat{F}(u, v) = \left[\frac{1}{H(u, v)} \frac{|H(u, v)|^2}{|H(u, v)|^2 + K} \right] G(u, v)$$

Practically, K is chosen manually to obtained the best visual result!

Wiener Filter: Example

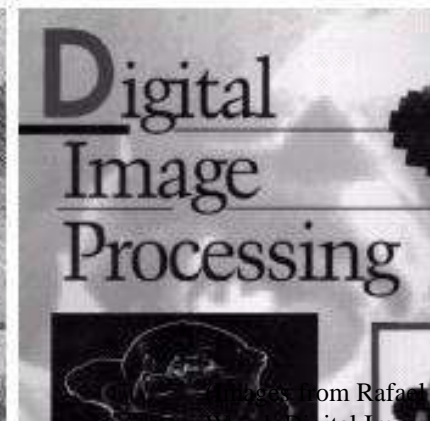
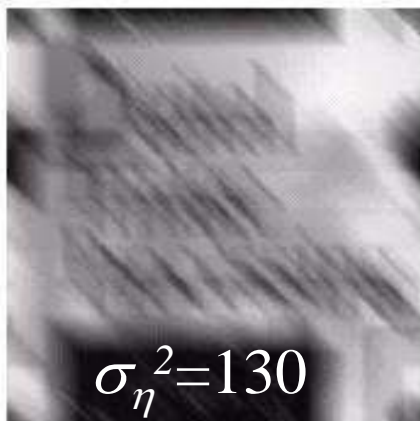
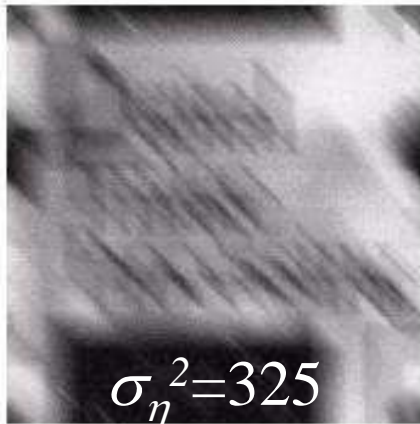
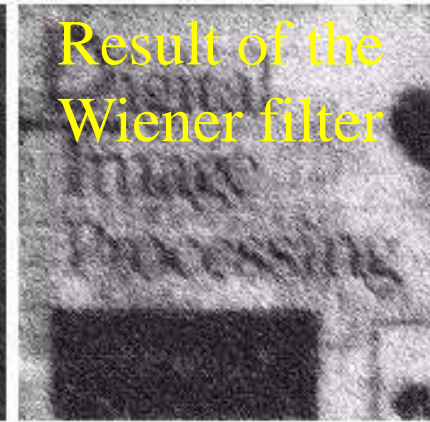
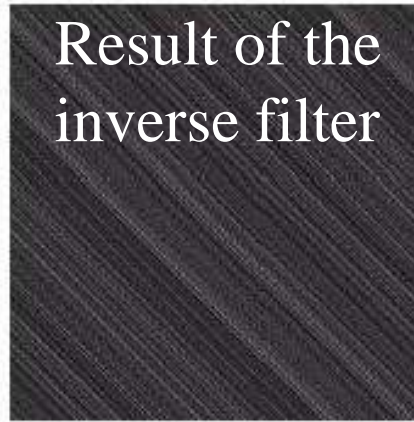
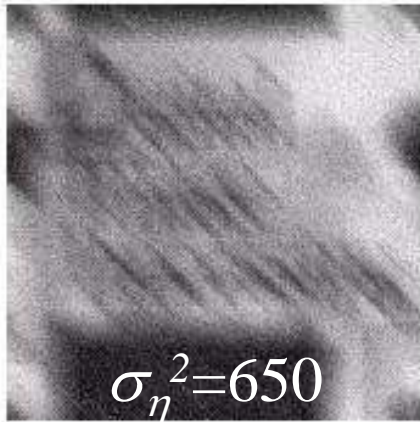


Wiener Filter: Example (cont.)



Example: Wiener Filter and Motion Blurring

Image
degraded
by motion
blur +
AWGN



Note: K is
chosen
manually

Constrained Least Squares Filter

Degradation model:

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

Written in a matrix form

$$\mathbf{g} = \mathbf{H}\mathbf{f} + \boldsymbol{\eta}$$

Objective: to find the minimum of a criterion function

$$C = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [\nabla^2 f(x, y)]^2$$

Subject to the constraint

$$\|\mathbf{g} - \mathbf{H}\hat{\mathbf{f}}\|^2 = \|\boldsymbol{\eta}\|^2$$

where $\|\mathbf{w}\|^2 = \mathbf{w}^T \mathbf{w}$

We get a constrained least square filter

$$\hat{F}(u, v) = \left[\frac{H^*(u, v)}{|H(u, v)|^2 + \gamma |P(u, v)|^2} \right] G(u, v)$$

where

$$P(u, v) = \text{Fourier transform of } p(x, y) = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Constrained Least Squares Filter: Example

Constrained least square filter

$$\hat{F}(u, v) = \left[\frac{H^*(u, v)}{|H(u, v)|^2 + \gamma |P(u, v)|^2} \right] G(u, v)$$

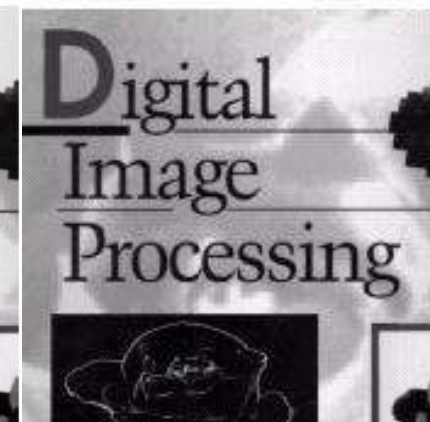
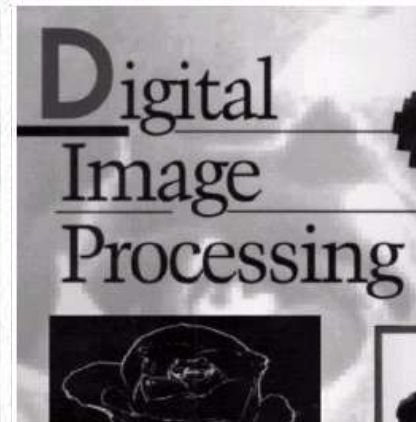
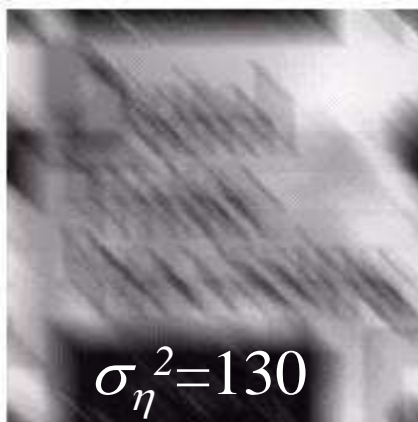
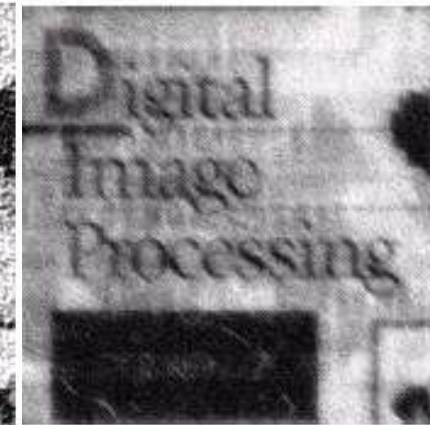
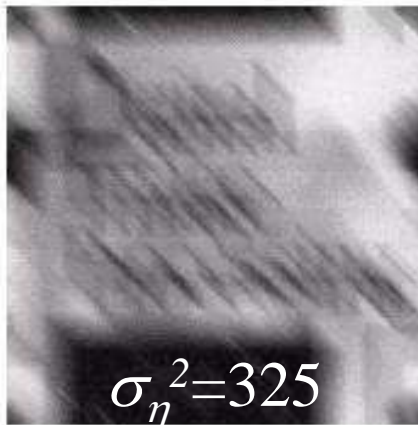
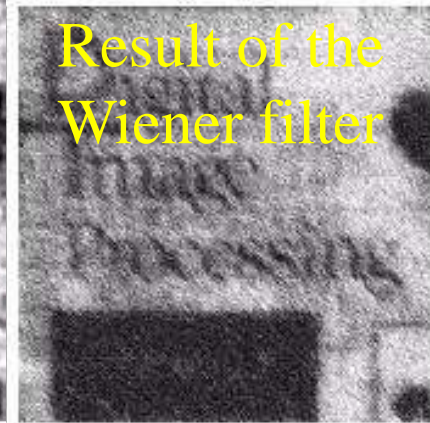
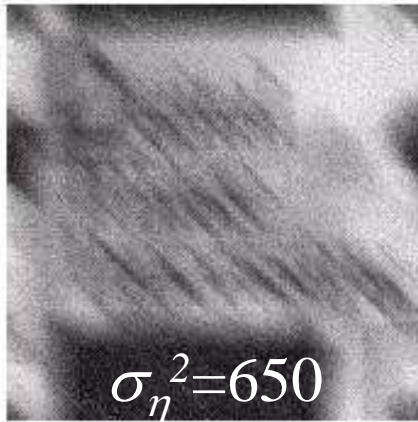
γ is adaptively adjusted to achieve the best result.



Results from the previous slide obtained from the constrained least square filter

Constrained Least Squares Filter: Example (cont.)

Image
degraded
by motion
blur +
AWGN



Constrained Least Squares Filter: Adjusting γ

Define $\mathbf{r} = \mathbf{g} - \mathbf{H}\hat{\mathbf{f}}$ It can be shown that $\phi(\gamma) = \mathbf{r}^T \mathbf{r} = \|\mathbf{r}\|^2$

We want to adjust gamma so that $\|\mathbf{r}\|^2 = \|\boldsymbol{\eta}\|^2 \pm a \rightarrow \textcircled{1}$

1. Specify an initial value of γ

where a = accuracy factor

2. Compute $\|\mathbf{r}\|^2$

3. Stop if $\textcircled{1}$ is satisfied

Otherwise return step 2 after increasing γ if $\|\mathbf{r}\|^2 < \|\boldsymbol{\eta}\|^2 - a$

or decreasing γ if $\|\mathbf{r}\|^2 > \|\boldsymbol{\eta}\|^2 + a$

Use the new value of γ to recompute

$$\hat{F}(u, v) = \left[\frac{H^*(u, v)}{|H(u, v)|^2 + \gamma |P(u, v)|^2} \right] G(u, v)$$

Constrained Least Squares Filter: Adjusting γ (cont.)

$$\hat{F}(u, v) = \left[\frac{H^*(u, v)}{|H(u, v)|^2 + \gamma |P(u, v)|^2} \right] G(u, v)$$

$$R(u, v) = G(u, v) - H(u, v) \hat{F}(u, v)$$

$$\|\mathbf{r}\|^2 = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} r^2(x, y)$$

For computing $\|\mathbf{r}\|^2$

$$m_\eta = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} \eta(x, y)$$

$$\sigma_\eta^2 = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [\eta(x, y) - m_\eta]^2$$

$$\|\boldsymbol{\eta}\|^2 = MN [\sigma_\eta^2 - m_\eta]$$

For computing $\|\boldsymbol{\eta}\|^2$

Constrained Least Squares Filter: Example

Original image

Use correct noise parameters

Correct parameters:

Initial $\gamma = 10^{-5}$

Correction factor = 10^{-6}

$a = 0.25$

$\sigma_{\eta}^2 = 10^{-5}$

Blurred image
Due to Turbulence

Use wrong noise parameters

Wrong noise parameter

$\sigma_{\eta}^2 = 10^{-2}$

(Images from Rafael C.
Gonzalez and Richard E.
Wood, Digital Image
Processing, 2nd Edition.

Results obtained from constrained least square filters

Geometric Mean filter

This filter represents a family of filters combined into a single expression

$$\hat{F}(u, v) = \left[\frac{H^*(u, v)}{|H(u, v)|^2} \right]^\alpha \left[\frac{H^*(u, v)}{|H(u, v)|^2 + \beta \left[\frac{S_\eta(u, v)}{S_f(u, v)} \right]} \right]^{1-\alpha} G(u, v)$$

$\alpha = 1 \rightarrow$ the inverse filter

$\alpha = 0 \rightarrow$ the Parametric Wiener filter

$\alpha = 0, \beta = 1 \rightarrow$ the standard Wiener filter

$\beta = 1, \alpha < 0.5 \rightarrow$ More like the inverse filter

$\beta = 1, \alpha > 0.5 \rightarrow$ More like the Wiener filter

Another name: the spectrum equalization filter

Geometric Transformation

These transformations are often called **rubber-sheet transformations**:
Printing an image on a rubber sheet and then stretch this sheet according to some predefined set of rules.

A geometric transformation consists of 2 basic operations:

- 1. A spatial transformation :**

Define how pixels are to be rearranged in the spatially transformed image.

- 2. Gray level interpolation :**

Assign gray level values to pixels in the spatially transformed image.

Geometric Transformation : Algorithm

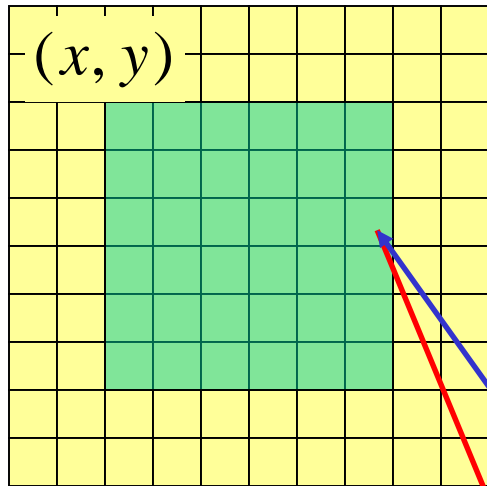
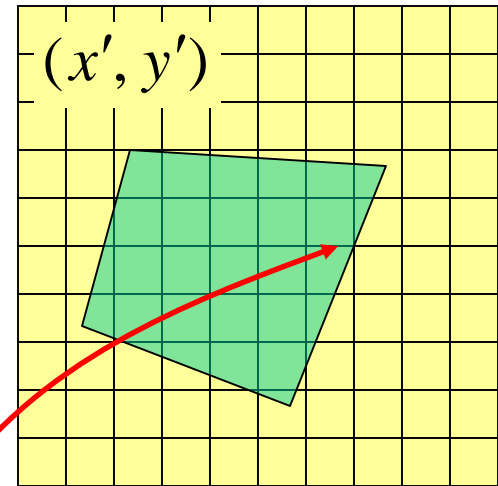


Image f to be restored



Distorted image g

1. Select coordinate (x, y) in f to be restored

2. Compute

$$x' = r(x, y)$$

$$y' = s(x, y)$$

3. Go to pixel (x', y') in a distorted image g

4. get pixel value at $g(x', y')$
By gray level interpolation

5. store that value in pixel $f(x, y)$

Spatial Transformation

To map between pixel coordinate (x, y) of f and pixel coordinate (x', y') of g

$$x' = r(x, y)$$

$$y' = s(x, y)$$

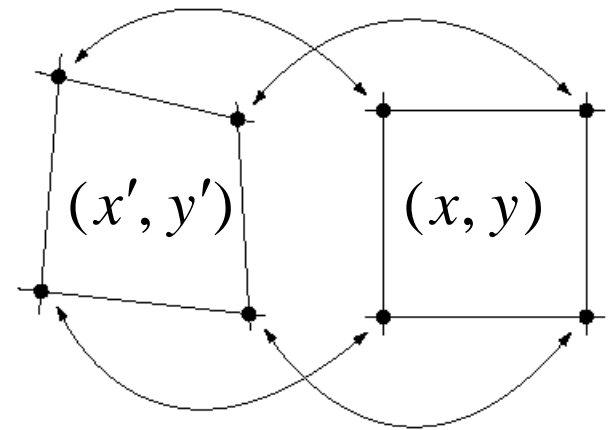
For a bilinear transformation mapping between a pair of Quadrilateral regions

$$x' = r(x, y) = c_1x + c_2y + c_3xy + c_4$$

$$y' = s(x, y) = c_5x + c_6y + c_7xy + c_8$$

To obtain $r(x, y)$ and $s(x, y)$, we need to know 4 pairs of coordinates

(x, y) and its corresponding (x', y') which are called **tiepoints**.

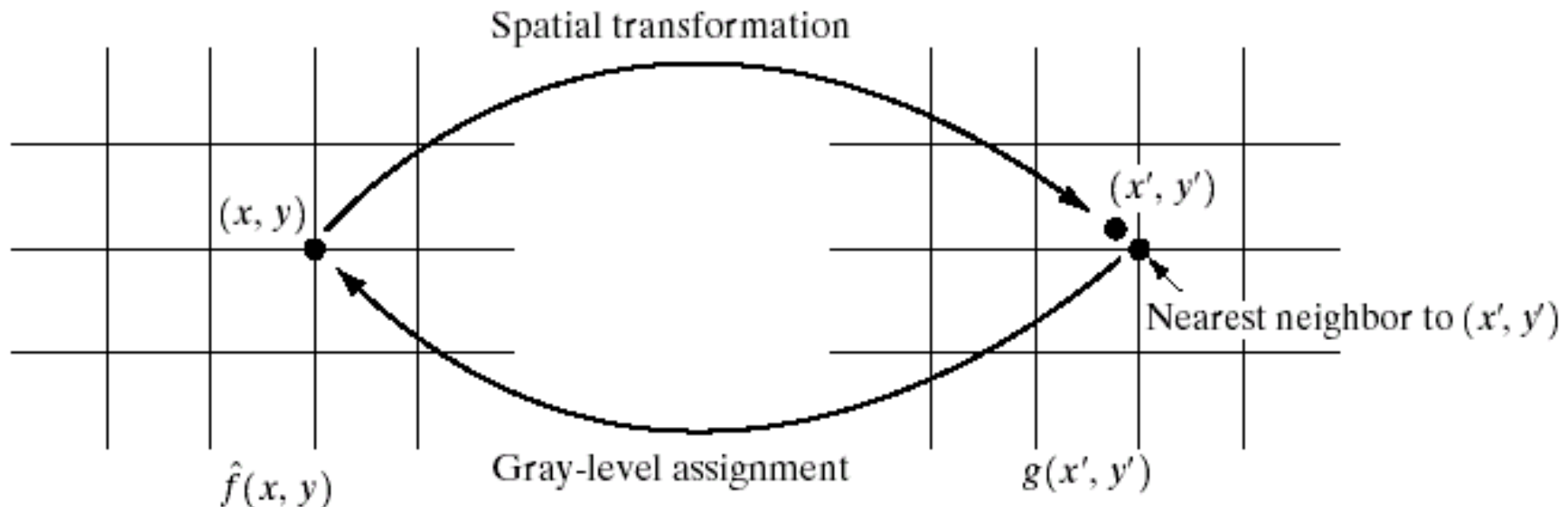


Gray Level Interpolation: Nearest Neighbor

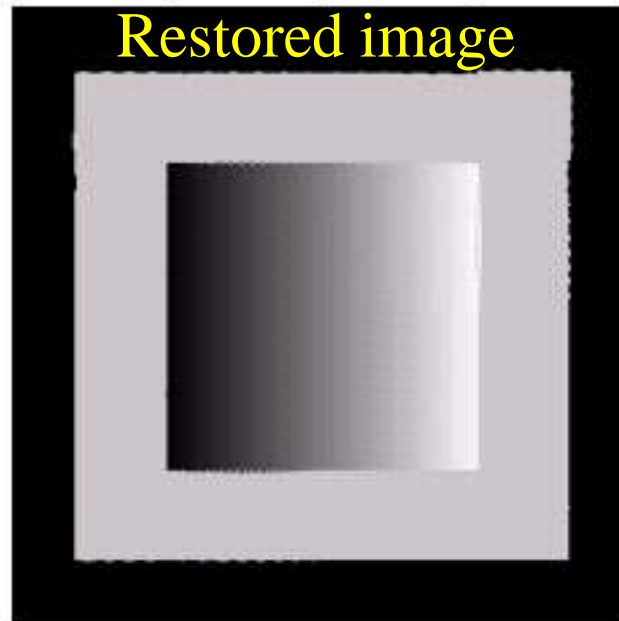
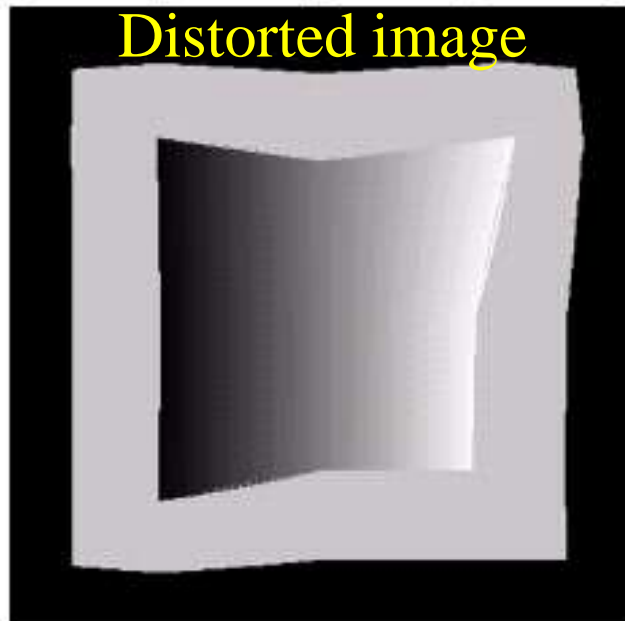
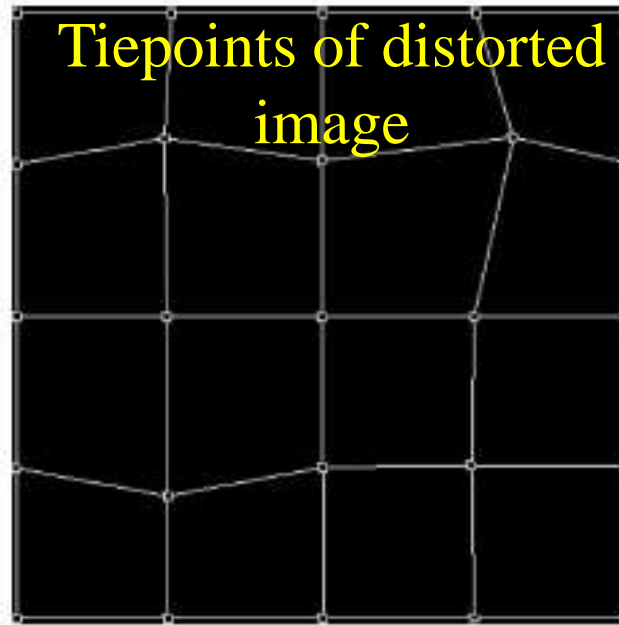
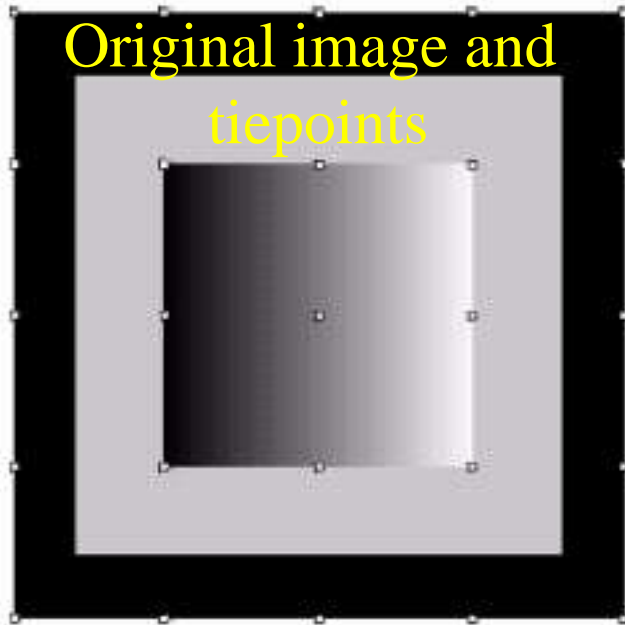
Since (x', y') may not be at an integer coordinate, we need to Interpolate the value of $g(x', y')$

Example interpolation methods that can be used:

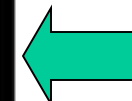
1. Nearest neighbor selection
2. Bilinear interpolation
3. Bicubic interpolation



Geometric Distortion and Restoration Example

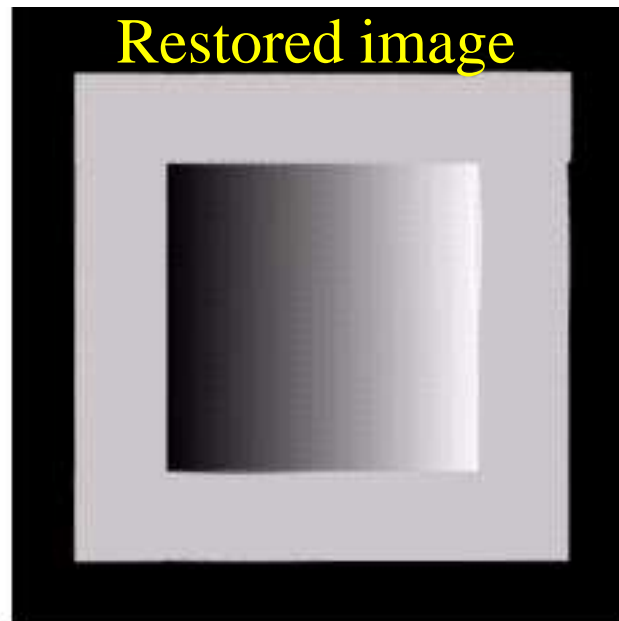
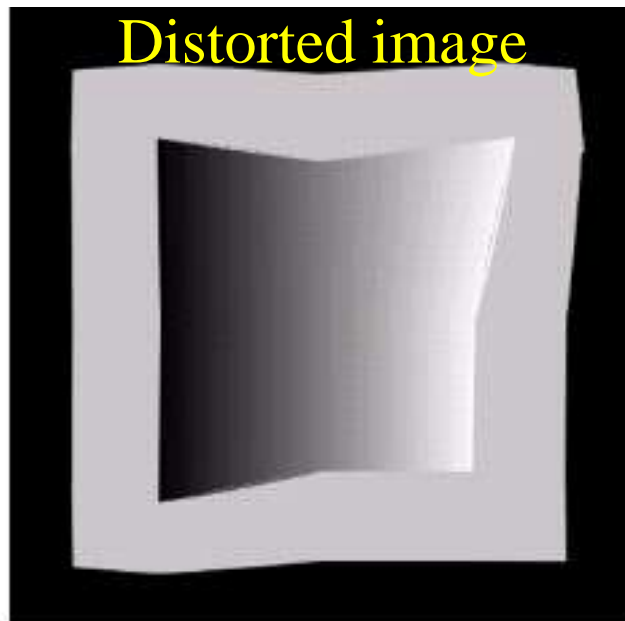
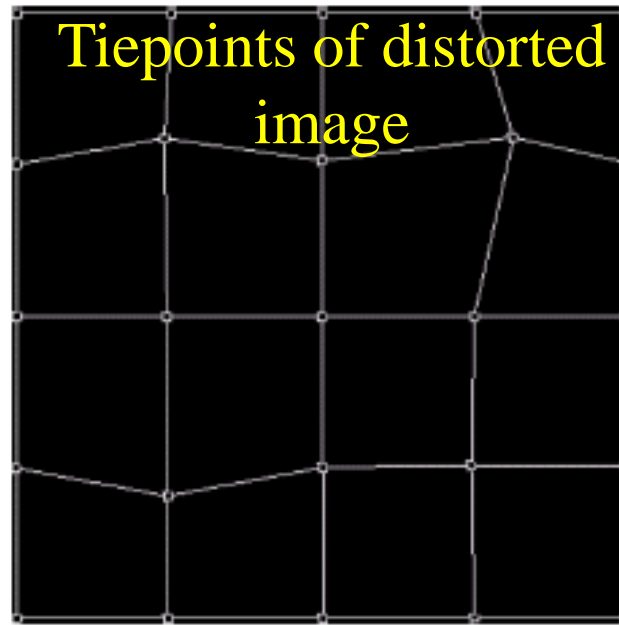
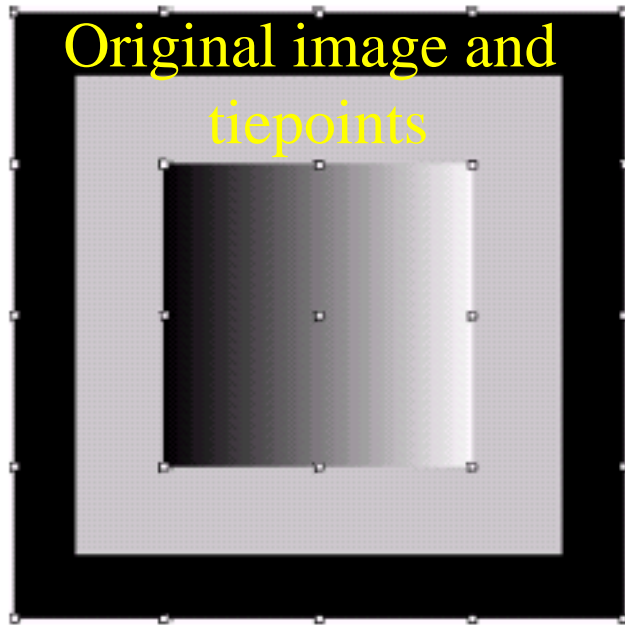


Use nearest neighbor interpolation



(Images from Rafael C. Gonzalez and Richard E. Wood, Digital Image Processing, 2nd Edition.

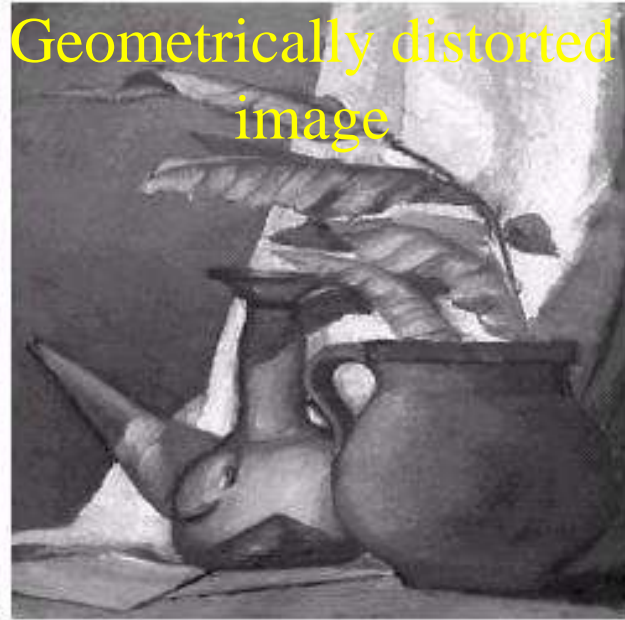
Geometric Distortion and Restoration Example (cont.)



← Use bilinear interpolation

(Images from Rafael C. Gonzalez and Richard E. Wood, Digital Image Processing, 2nd Edition.)

Example: Geometric Restoration



← Use the same Spatial Trans. as in the previous example

