Image Restoration

李东晓

lidx@zju.edu.cn

浙江大学 信息与电子工程学院
College of Information Science & Electronic Engineering, Zhejiang University

Contents

- 5.1 A Model of the Image Degradation / Restoration Process
- 5.5 Linear, Position-Invariant Degradations
- 5.2 Noise Models
- 5.3 Restoration in the Presence of Noise Only—Spatial Filtering
- 5.4 Periodic Noise Reduction by Frequency Domain Filtering

What is Image Restoration?

Image restoration attempts to restore images that have been degraded

- Identify the degradation process and attempt to reverse it
- Similar to image enhancement, but more objective

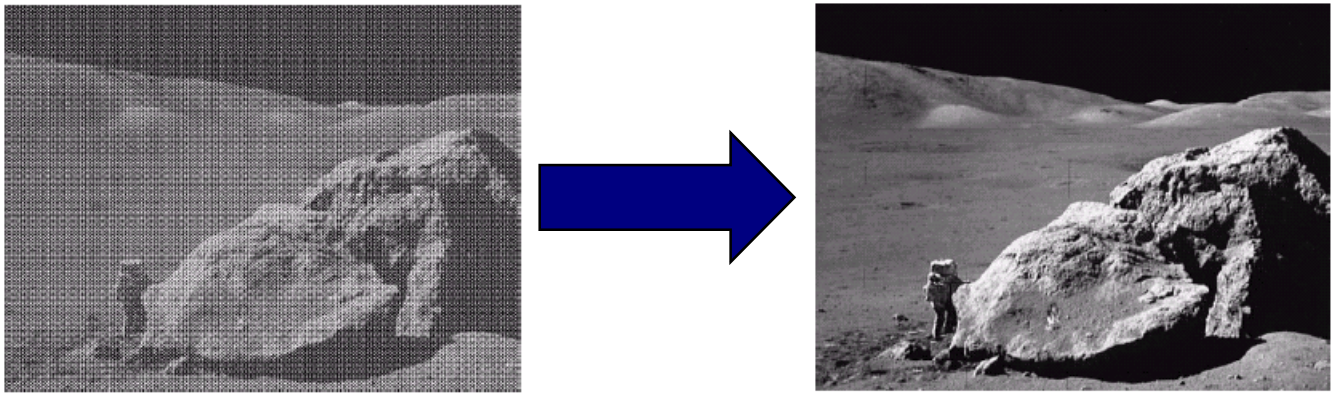
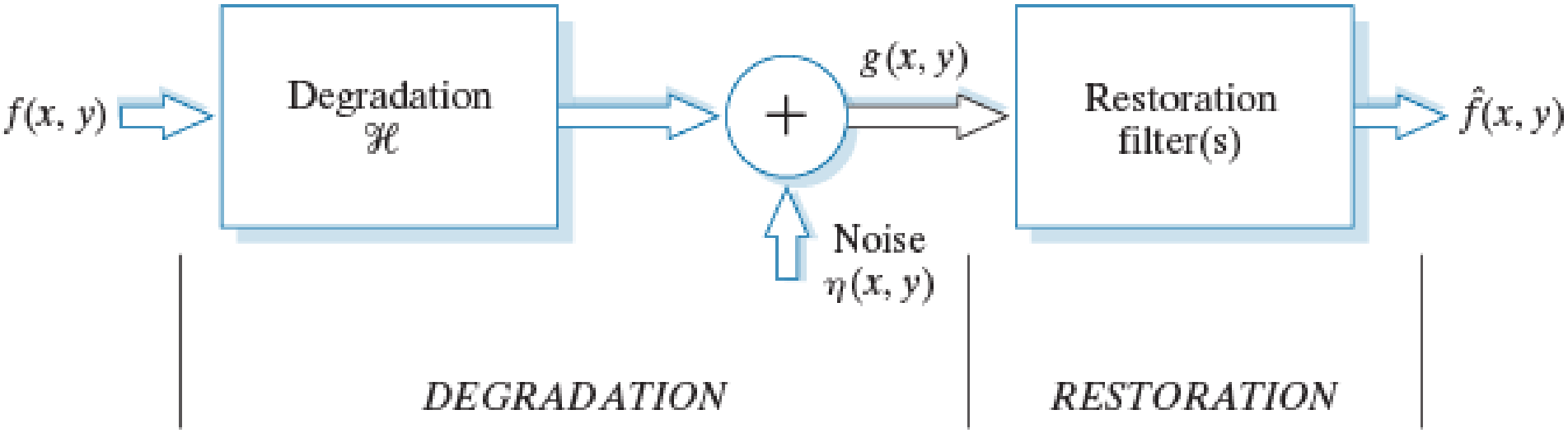


Image Degradation/Restoration Process

Spatial Domain: $g(x, y) = h(x, y) \star f(x, y) + \eta(x, y)$

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

H: linear, position-invariant
N: additive noise



Linear, Position-Invariant Degradations

Properties of the degradation function H

- Linear system

$$H[af_1(x, y) + bf_2(x, y)] = aH[f_1(x, y)] + bH[f_2(x, y)]$$

- Position(space)-invariant system

$$g(x, y) = H[f(x, y)]$$

$$H[f(x - \alpha, y - \beta)] = g(x - \alpha, y - \beta)$$

- c.f. 1-D signal
  - LTI (linear time-invariant system)

Linear, Position-Invariant Degradations

$$f(x,y)=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\alpha,\beta)\delta(x-\alpha,y-\beta)d\alpha d\beta$$

impulse

$$g(x,y)=H[f(x,y)]=H\left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\alpha,\beta)\delta(x-\alpha,y-\beta)d\alpha d\beta\right]$$

linear

$$g(x,y)=H[f(x,y)]=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} H\left[f(\alpha,\beta)\delta(x-\alpha,y-\beta)\right]d\alpha d\beta$$

$$g(x,y)=H[f(x,y)]=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\alpha,\beta)H\left[\delta(x-\alpha,y-\beta)\right]d\alpha d\beta$$

$$h(x,\alpha,y,\beta)=H\left[\delta(x-\alpha,y-\beta)\right]$$

$h(x,y)=H\left[\delta(x,y)\right]$

position-invariant

Impulse response (point spread function)

$$H\left[\delta(x-\alpha,y-\beta)\right]=h(x-\alpha,y-\beta)$$

Linear, Position-Invariant Degradations

$$g(x,y) = H[f(x,y)] = H\left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\alpha,\beta)\delta(x-\alpha,y-\beta)d\alpha d\beta\right]$$

linear

$$g(x,y) = H[f(x,y)] = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\alpha,\beta)H[\delta(x-\alpha,y-\beta)]d\alpha d\beta$$

position-invariant

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

$$\eta(x,y) \neq 0$$

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

Image Degradation/Restoration Model

- Linear system theory is ready
- Non-linear, position-dependent system
  - May be general and more accurate
  - Difficult to solve computationally
- Image restoration:

  for linear, position-invariant degradations,

  find H(u,v) and apply inverse process,

  also called image deconvolution

Noise and Images

The sources of noise in digital images arise during image acquisition (digitization) and transmission

- Imaging sensors can be affected by ambient conditions
- Interference can be added to an image during transmission

Noise Model

We can consider a noisy image to be modelled as follows (Assume *H* is an identity operator):

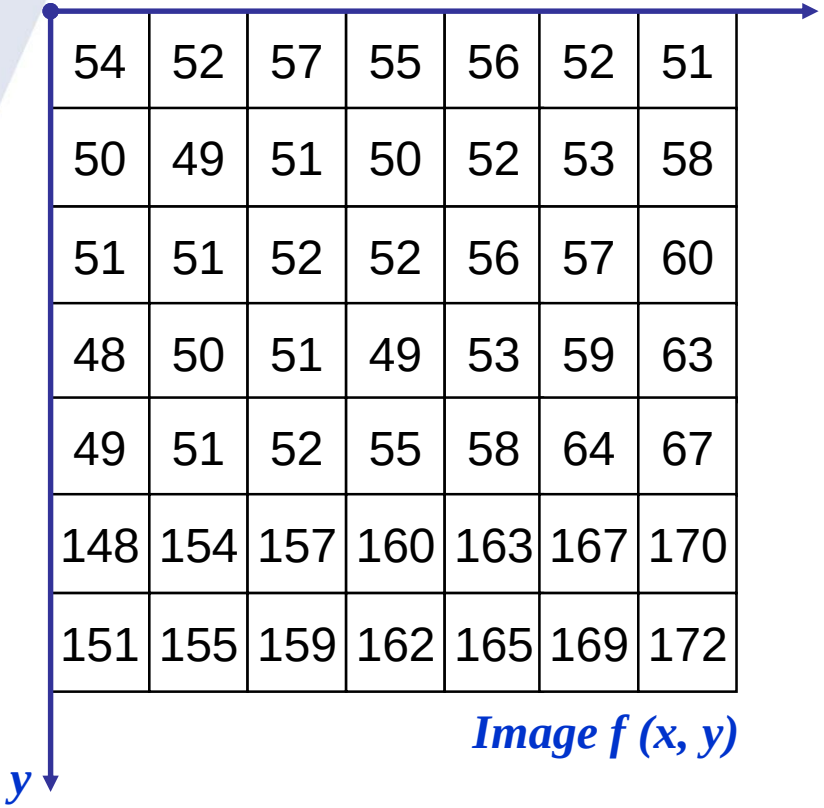
$$g(x, y) = f(x, y) + \eta(x, y)$$

where $f(x, y)$ is the original image pixel, $\eta(x, y)$ is the noise term and $g(x, y)$ is the resulting noisy pixel.

If we can estimate the model the noise in an image is based on, this will help us to figure out how to restore the image.

Noise Corruption Example

Original Image



54	52	57	55	56	52	51
50	49	51	50	52	53	58
51	51	52	52	56	57	60
48	50	51	49	53	59	63
49	51	52	55	58	64	67
148	154	157	160	163	167	170
151	155	159	162	165	169	172

Image f (x, y)

Noisy Image

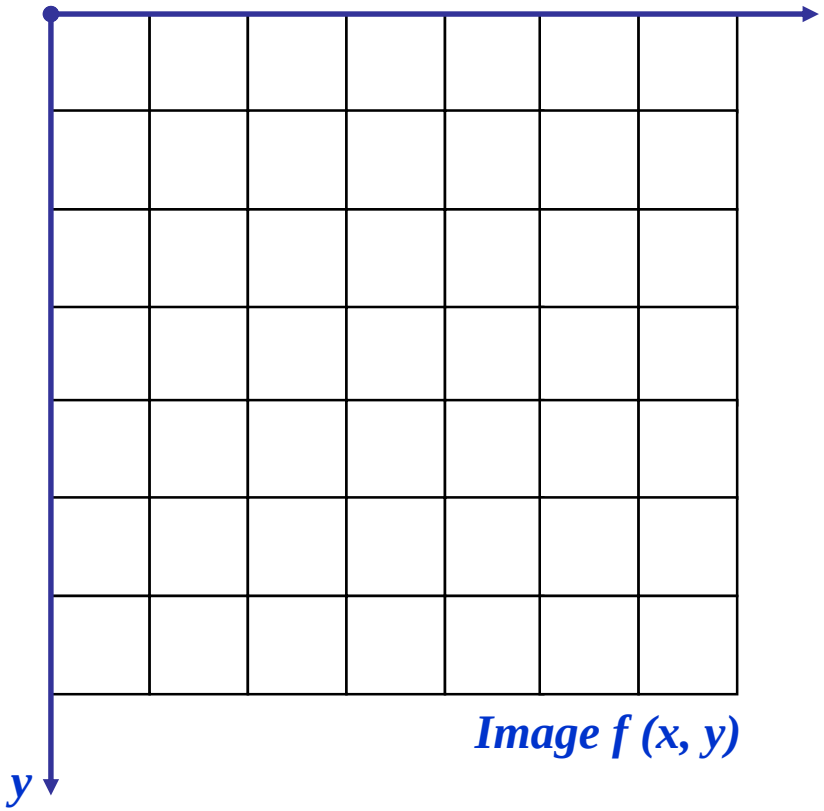
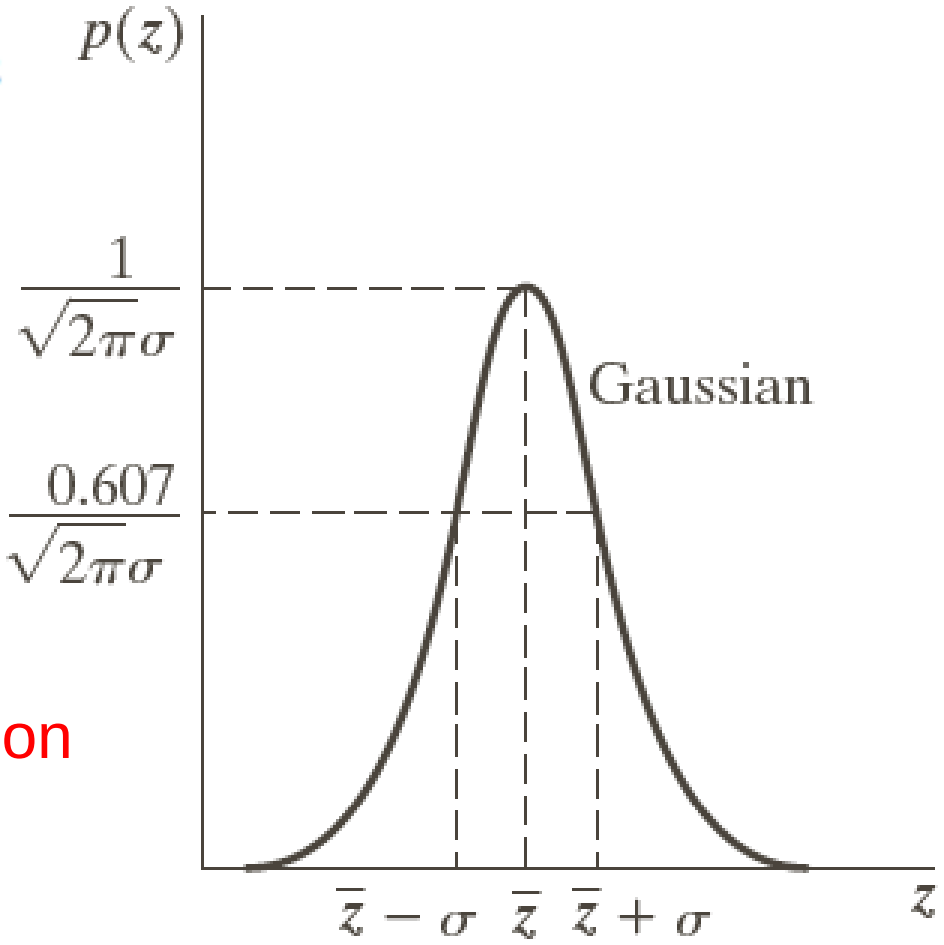


Image f (x, y)

Noise Models

- Gaussian noise

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



PDF: Probability Density Function

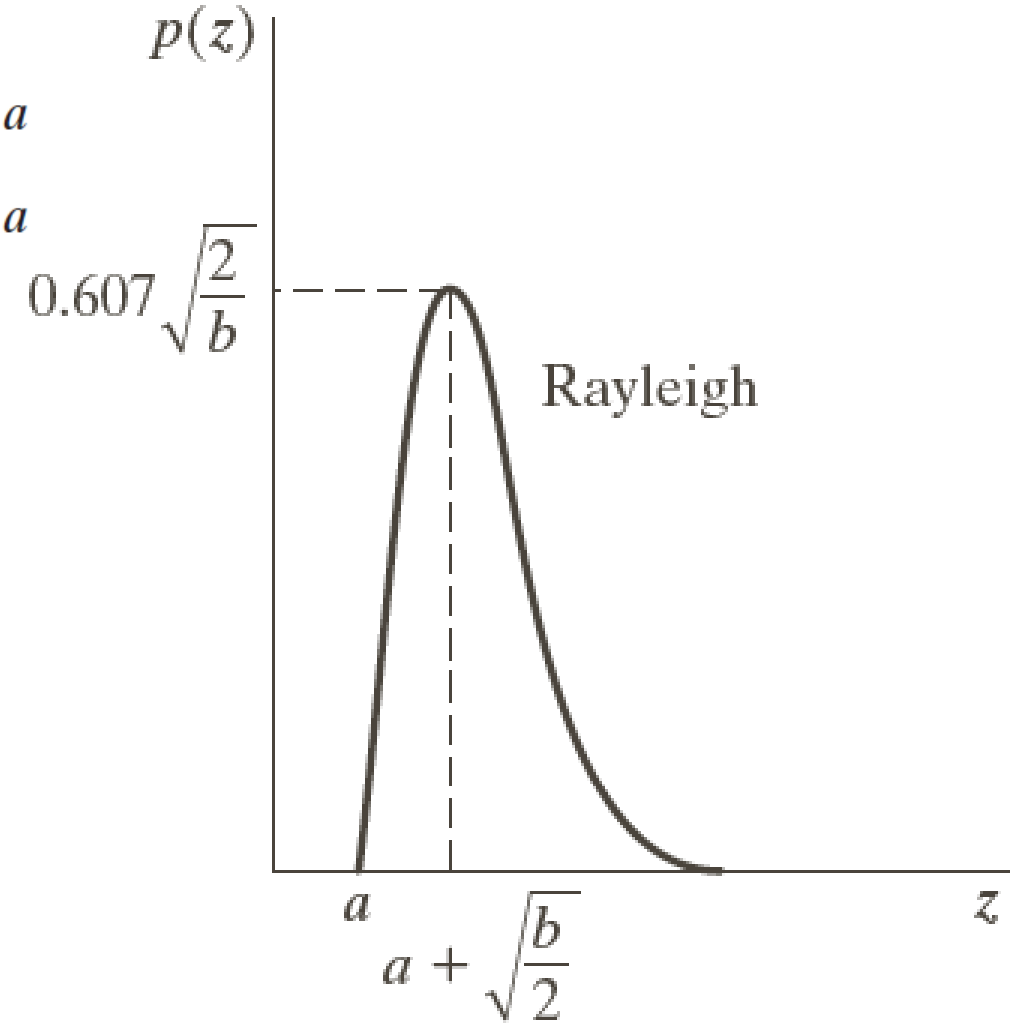
Noise Models

- Rayleigh noise

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

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

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



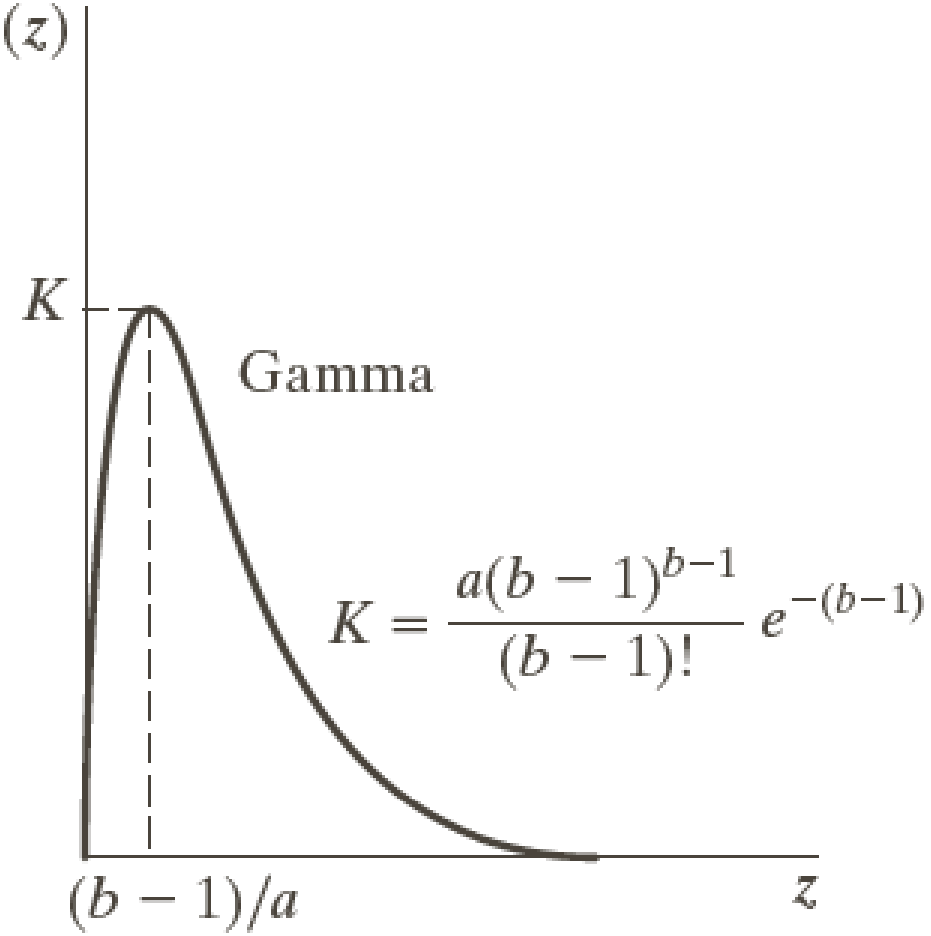
Noise Models

- Erlang (gamma) noise

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

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

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



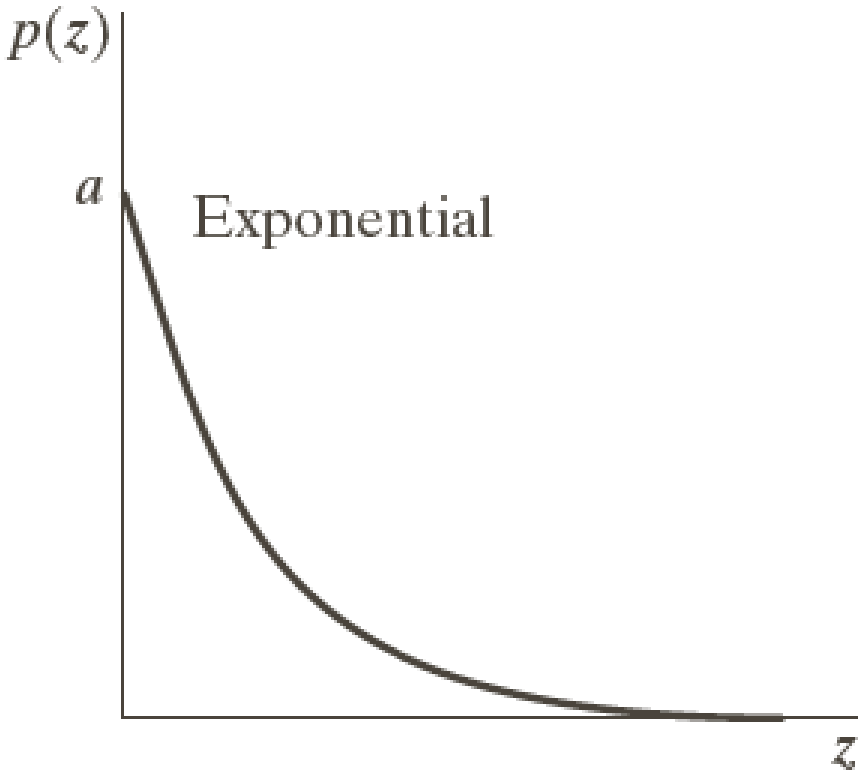
Noise Models

- Exponential noise

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

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

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



Erlang PDF, with $b = 1$

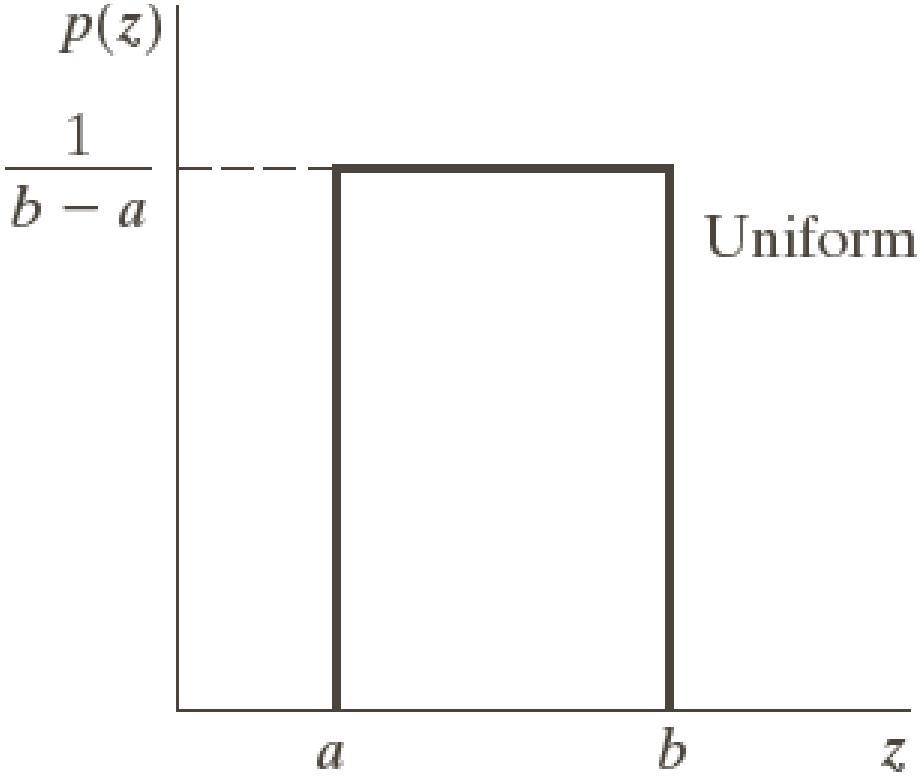
Noise Models

- Uniform noise

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

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

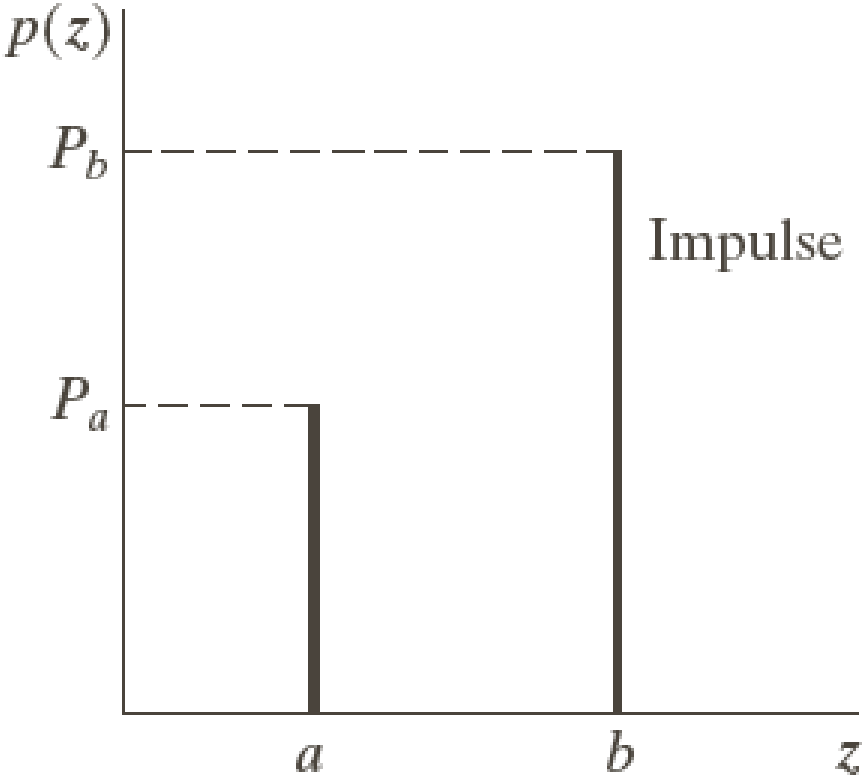
$$\sigma^2 = \frac{(b-a)^2}{12}$$



Noise Models

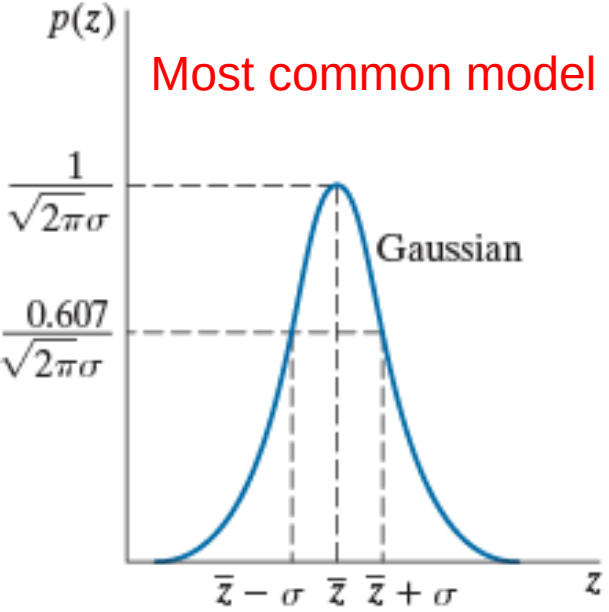
- Impulse (salt-and-pepper) noise

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

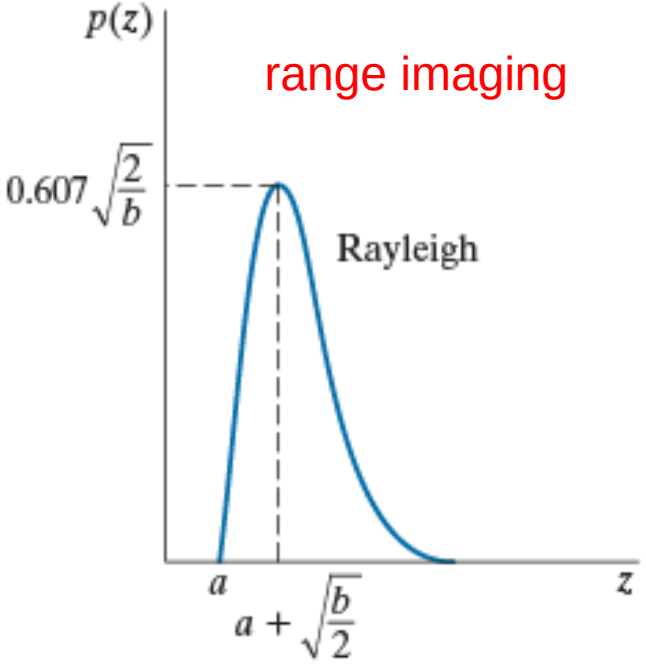


Noise Models

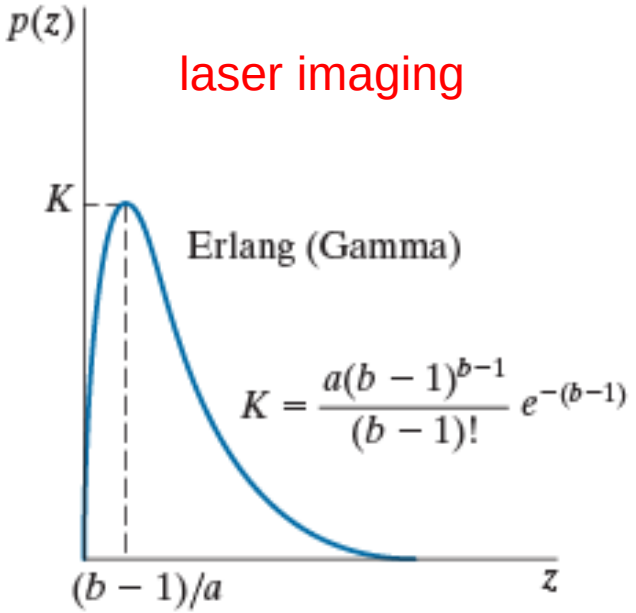
Most common model



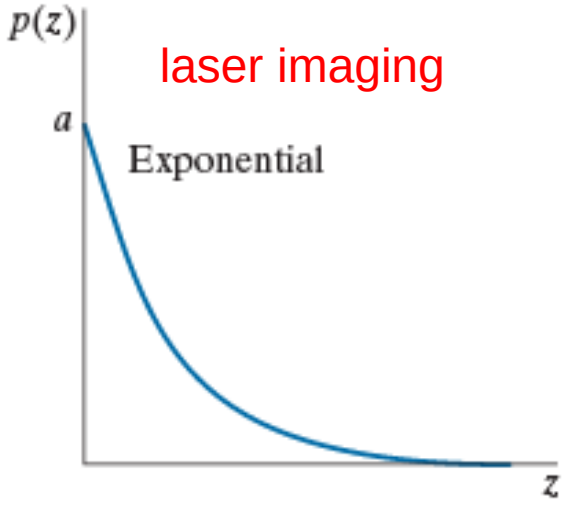
range imaging

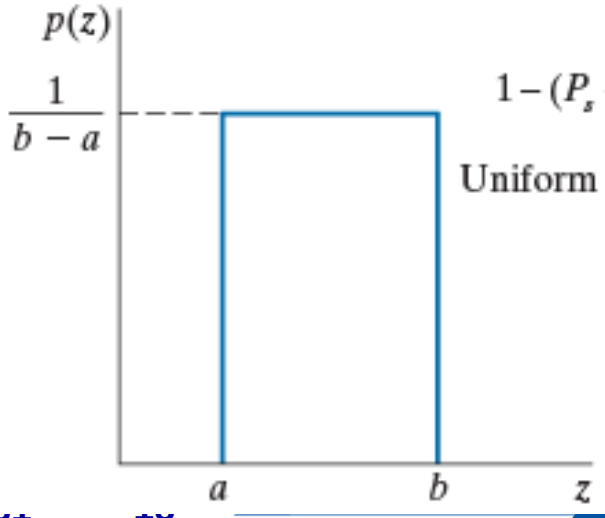


laser imaging

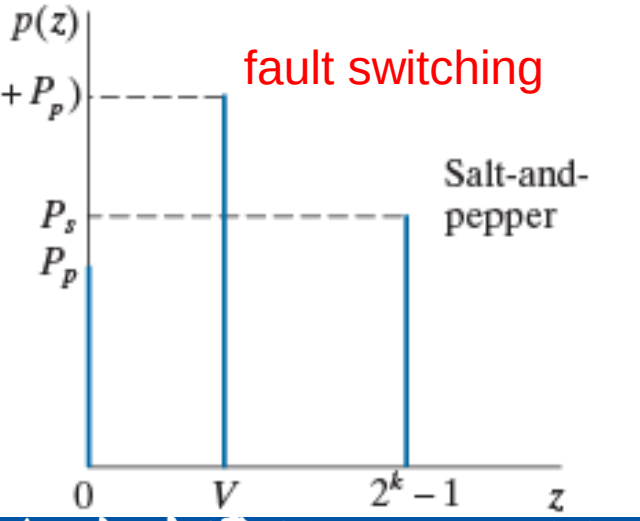


laser imaging





fault switching



Range Imaging – Depth Camera

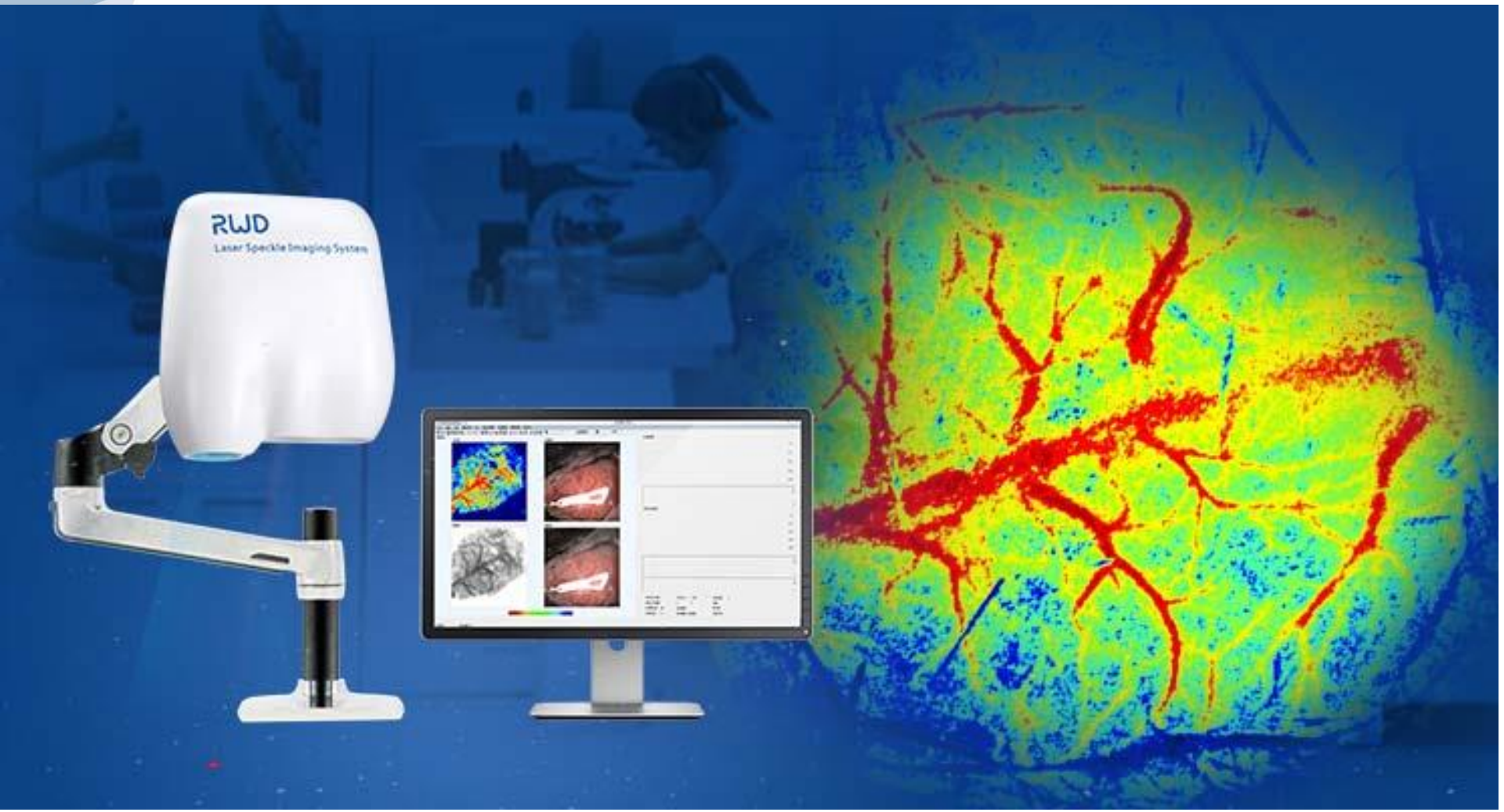
Microsoft Kinect V1

Microsoft Kinect V2

Microsoft Azure Kinect

PMD

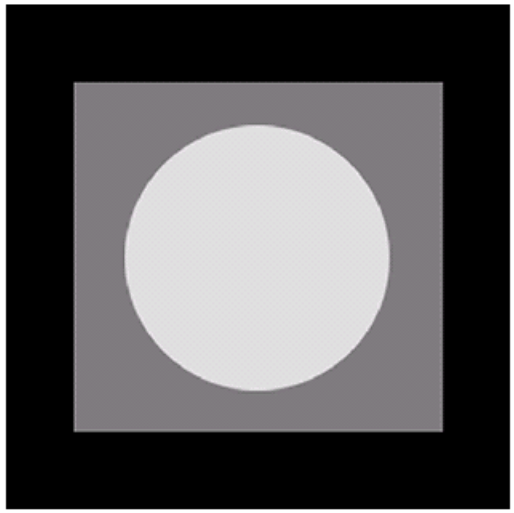
Laser Imaging



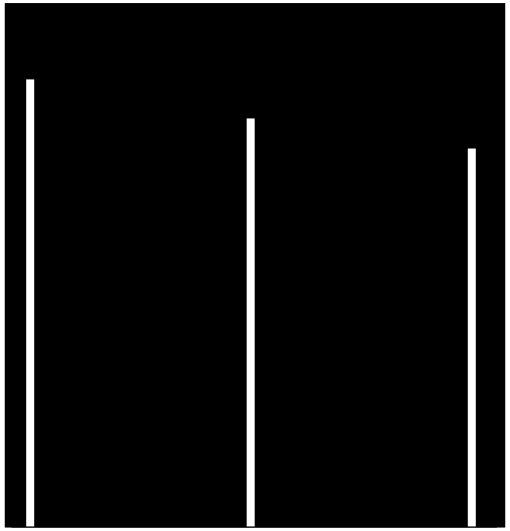
Noise Example

The test pattern to the right is ideal for demonstrating the addition of noise

The following slides will show the result of adding noise based on various models to this image

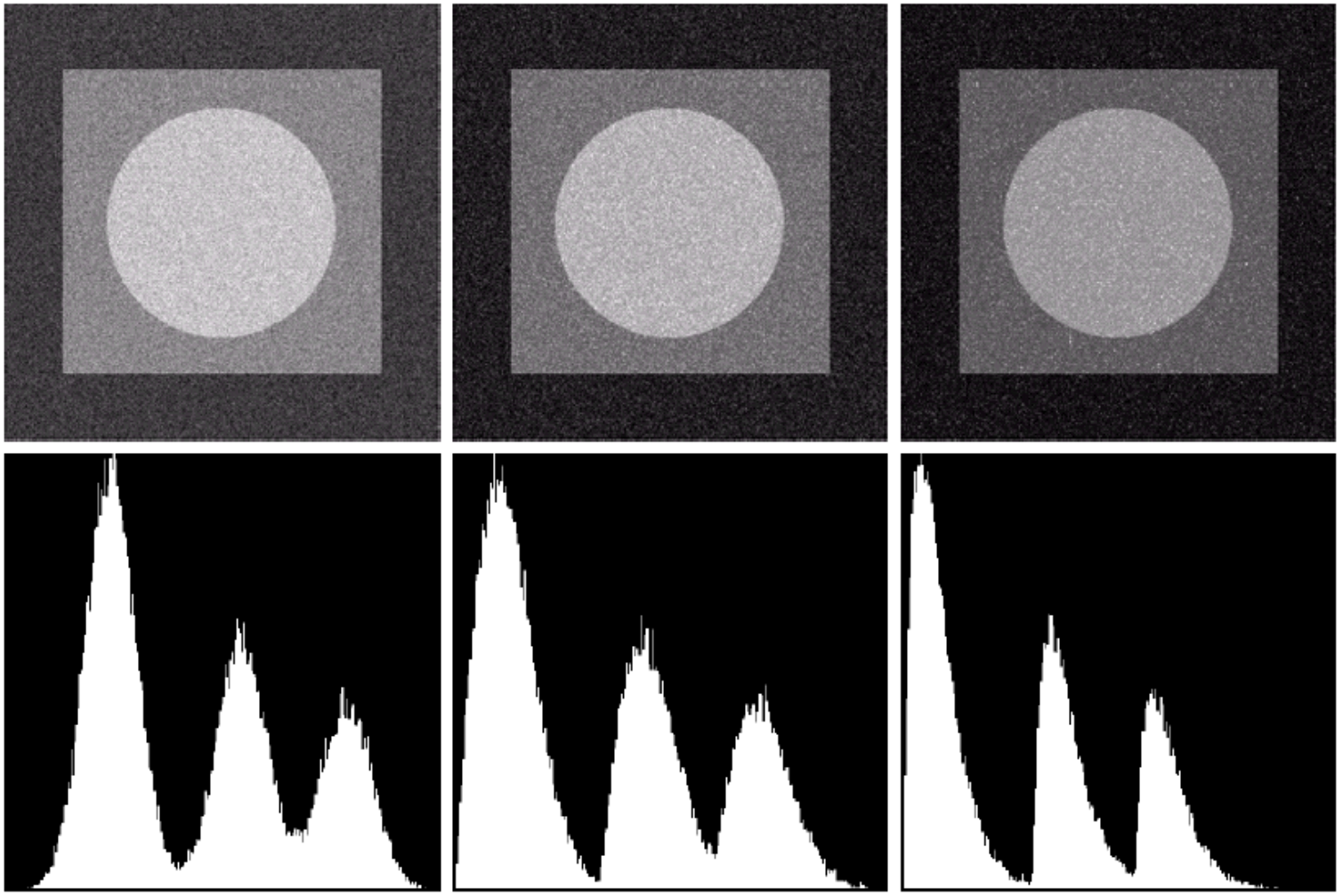


Image



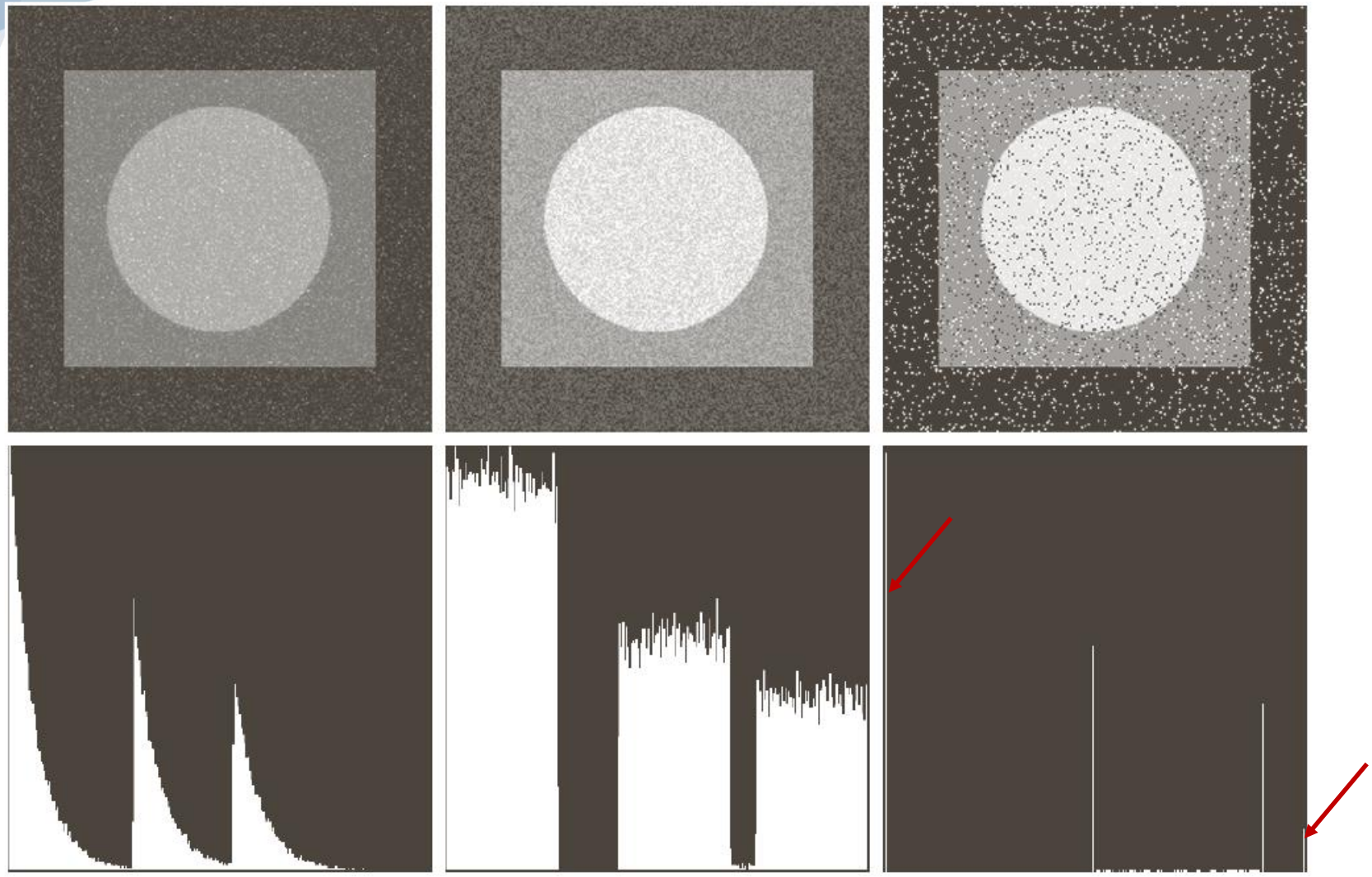
Histogram

Noise Example (cont…)



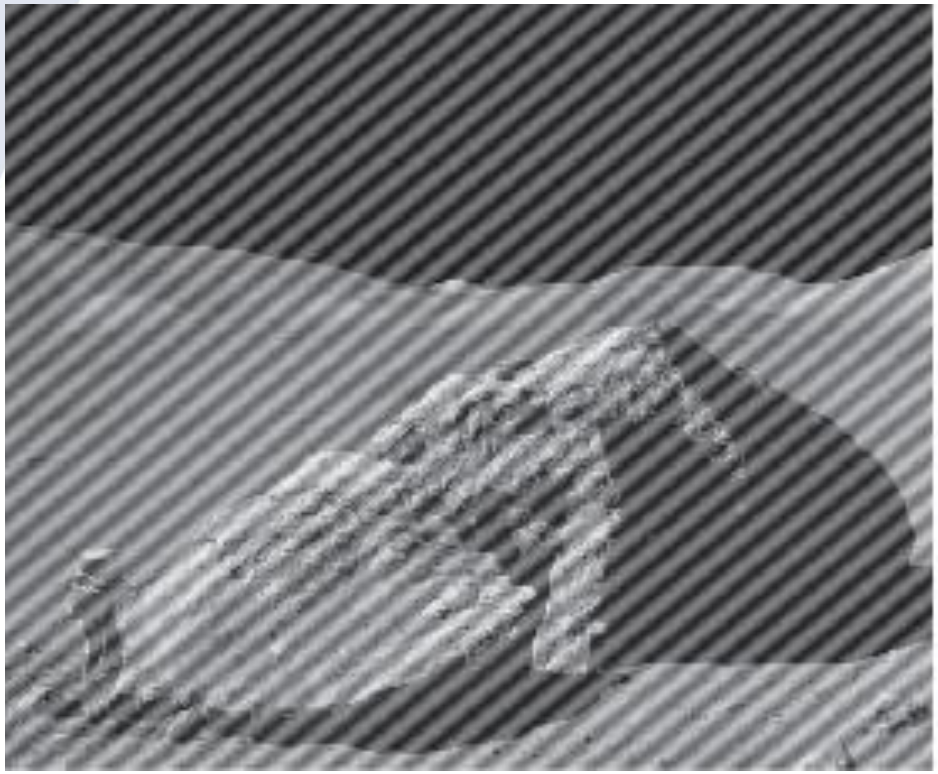
Gaussian Rayleigh Erlang

Noise Example (cont…)



Exponential Uniform Impulse

Periodic Noise

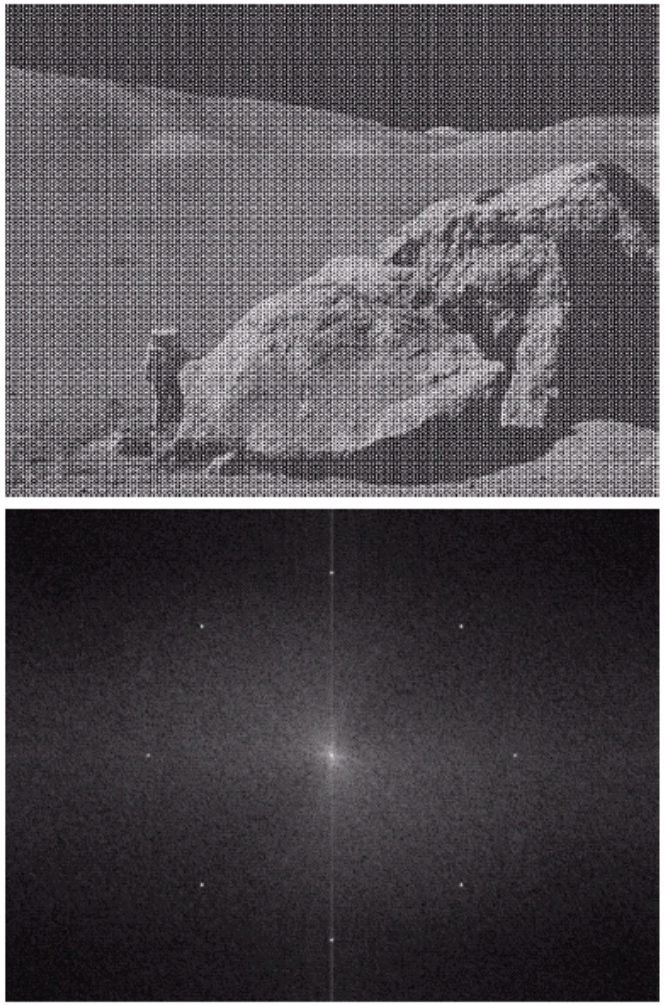


Periodic Noise

Typically arises due to electrical or electromagnetic interference

Gives rise to regular noise patterns in an image

Frequency domain techniques in the Fourier domain are most effective at removing periodic noise



Estimation of Noise Parameters

- A set of images of “flat” environments
- Small patches of constant background

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

Gaussian

Rayleigh

Uniform

Restoration in the Presence of Noise Only

- Degradation Model

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

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

- Spatial Filtering
  - Mean Filters
  - Order-Statistic Filters
  - Adaptive Filters
- Frequency Filtering

Mean Filters

- *Arithmetic Mean* Filter

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

$1/9$	$1/9$	$1/9$
$1/9$	$1/9$	$1/9$
$1/9$	$1/9$	$1/9$

This is implemented as the simple smoothing filter which blurs the image to remove noise.

Mean Filters

- *Geometric Mean Filter*

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

Achieves similar smoothing to the arithmetic mean, but tends to lose less image detail

Mean 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

Also does well for other kinds of noise such as Gaussian noise

Mean Filters

- *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 and adjusting its value changes the filter's behaviour

Positive values of Q eliminate pepper noise

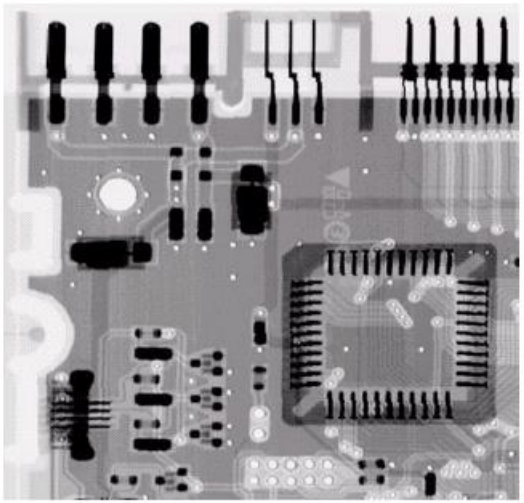
Negative values of Q eliminate salt noise

$Q = 0 \rightarrow$ *Arithmetic Mean*

$Q = -1 \rightarrow$ *Harmonic Mean*

Noise Removal Examples

Original Image



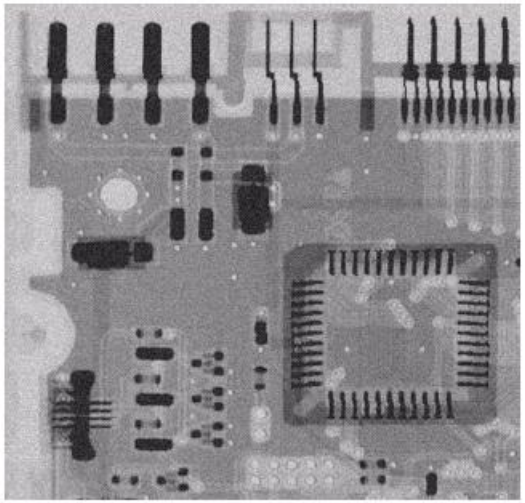
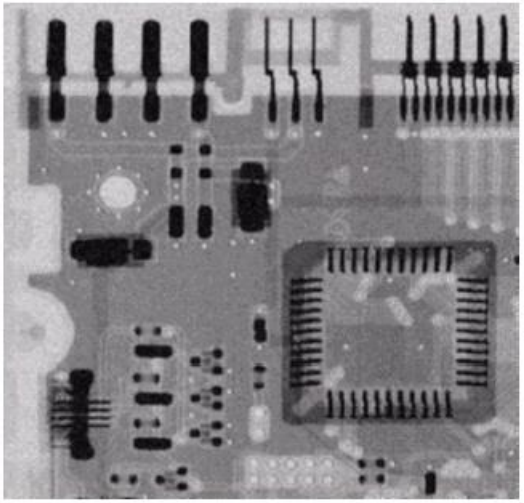
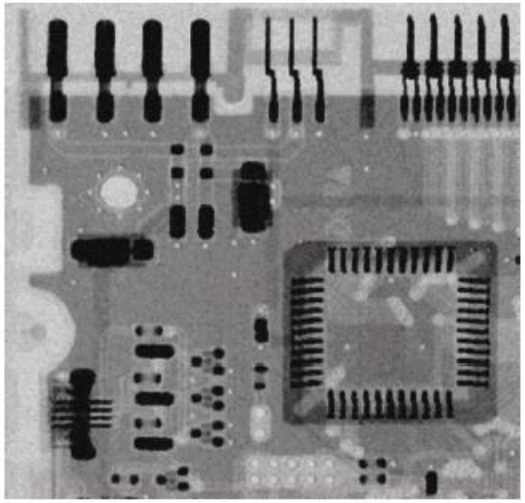


Image Corrupted By Gaussian Noise

After A 3*3 Arithmetic Mean Filter

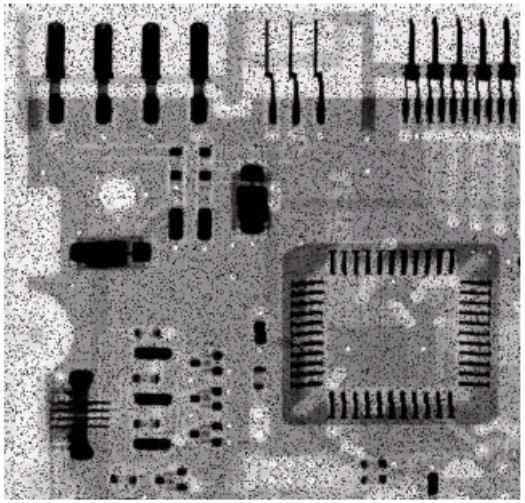




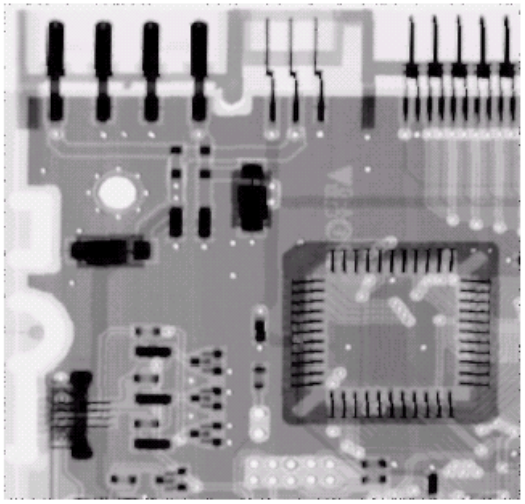
After A 3*3 Geometric Mean Filter

Noise Removal Examples (cont…)

Image Corrupted By Pepper Noise



Result of Filtering Above With 3*3 Contraharmonic Q=1.5



Noise Removal Examples (cont…)

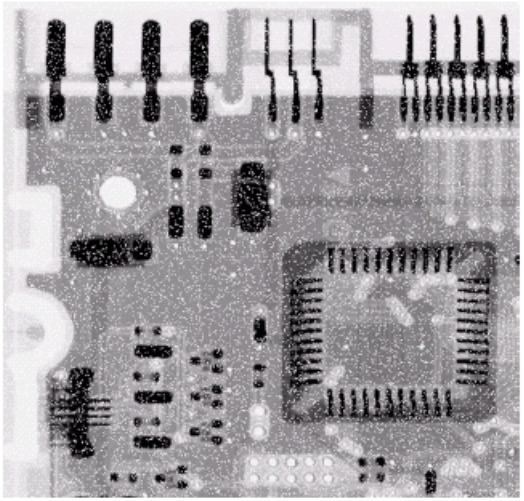
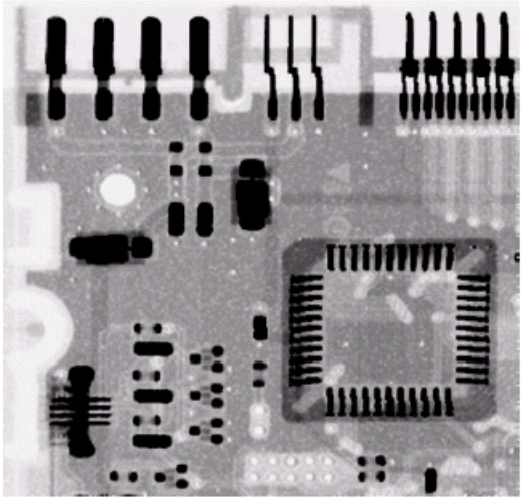


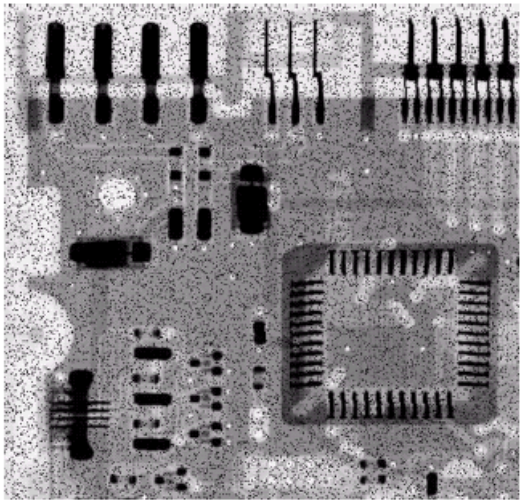
Image Corrupted By Salt Noise



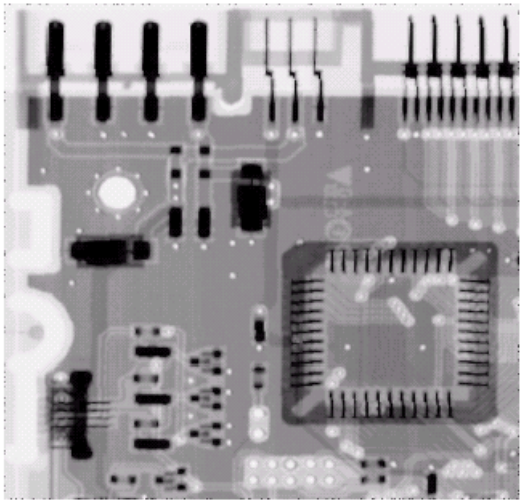
Result of Filtering Above With 3*3 Contraharmonic Q=-1.5

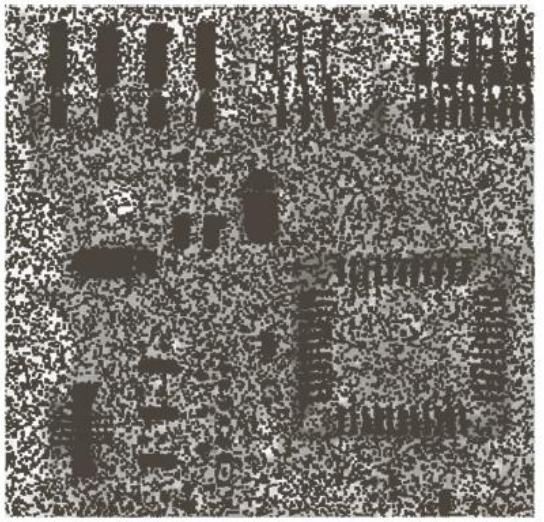
Choosing the wrong sign for Q

Image Corrupted By Pepper Noise



Result of Filtering Above With 3*3 Contraharmonic Q=1.5





Q = -1.5

Choosing the wrong sign for Q

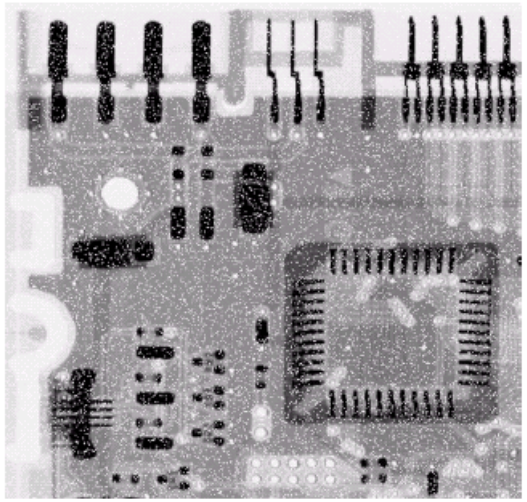
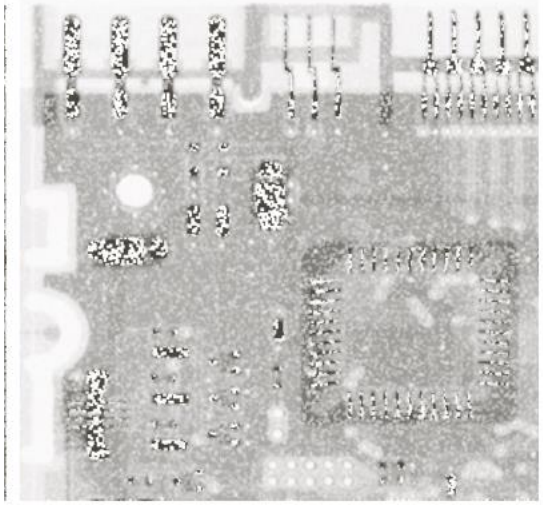
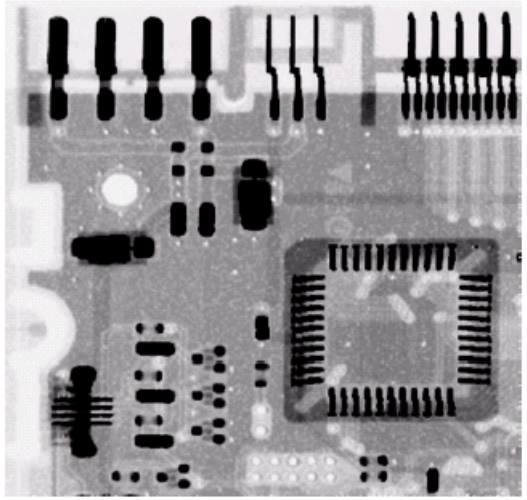


Image Corrupted By Salt Noise

Q = 1.5





Result of Filtering Above With 3*3 Contraharmonic Q=-1.5

Order Statistics Filters

Spatial filters that are based on ordering the pixel values that make up the neighbourhood operated on by the filter

Useful spatial filters include

- Median filter
- Max and min filter
- Midpoint filter
- Alpha trimmed mean filter

Median Filter

Median Filter:

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

Excellent at noise removal, without the smoothing effects that can occur with other smoothing filters

Particularly effective when salt and pepper noise is present

Noise Corruption Example

Original Image

54	52	57	55	56	52	51
50	49	51	50	52	53	58
51	204	52	52	0	57	60
48	50	51	49	53	59	63
49	51	52	55	58	64	67
50	54	57	60	63	67	70
51	55	59	62	65	69	72

Image f (x, y)

Filtered Image

	51			52		

Image f (x, y)

Max and Min Filter

Max Filter:

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

Min Filter:

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

Max filter is good for pepper noise and min is good for salt noise

Noise Corruption Example

Original Image

54	52	57	55	56	52	51
50	49	51	50	52	53	58
51	204	52	52	0	57	60
48	50	51	49	53	59	63
49	51	52	55	58	64	67
50	54	57	60	63	67	70
51	55	59	62	65	69	72

x

y

Image f (x, y)

Filtered Image

x

y

Image f (x, y)

Midpoint Filter

Midpoint Filter:

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

Good for random Gaussian and uniform noise

Alpha-Trimmed Mean Filter

Alpha-Trimmed Mean Filter:

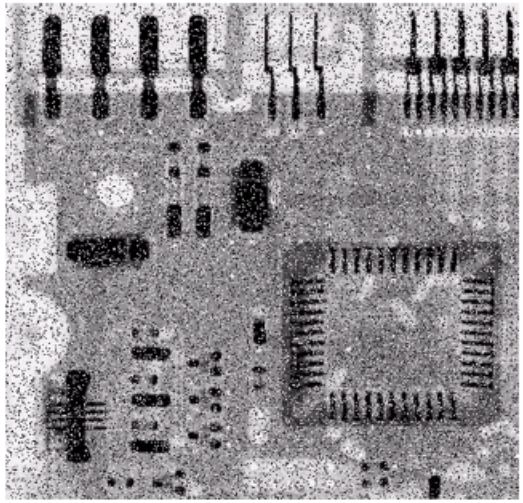
$$\hat{f}(x,y) = \frac{1}{mn-d} \sum_{(s,t)\in S_{xy}} g_r(s,t)$$

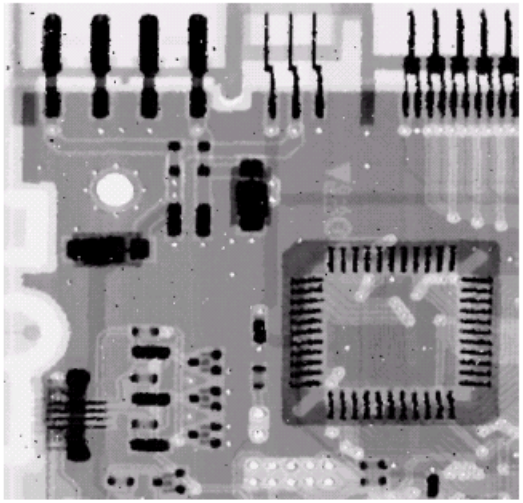
We can delete the $d/2$ lowest and $d/2$ highest grey levels

So $g_r(s, t)$ represents the remaining $mn - d$ pixels

Noise Removal Examples

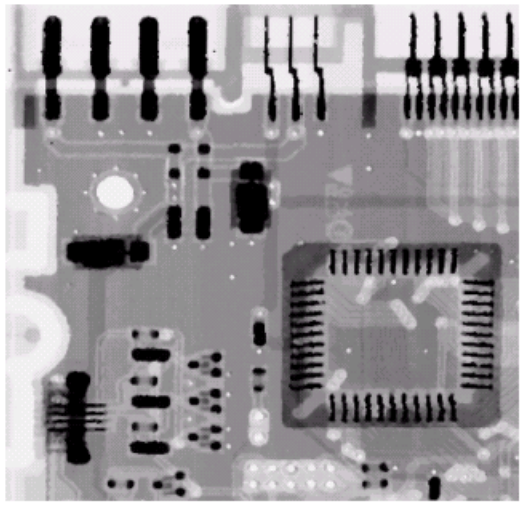
Image Corrupted By Salt And Pepper Noise

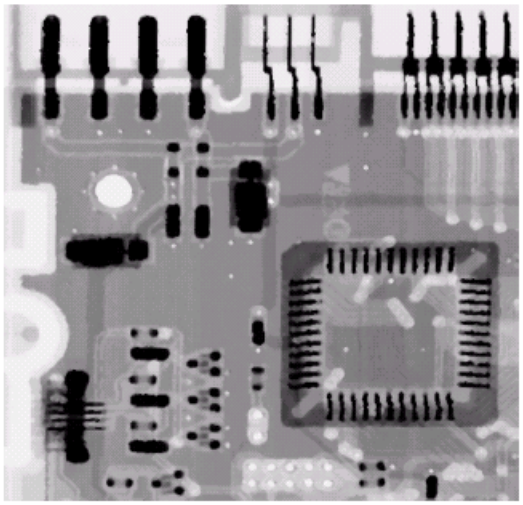




Result of 1 Pass With A 3*3 Median Filter

Result of 2 Passes With A 3*3 Median Filter

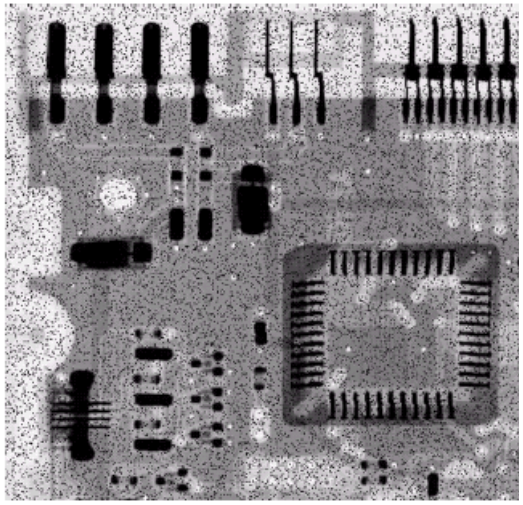




Result of 3 Passes With A 3*3 Median Filter

Noise Removal Examples (cont…)

Image Corrupted By Pepper Noise



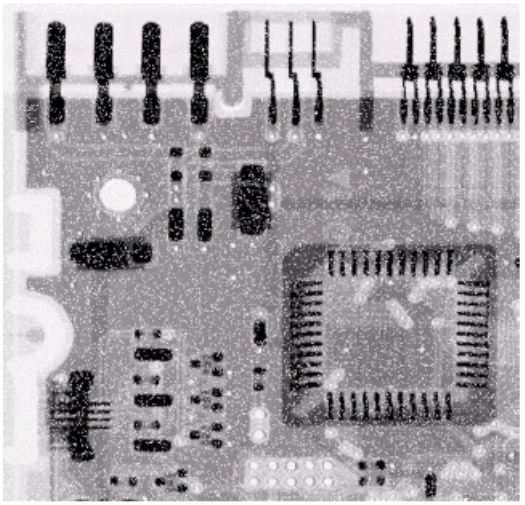
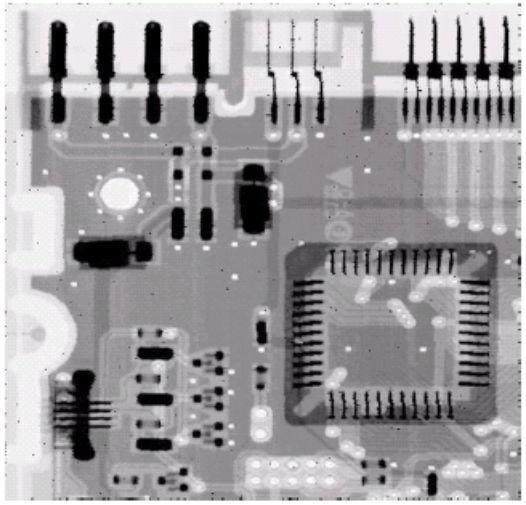
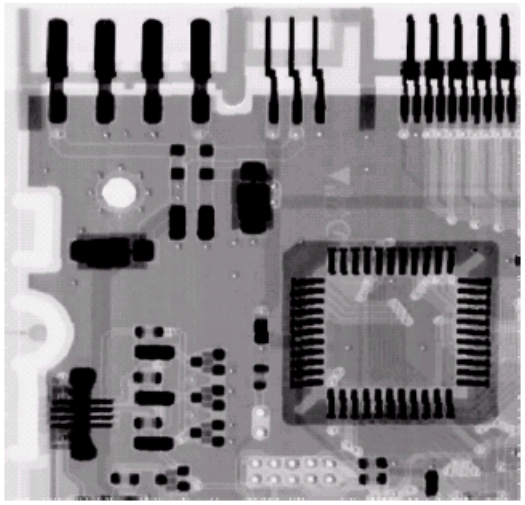


Image Corrupted By Salt Noise

Result Of Filtering Above With A 3*3 Max Filter





Result Of Filtering Above With A 3*3 Min Filter

Noise Removal Examples (cont…)

Image Corrupted By Uniform Noise

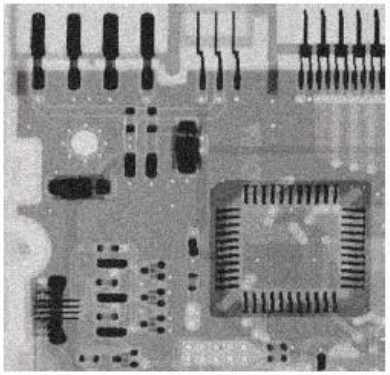
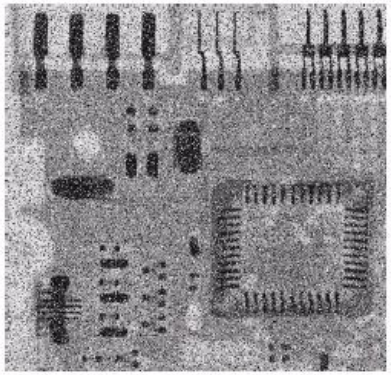
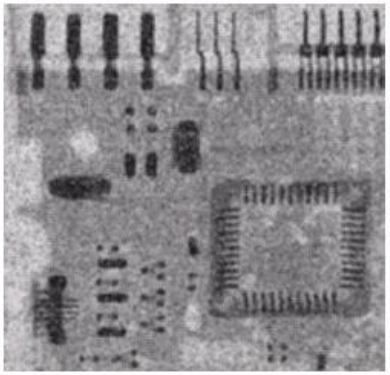


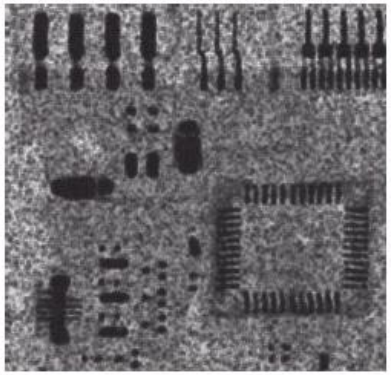
Image Further Corrupted By Salt and Pepper Noise



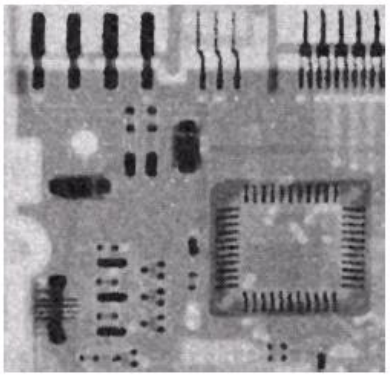
Filtered By 5*5 Arithmetic Mean Filter



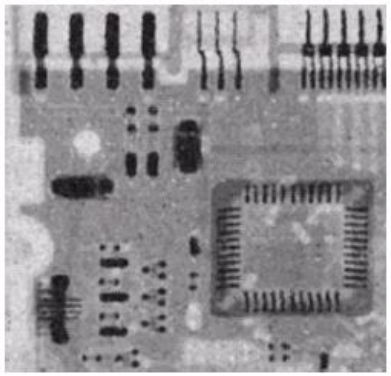
Filtered By 5*5 Geometric Mean Filter



Filtered By 5*5 Median Filter



Filtered By 5*5 Alpha-Trimmed Mean Filter



Adaptive Filters

The filters discussed so far are applied to an entire image without any regard for how image characteristics vary from one point to another

The behaviour of **adaptive filters** changes depending on the **local** characteristics of the image inside the filter region

Adaptive Filters

- Adaptive, local noise reduction filter

$$\hat{f}(x,y) = g(x,y) - \frac{\sigma_\eta^2}{\sigma_{S_{xy}}^2}\left[g(x,y) - \overline{z}_{S_{xy}}\right]$$

σ_η^2 Variance of the noise

$\overline{z}_{S_{xy}}$ Local mean of the pixels in S_{xy}

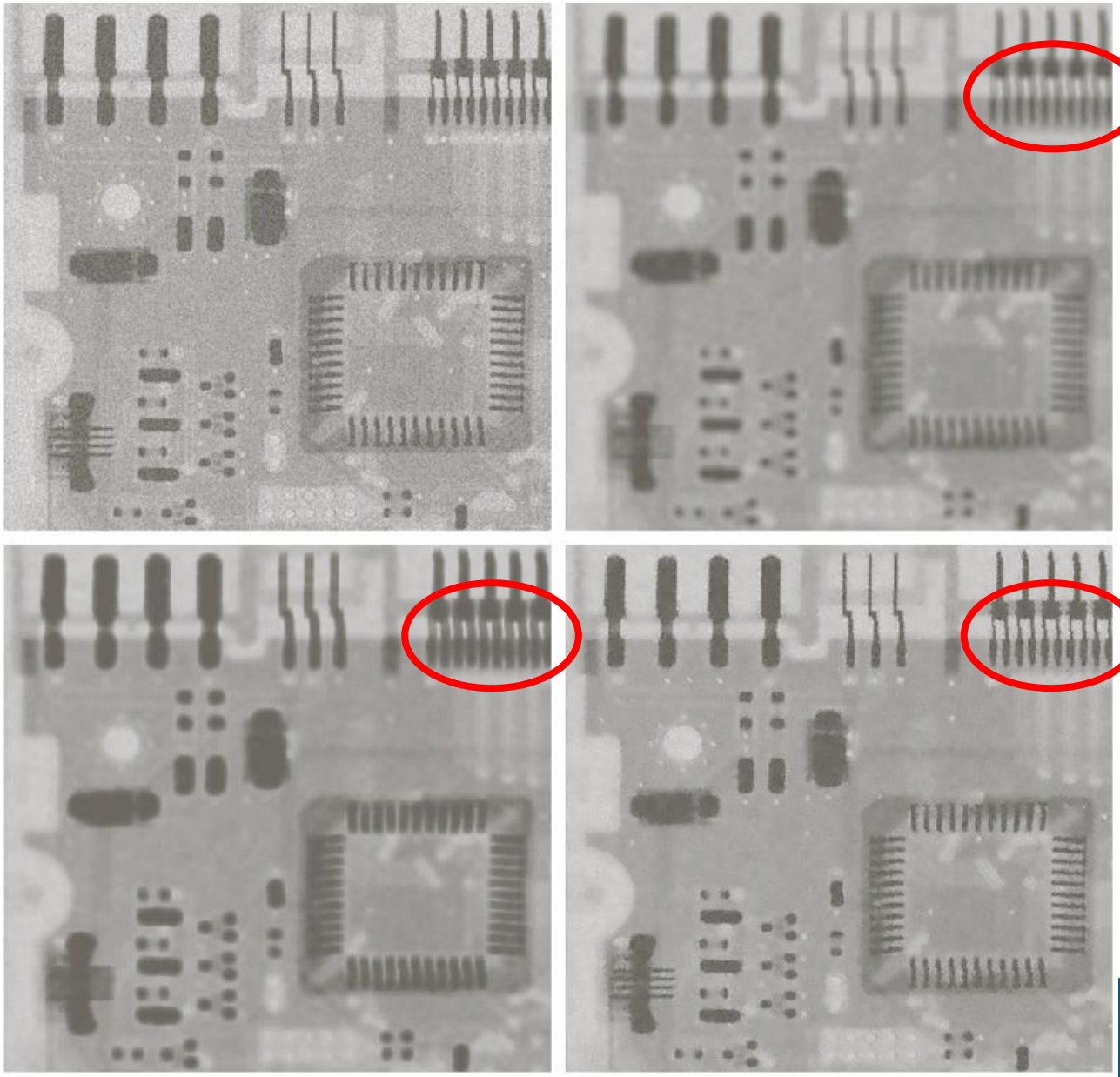
$\sigma_{S_{xy}}^2$ Local variance of the pixels in S_{xy}

1. If σ_η^2 is zero, the filter should return simply the value of g at (x, y). This is the trivial, zero-noise case in which g is equal to f at (x, y).
2. If the local variance $\sigma_{S_{xy}}^2$ is high relative to σ_η^2, the filter should return a value close to g at (x, y). A high local variance typically is associated with edges, and these should be preserved.
3. If the two variances are equal, we want the filter to return the arithmetic mean value of the pixels in S_{xy}. This condition occurs when the local area has the same properties as the overall image, and local noise is to be reduced by averaging.

Noise Removal Example

a b
c d

FIGURE 5.13
(a) Image corrupted by additive Gaussian noise of zero mean and variance 1000.
(b) Result of arithmetic mean filtering.
(c) Result of geometric mean filtering.
(d) Result of adaptive noise reduction filtering. All filters were of size 7×7.



Adaptive Median Filtering

Remember that filtering looks at each original pixel image in turn and generates a new filtered pixel

First examine the following notation:

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

Adaptive Median Filtering (cont…)

Level A: $A1 = z_{med} - z_{min}$

$A2 = z_{med} - z_{max}$

If $A1 > 0$ and $A2 < 0$, Go to level B

Else increase the window size

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

Else output z_{med}

Level 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 Median Filtering (cont…)

The key to understanding the algorithm is to remember that the adaptive median filter has three purposes:

- Remove impulse noise
- Provide smoothing of other noise
- Reduce distortion

Adaptive Filtering Example

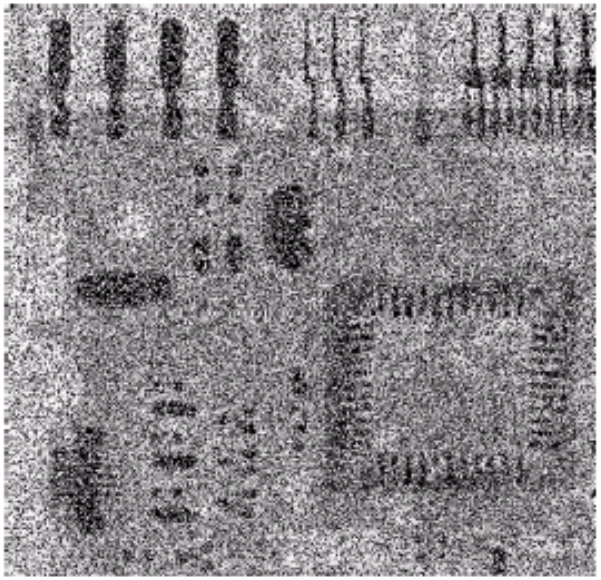
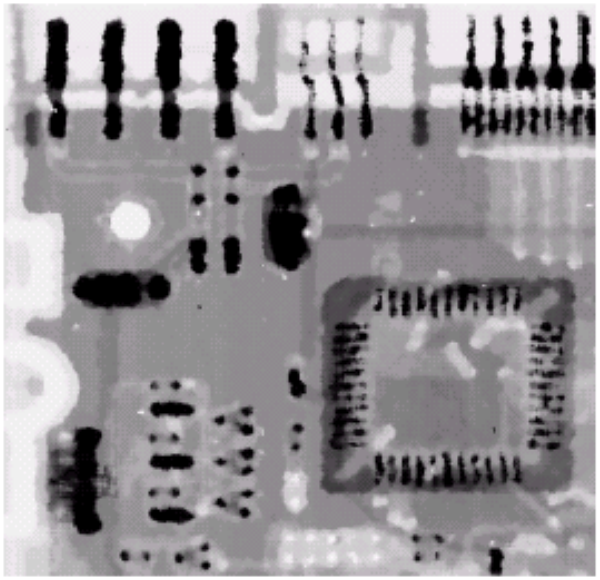
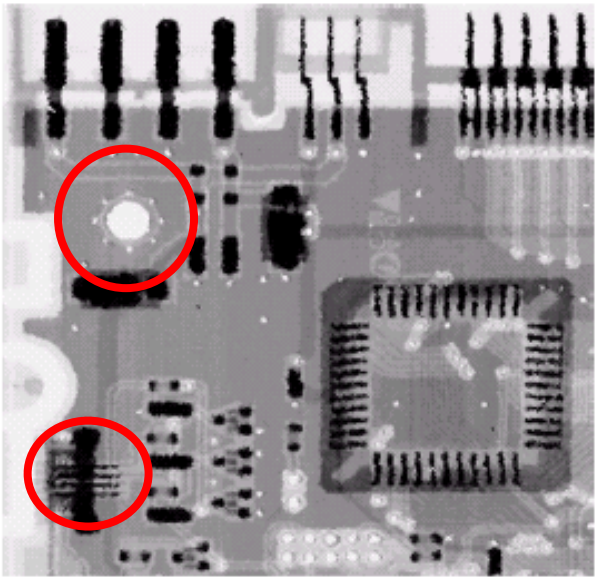


Image corrupted by salt and pepper noise with probabilities $P_a = P_b$=0.25



Result of filtering with a 7 * 7 median filter



Result of adaptive median filtering with i = 7

Band Reject Filters

Removing periodic noise from an image involves removing a particular range of frequencies from that image

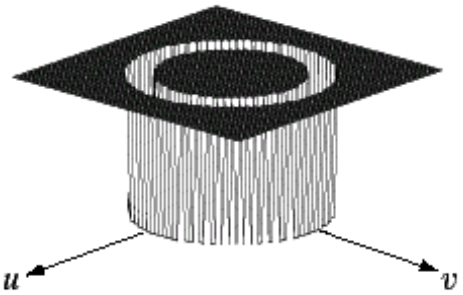
Band reject filters can be used for this purpose

An ideal band reject filter is given as follows:

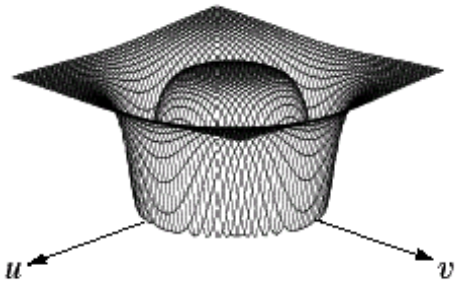
$$H(u,v)=\begin{cases}1 & if\ D(u,v)<D_0-\dfrac{W}{2}\\ 0 & if\ D_0-\dfrac{W}{2}\le D(u,v)\le D_0+\dfrac{W}{2}\\ 1 & if\ D(u,v)>D_0+\dfrac{W}{2}\end{cases}$$

Band Reject Filters (cont…)

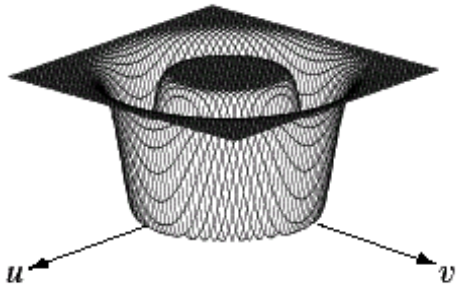
The ideal band reject filter is shown below, along with Butterworth and Gaussian versions of the filter



Ideal Band Reject Filter



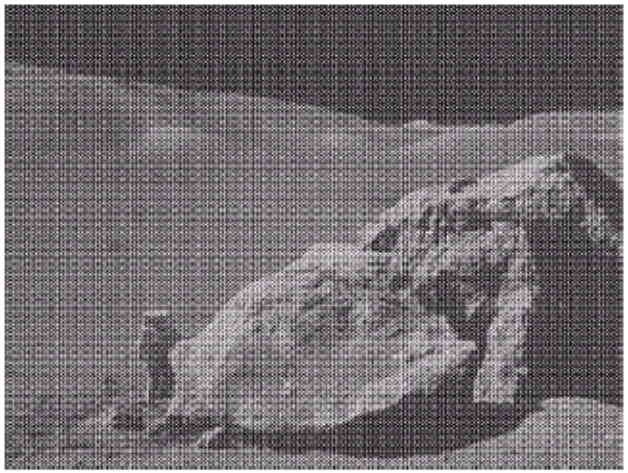
Butterworth Band Reject Filter (of order 1)



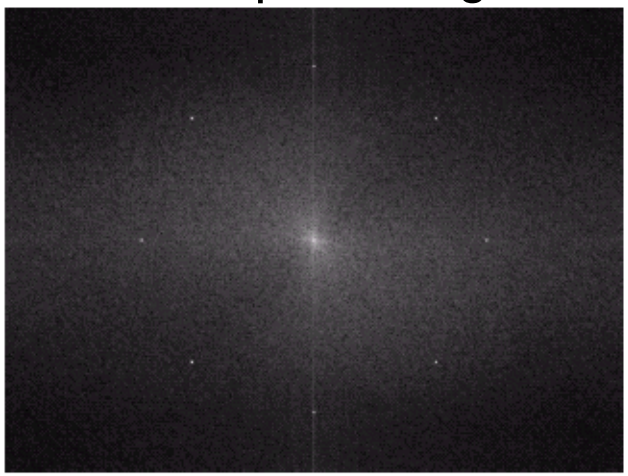
Gaussian Band Reject Filter

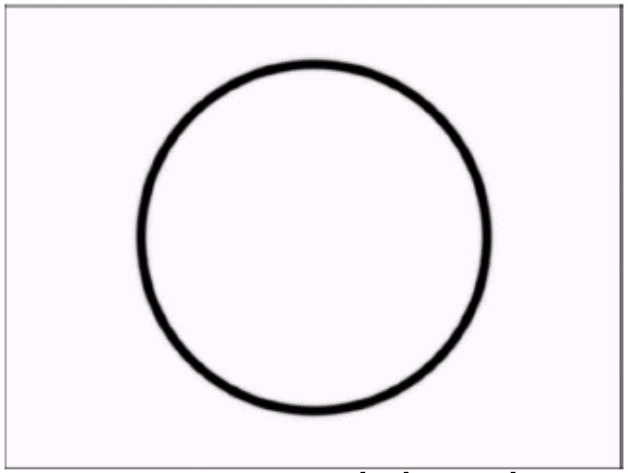
Band Reject Filter Example

Image corrupted by sinusoidal noise



Fourier spectrum of corrupted image





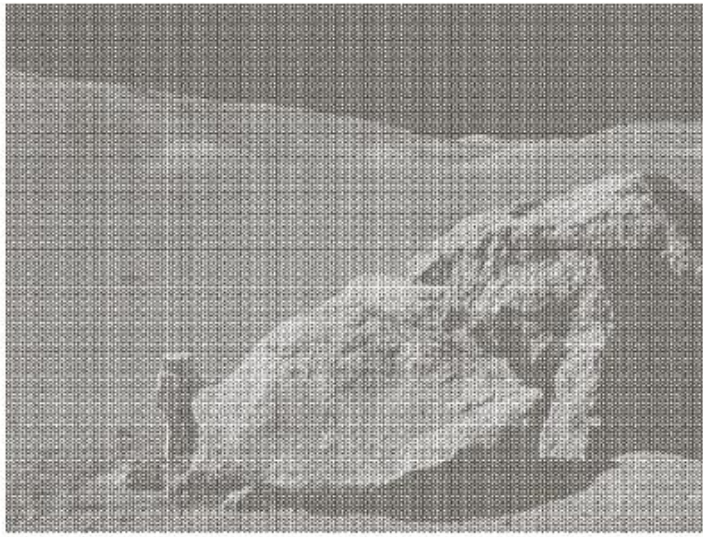
Butterworth band reject filter

Filtered image

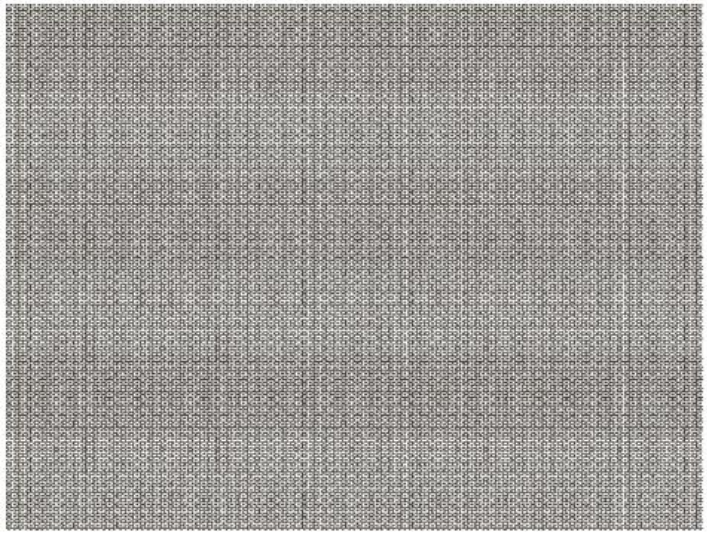
Bandpass Filters

- Obtain a BPF from a BRF

$$H_{\mathrm{BP}}(u, v) = 1 - H_{\mathrm{BR}}(u, v)$$



Degraded image

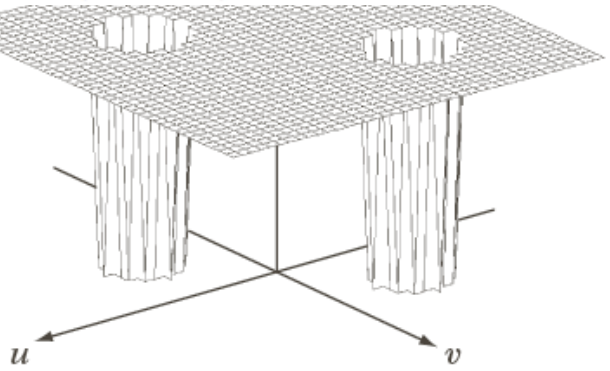


Noise pattern

Notch Filters

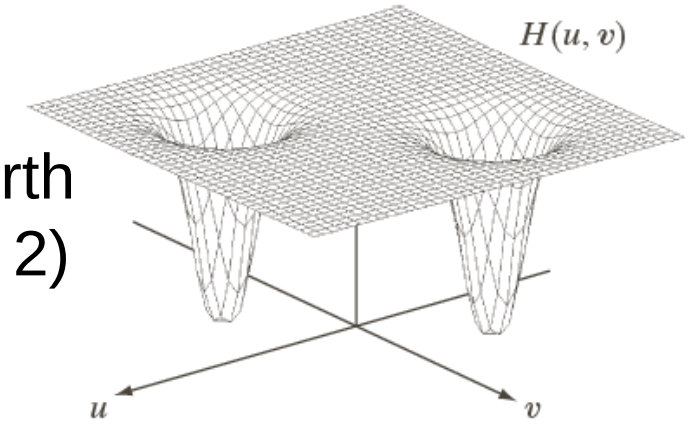
A notch filter rejects (or passes) frequencies in predefined neighborhoods about a center frequency.

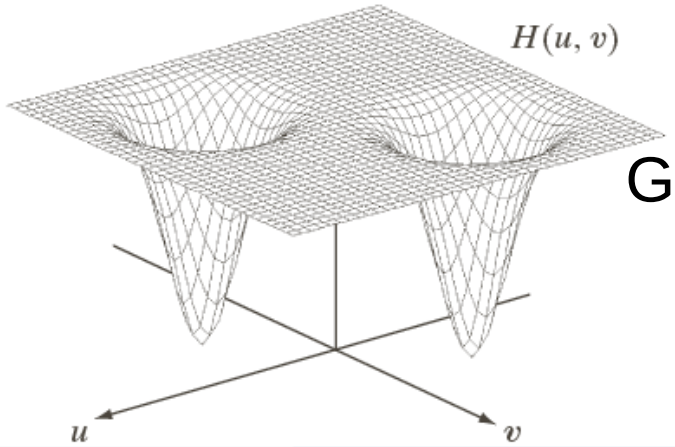
$$H_{NP}(u,v) = 1 - H_{NR}(u,v)$$



Ideal

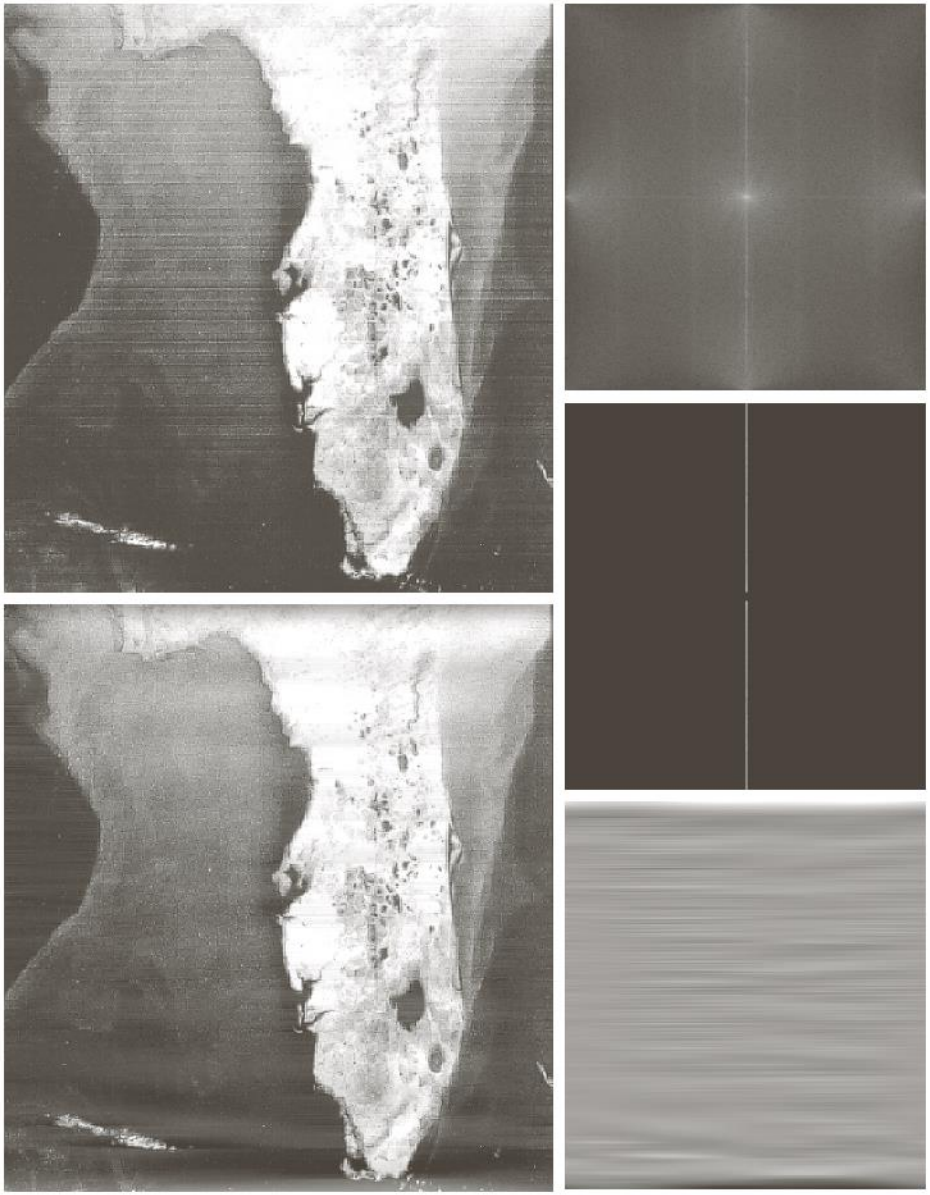
Butterworth (of order 2)





Gaussian

Notch Filter Example



a b
c
e d

FIGURE 5.19
(a) Satellite image of Florida and the Gulf of Mexico showing horizontal scan lines. (b) Spectrum. (c) Notch pass filter superimposed on (b). (d) Spatial noise pattern. (e) Result of notch reject filtering. (Original image courtesy of NOAA.)

Optimum Notch Filtering

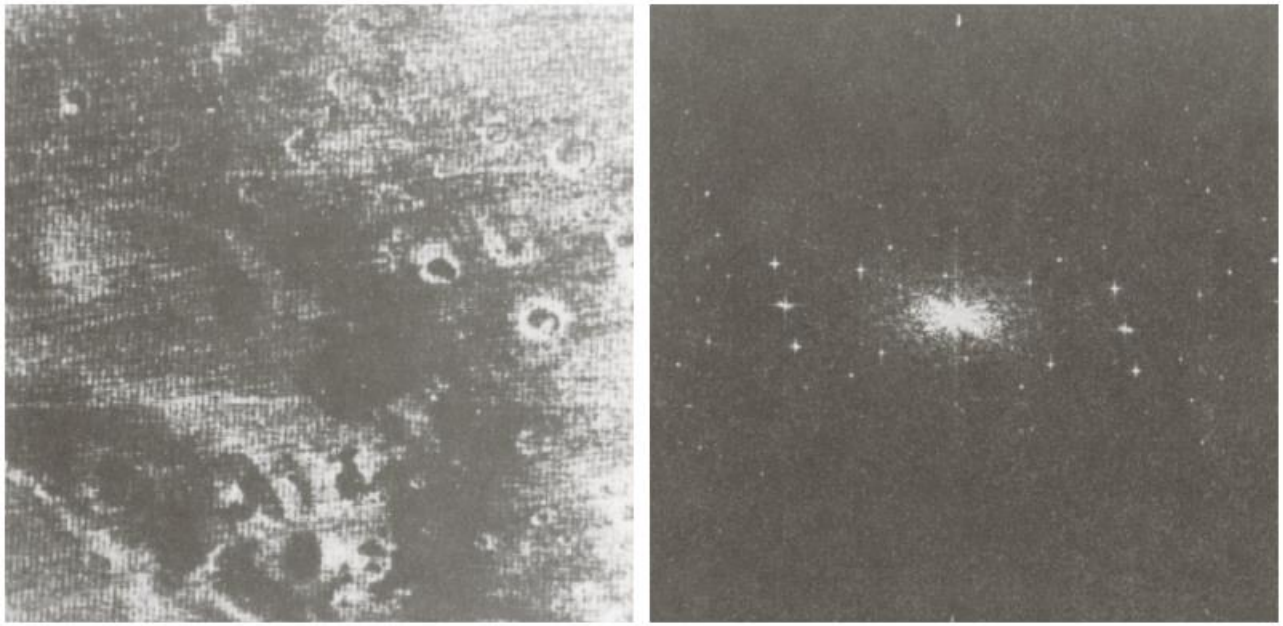
- Minimize the variance of $\hat{f}(x, y)$ over a specified neighborhood of every point (x,y)

$$\eta(x, y) = \Im^{-1}\{H_{\mathrm{NP}}(u, v)G(u, v)\}$$

$$\hat{f}(x, y) = g(x, y) - w(x, y)\eta(x, y)$$

a b

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



Optimum Notch Filtering

- Minimize the variance of $\hat{f}(x, y)$ over a specified neighborhood of every point (x,y)

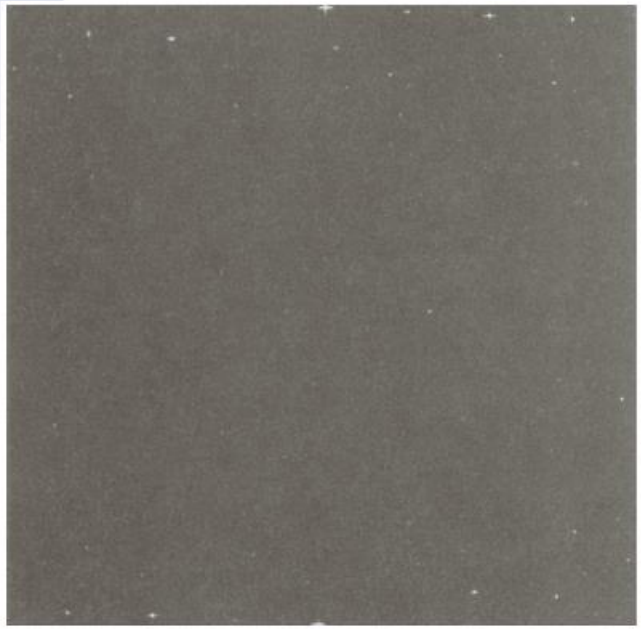
$$\eta(x, y) = \Im^{-1}\{H_{\mathrm{NP}}(u, v)G(u, v)\}$$

$$\hat{f}(x, y) = g(x, y) - w(x, y)\eta(x, y)$$

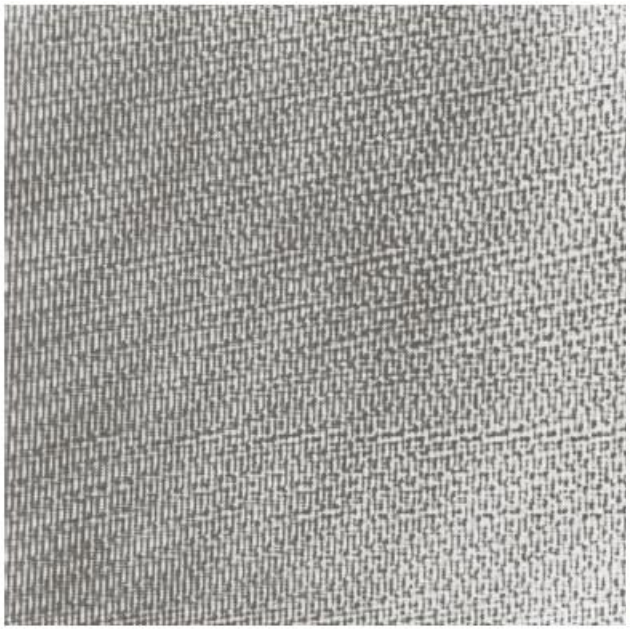
$$\frac{\partial \sigma^2(x, y)}{\partial w(x, y)} = 0$$

$$w(x, y) = \frac{\overline{g(x, y)\eta(x, y)} - \overline{g}(x, y)\overline{\eta}(x, y)}{\overline{\eta^2}(x, y) - \overline{\eta}^2(x, y)}$$

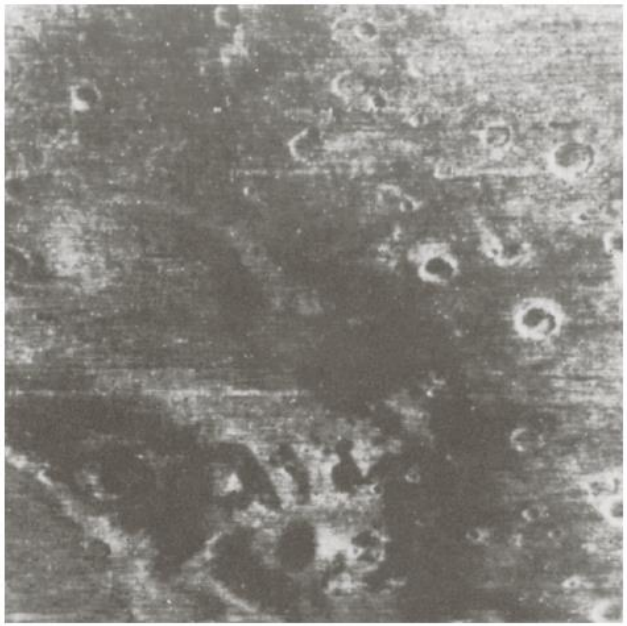
Optimum Notch Filtering Example



$N(u,v)$



$\eta(x,y)$



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

Assignments

- 5.10, 5.11, 5.19, 5.21

- 第2次编程作业

从Laboratory Projects_DIP3E.pdf的Proj04-xx或Proj05-xx中选做1个题目。也可针对DIP4E Chapter 4-5内容，自拟任务。

Assignments

每个编程作业要求递交1份实验报告，命名“学号姓名_prjX.pdf”，内容提纲包括：

- 实验任务： 描述本次实验的任务，即所选择的ProjXX-xx题目，或自拟题目。
- 算法设计：理论上描述所设计的算法。
- 代码实现：描述编程环境，给出自己编写的核心代码。
- 实验结果：描述具体的实验过程，给出每个小实验的输入数据、算法参数和实验结果，并对结果做简要的讨论。
- 总结：简要总结本次实验的技术内容，以及心得体会。