IT4130 – Image Understanding and Processing

Lecture 04 – Spatial Filtering

Recapitulation

Point Processing techniques:

- Image Negatives
- Logarithmic transformation for dynamic range compression
- Power – Law transformation
- Contrast Stretching
- Gray level slicing
- Histogram Based processing techniques
    - Histogram Equalization
    - Histogram Modification
- Image Differencing
- Image Averaging

Image Enhancement

- On completion the students will be able to:

Mask Processing Techniques

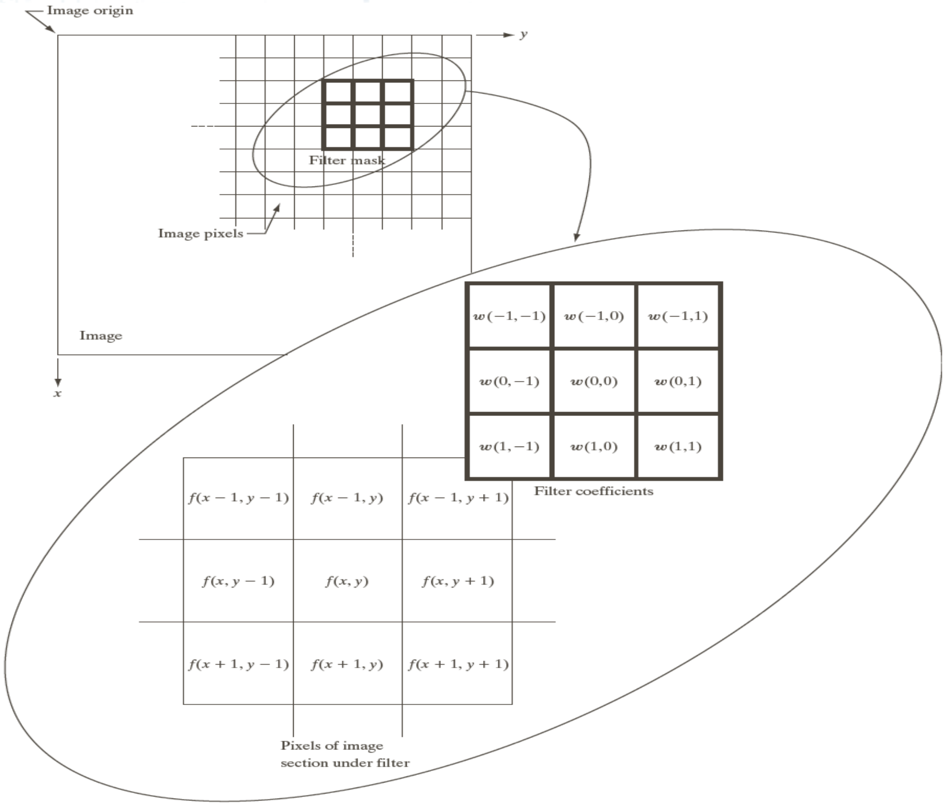
- Linear Smoothing Filter
- Median Filter (nonlinear)
- Sharpening Filter

Neighborhood

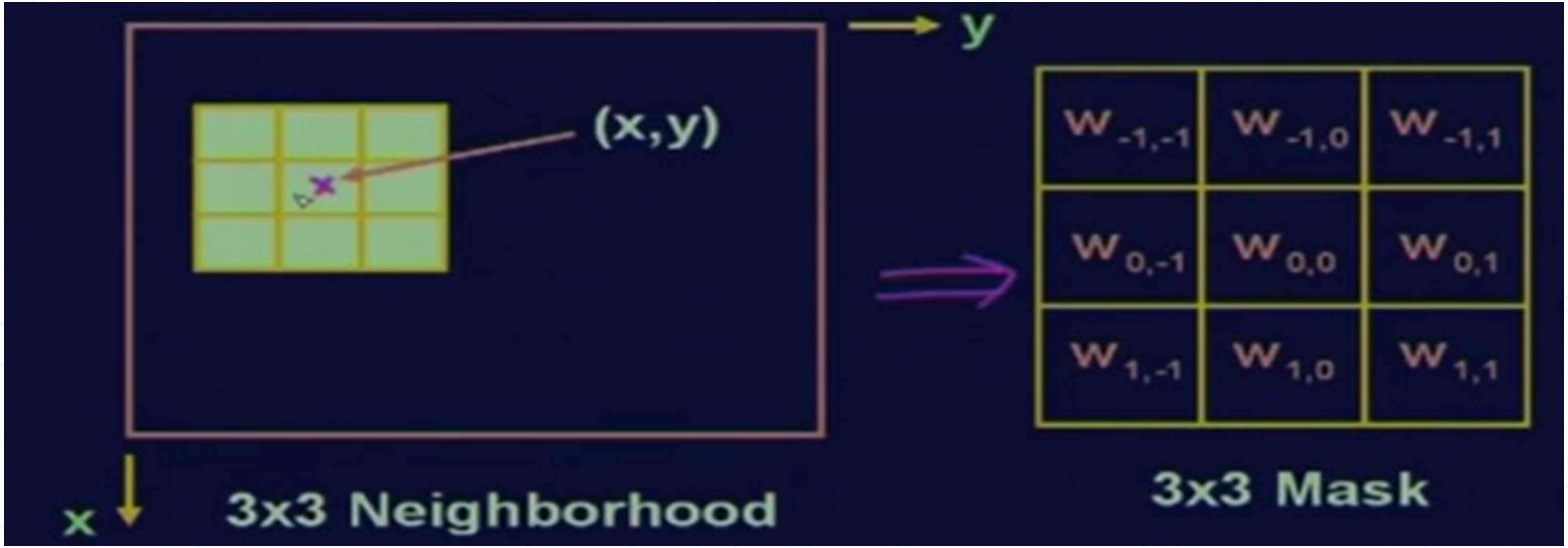
- Arithmetic and logical operations may take place on a subset of the image
  - Typically neighborhood oriented
- Formulated in the context of *mask* operations (also called *template, window or filter* operations)
- Basic concept: let the value of a pixel be a function of its (current) gray level and the gray level of its neighbors (in some sense)

Mask Processing

- Sub image is called *filter*, *mask*, *kernel*, *template* or *window*
- Sub image values are *coefficients*.
- For pixels near the boundary of the image, result may be computed using partial neighborhoods or by padding the input appropriately.



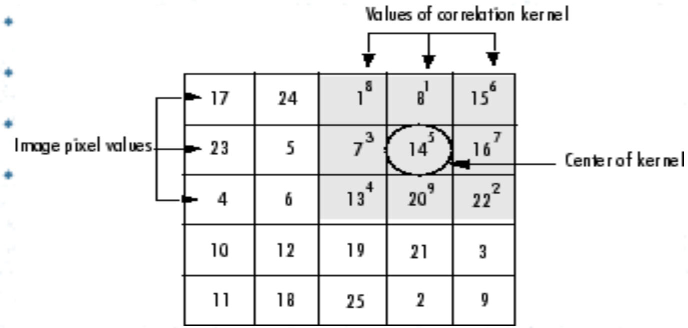
Mask Processing



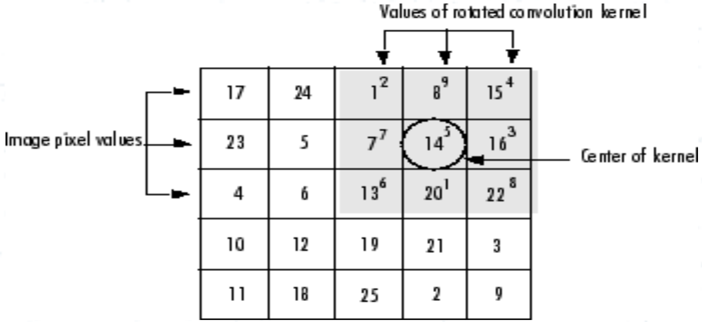
$$g(x,y) = \sum_{i=-1}^{1} \sum_{j=-1}^{1} w_{i,j}\, f(x+i, y+j)$$

Convolution and correlation

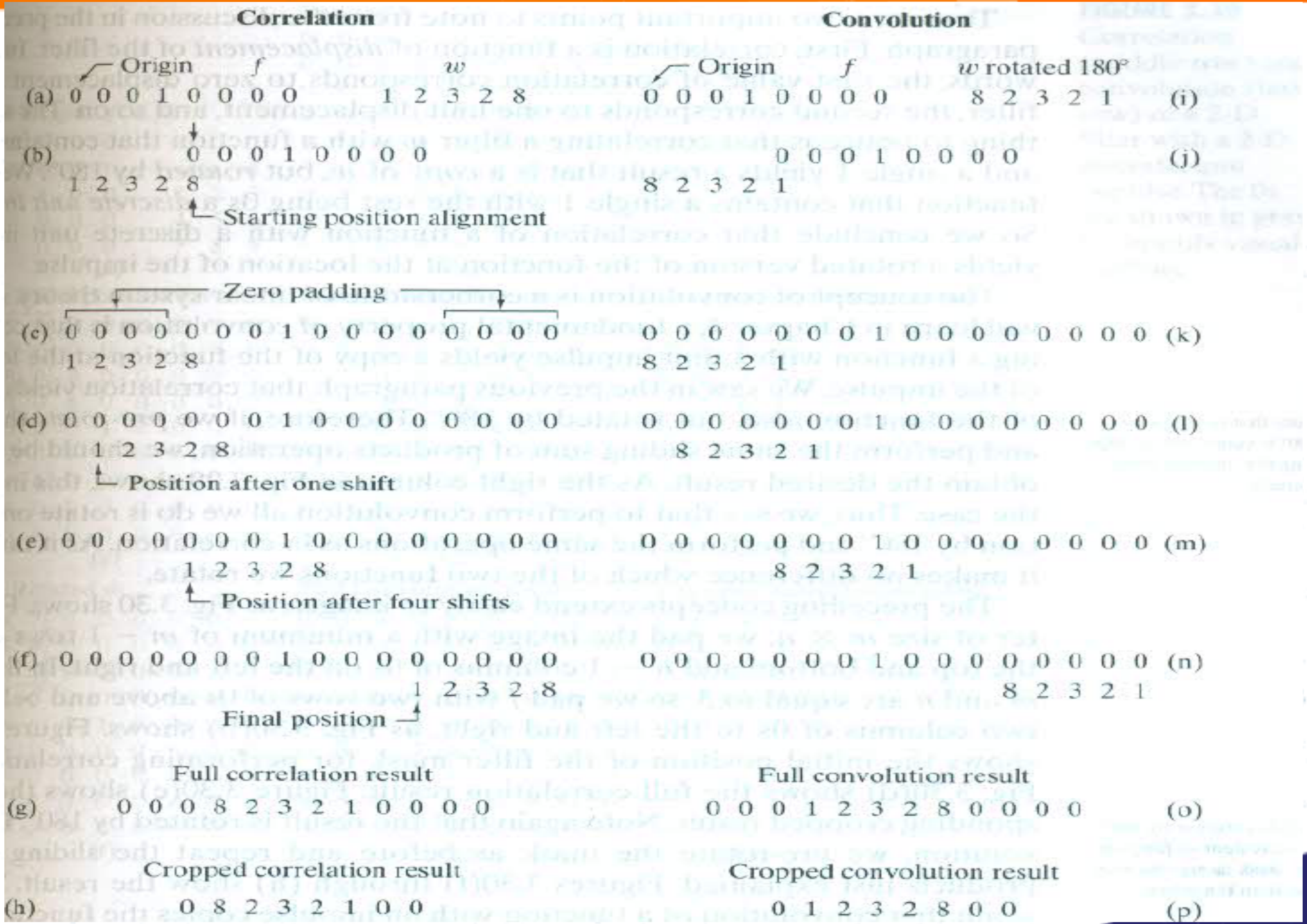
- Correlation
  - In correlation, the value of an output pixel is computed as a weighted sum of neighboring pixels.



- Convolution
  - A convolution kernel is a correlation kernel that has been rotated 180 degrees.



Correlation of a function with a discrete unit impulse yields a rotated version of the function at the impulse location



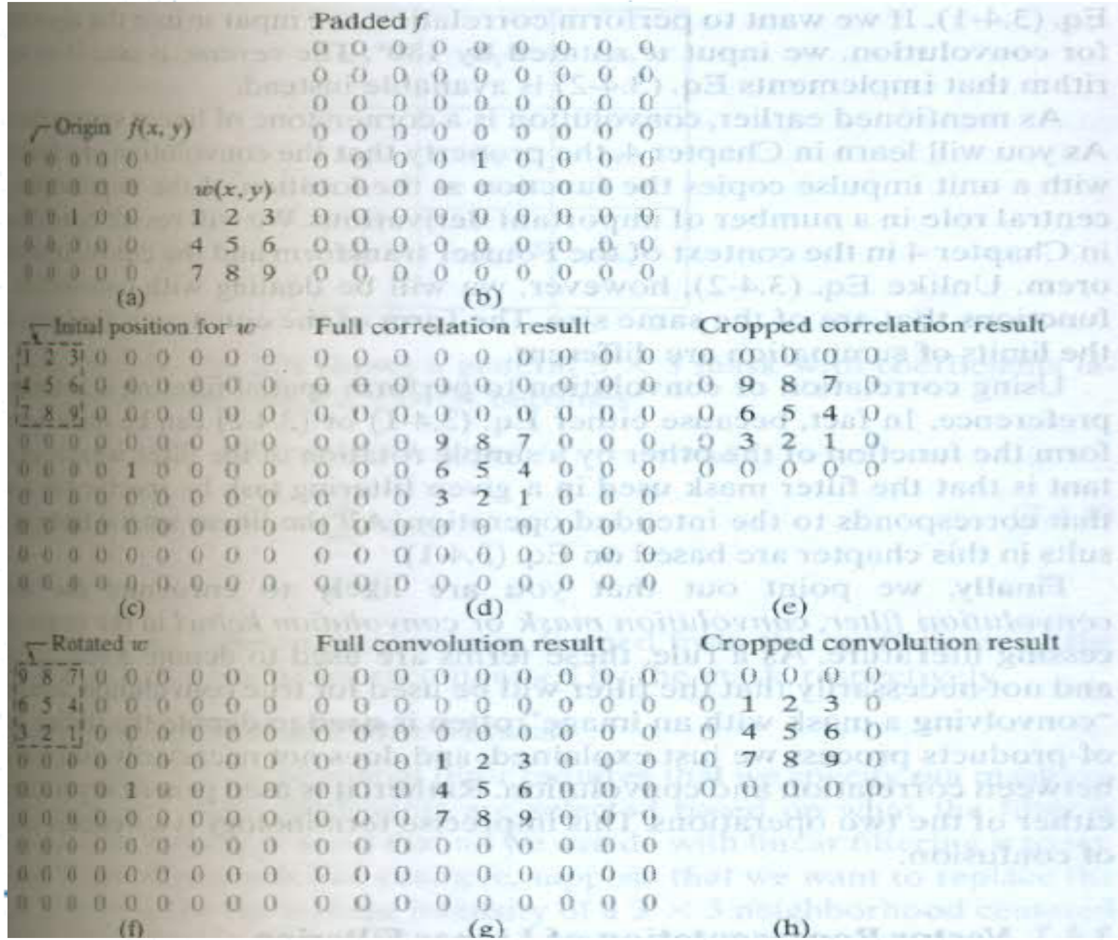


FIGURE 3.30 Correlation (middle row) and convolution (last row) of a 2-D filter with a 2-D discrete, unit impulse. The 0s are shown in gray to simplify visual analysis.

Operator

- Use of spatial masks for filtering is called *spatial filtering*
  - May be **linear** or **nonlinear**
- Linear filters
  - ***Lowpass***: attenuate (or eliminate) high frequency components such as characterized by edges and sharp details in an image
    - Net effect is image blurring
  - ***Highpass***: attenuate (or eliminate) low frequency components such as slowly varying characteristics
    - Net effect is a sharpening of edges and other details
  - ***Bandpass***: attenuate (or eliminate) a given frequency range
    - Used primarily for image restoration (are of little interest for image enhancement)

Non-linear filters

- Nonlinear spatial filters also operate on neighborhoods
- Their operation is based directly on pixel values in the neighborhood under consideration
  - They do not explicitly use coefficient values as in the linear spatial filters
- Example nonlinear spatial filters
  - ***Median filter***: Computes the median gray-level value of the neighborhood. Used for noise reduction.
  - ***Max filter***: Used to find the brightest points in an image
  - ***Min filter***: Used to find the dimmest points in an image

Linear filtering

- Sample neighborhood operation

$$b_5 = \sum_{i=1}^{9} w_i z_i$$

- In generic form

$$g(x, y) = \sum_{m=-a}^{a} \sum_{n=-b}^{b} w(m,n) f(x-m, y-n)$$

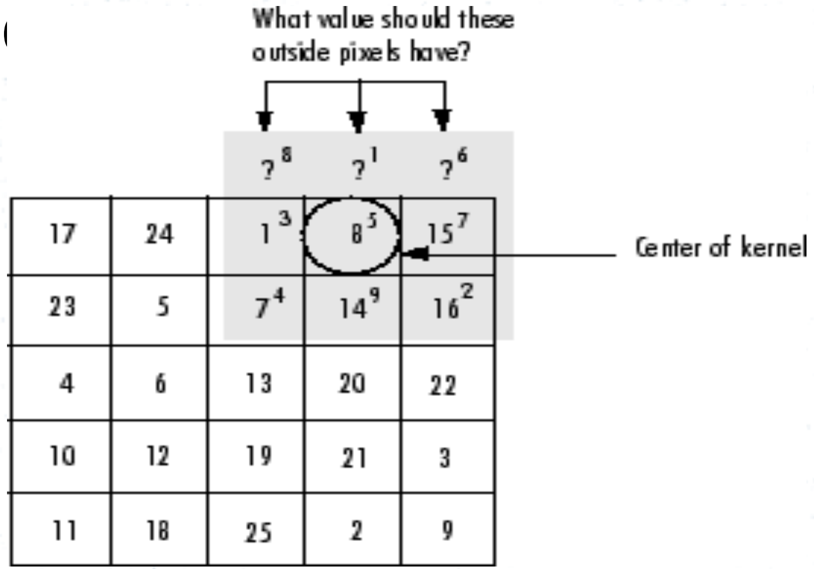
- The operation will be different based on the proper selection of weights.
- Example operations
  - Blurring
  - De-noising
  - Edge detection

Z_1	Z_2	Z_3
Z_4	Z_5	Z_6
Z_7	Z_8	Z_9

W_1	W_2	W_3
W_4	W_5	W_6
W_7	W_8	W_9

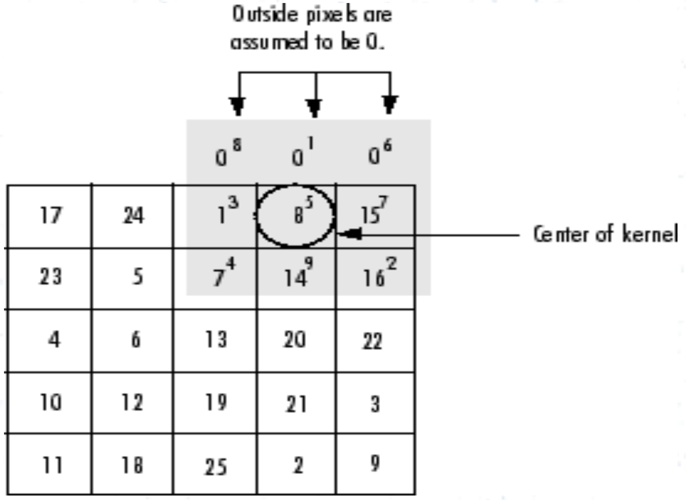
Boundary handling

- When the Values of the Kernel Fall Outside the Imag

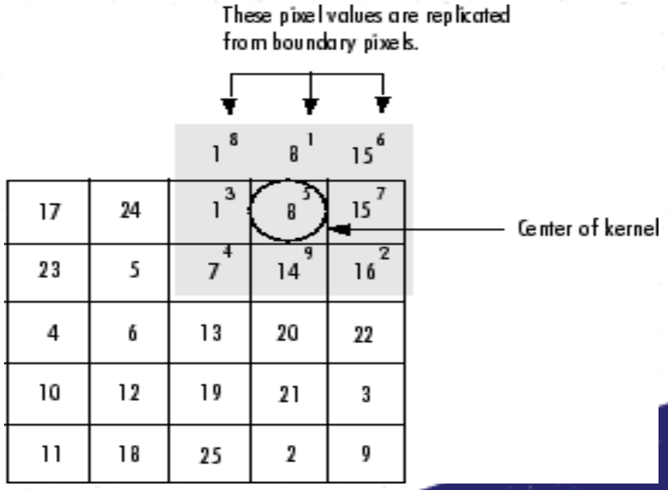


Boundary handling contd.

- Zero Padding of Outside Pixels



- Replicated Boundary Pixels



Lowpass Filter (Averaging Filter)

$$\frac{1}{9} \times \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \longrightarrow \text{Box filter}$$

$$g(x,y) = \frac{1}{9} \sum_{i=-1}^{1} \sum_{j=-1}^{1} f(x+i, y+j)$$

Lowpass - blur

- Using a 5x5 basic low pass filter

$$\frac{1}{25}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Lowpass filters

- Also called as *averaging filters*.
- The shape of the impulse response needed to implement a lowpass (smoothing) filter indicates the filter should have all positive coefficients
- For a 3x3 mask, the simplest arrangement is to have all the coefficient values equal to one (*neighborhood averaging*)
  - scale the result by dividing by 9

Example Filters

a b

FIGURE 3.34 Two 3 × 3 smoothing (averaging) filter masks. The constant multiplier in front of each mask is equal to the sum of the values of its coefficients, as is required to compute an average.

$$\frac{1}{9} \times \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array} \qquad \frac{1}{16} \times \begin{array}{|c|c|c|} \hline 1 & 2 & 1 \\ \hline 2 & 4 & 2 \\ \hline 1 & 2 & 1 \\ \hline \end{array}$$

Applications

- Noise reduction
  - Random noise results in sharp transitions in gray levels.
  - Blur edges
- Reducing irrelevant details
  - Smaller than the filter mask
- Bridging small gaps in lines and curves

Filter size effects

a	b
c	d
e	f

FIGURE 3.35 (a) Original image, of size 500 × 500 pixels. (b)–(f) Results of smoothing with square averaging filter masks of sizes $n = 3, 5, 9, 15,$ and 35, respectively. The black squares at the top are of sizes 3, 5, 9, 15, 25, 35, 45, and 55 pixels, respectively; their borders are 25 pixels apart. The letters at the bottom range in size from 10 to 24 points, in increments of 2 points; the large letter at the top is 60 points. The vertical bars are 5 pixels wide and 100 pixels high; their separation is 20 pixels. The diameter of the circles is 25 pixels, and their borders are 15 pixels apart; their gray levels range from 0% to 100% black in increments of 20%. The background of the image is 10% black. The noisy rectangles are of size 50 × 120 pixels.

- Explain the importance of
  - b and c
  - d
  - e and f

Averaging masks

- Box filter
  - Equal coefficients
- Weighted average
  - Distance based
  - $g(x, y) = \frac{\sum_{n=0}^{N-1} \sum_{m=0}^{M-1} w(m,n) f(x-m, y-n)}{\sum_{n=0}^{N-1} \sum_{m=0}^{M-1} w(m,n)}$

Example - Averaging

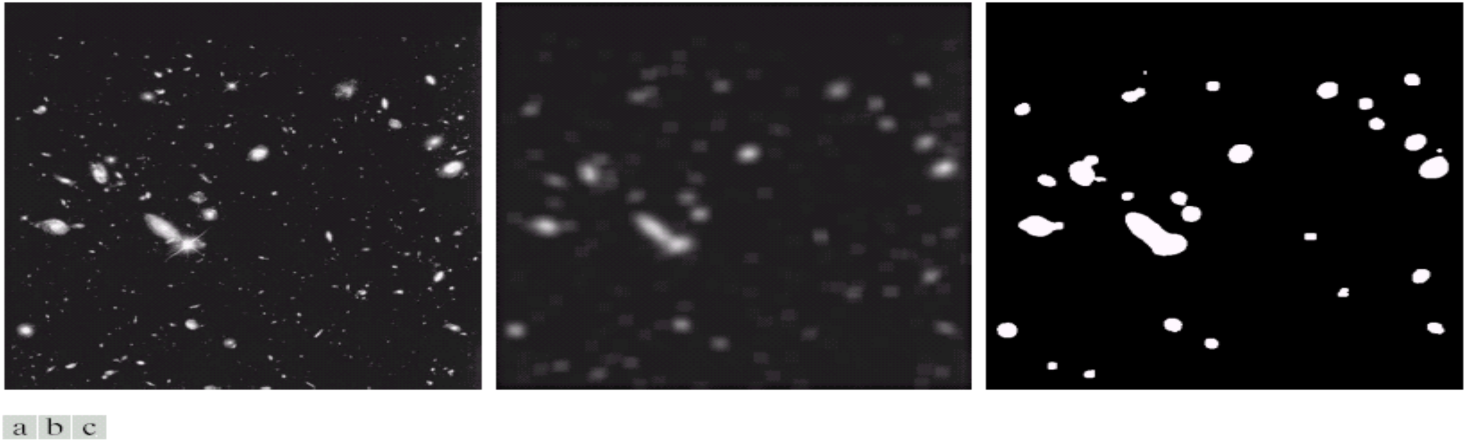


FIGURE 3.36 (a) Image from the Hubble Space Telescope. (b) Image processed by a 15 × 15 averaging mask. (c) Result of thresholding (b). (Original image courtesy of NASA.)

Highpass - Edge detection

- A sample image filtered using a 3x3 basic high pass

$$\frac{1}{9}\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Non-linear filters

- Their operation is based directly on pixel values in the neighborhood under consideration
  - They do not explicitly use coefficient values as in the linear spatial filters
- For example, Order-Statistics filters are spatial filters whose response is based on ordering (ranking) the pixels contained in the image area encompassed by the filter

Order-Statistics Filters

- Median filter
  - Computes the median gray-level value of the neighborhood. Used for noise reduction.
- Max filter
  - Used to find the brightest points in an image
    - $R = \max\{z_k \mid k = 1,2,\ldots,9\}$
- *Min filter*
  - Used to find the dimmest points in an image
    - $R = \min\{z_k \mid k = 1,2,\ldots,9\}$
- Midpoint filter
- Alpha-trimmed mean filter

Median filter

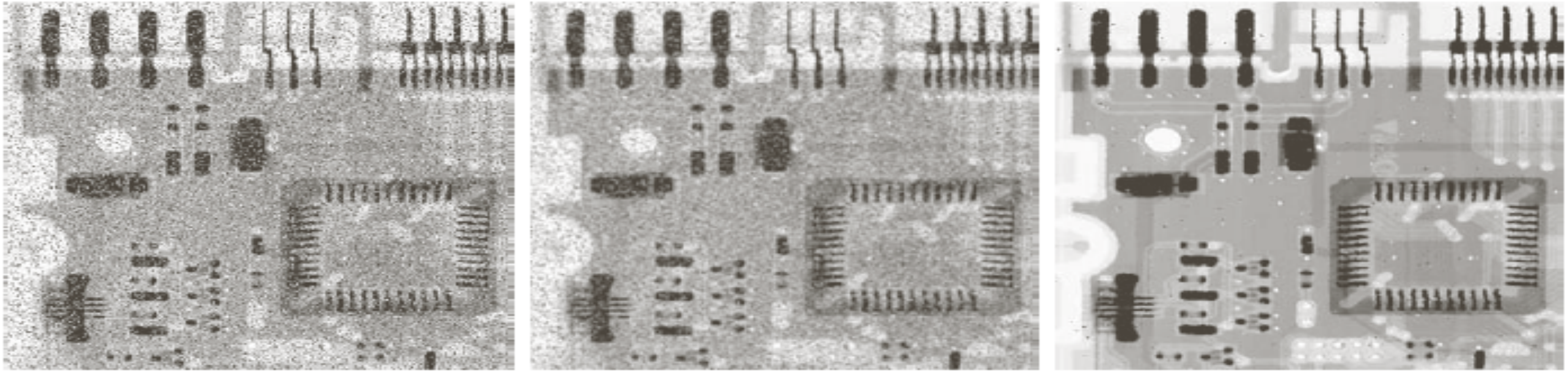
- Process is replaces the value of a pixel by the median of the gray levels in region A_{xy} of that pixel:

$$s(x, y) = \underset{(i,j)\in A_{xy}}{median}\{r(i, j)\}$$

Smoothing with median

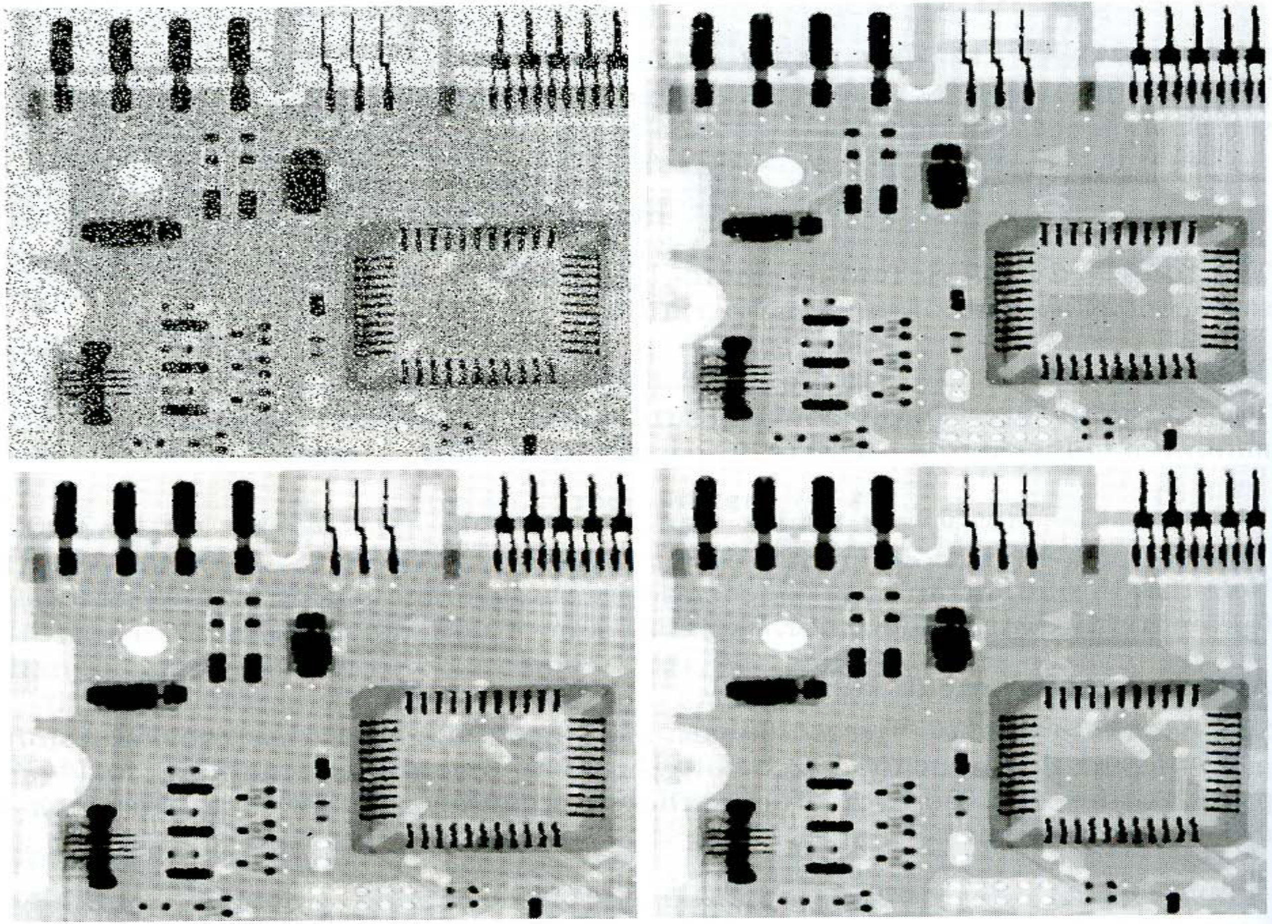
- Smoothing
  - Averaging
    - One problem with the lowpass filter is it blurs edges and other sharp details.
  - Median
- If the intent is to achieve noise reduction, one approach can be to use *median filtering*
  - The value of each pixel is replaced by the median pixel value in the neighborhood (as opposed to the average)
  - Particularly effective when the noise consists of strong, spike like components (impulse noise/ salt and pepper noise) and edge sharpness is to be preserved
- The median *m* of a set of values is such that half of the values are greater than *m* and half are less than *m*
- To implement, sort the pixel values in the neighborhood, choose the median and assign this value to the pixel of interest
- Forces pixels with distinct intensities to be more like their neighbors

Example - Median filtering



a b c

FIGURE 3.35 (a) X-ray image of circuit board corrupted by salt-and-pepper noise. (b) Noise reduction with a 3×3 averaging mask. (c) Noise reduction with a 3×3 median filter. (Original image courtesy of Mr. Joseph E. Pascente, Lixi, Inc.)



a b
c d

FIGURE 5.10
(a) Image corrupted by salt-and-pepper noise with probabilities $P_a = P_b = 0.1$. (b) Result of one pass with a median filter of size 3×3. (c) Result of processing (b) with this filter. (d) Result of processing (c) with the same filter.

Max and Min filter

- Using the 100th percentile results in the so-called max filter, given by

$$s(x, y) = \max_{(i,j)\in A_{xy}} \{r(i, j)\}$$

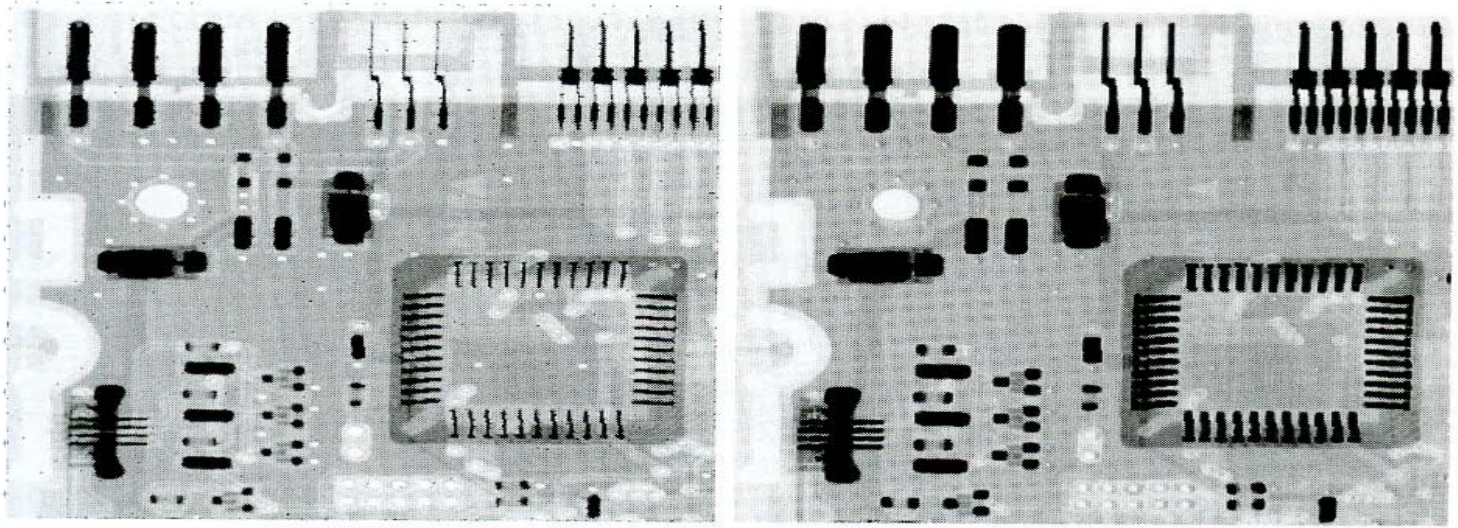
This filter is useful for finding the brightest points in an image. Since pepper noise has very low values, it is reduced by this filter as a result of the max selection processing the subimage area A_{xy}.

- The 0th percentile filter is min filter:

$$s(x, y) = \min_{(i,j)\in A_{xy}} \{r(i, j)\}$$

This filter is useful for finding the darkest points in an image. Also, it reduces salt noise as a result of the min operation.

Example – Min filtering



a b

FIGURE 5.11 (a) Result of filtering Fig. 5.8(a) with a max filter of size 3 × 3. (b) Result of filtering 5.8(b) with a min filter of the same size.

Midpoint filter

- The midpoint filter simply computes the midpoint between the maximum and minimum values in the area encompassed by the filter:

$$s(x,y) = \frac{1}{2}\left[\max_{(i,j)\in A_{xy}}\{r(i,j)\} + \min_{(i,j)\in A_{xy}}\{r(i,j)\}\right]$$

Note: This filter works best for randomly distributed noise, like Gaussian or uniform noise.

Alpha-trimmed mean filter

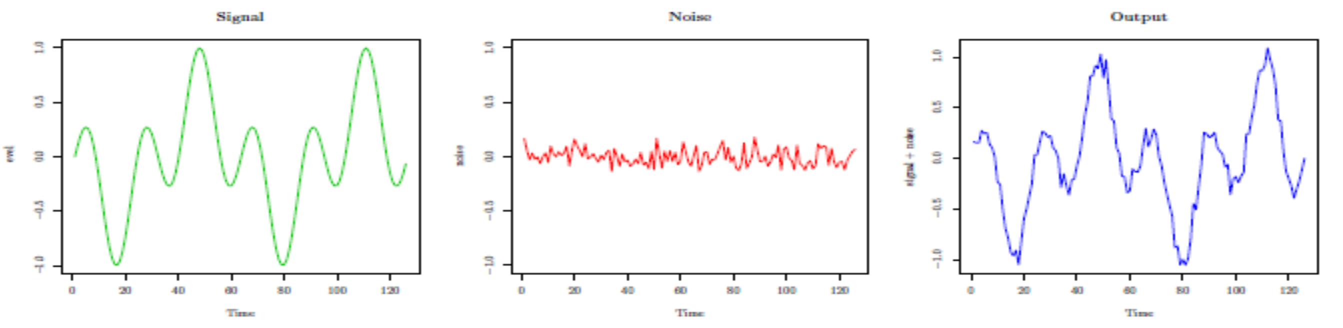
- Suppose that we delete the d/2 lowest and the d/2 highest gray-level values of r(i,j) in the area A_{xy}.
- Let $r_k(i,j)$ represent the remaining mn-d pixels. And averaging these remain pixels is denoted as:

$$s(x,y) = \frac{1}{mn-d} \sum_{(i,j)\in A_{xy}} r_k(i,j)$$

Where the value of d can range from 0 to mn-1. When d=0, It is arithmetic mean filter and d=(mn-1)/2 is a median filter. It is useful for the multiple types of noise such as the combination of salt-and-pepper and Gaussian noise.

Noise

- Any real world sensor is affected by a certain degree of noise, whether it is thermal, electrical or otherwise.
- S



Noise (Contd.)

- Gaussian Noise

$\sigma = 10$

$\sigma = 20$

$\sigma = 50$

Salt and Pepper Noise

p=0.01

p=0.05

p=0.1

Mean Filters

- These are simple methods to reduce noise in spatial domain.
  - Arithmetic mean filter
  - Geometric mean filter
  - Harmonic mean filter
  - Contraharmonic mean filter
- Let A_{xy} represent the set of coordinates in a rectangular subimage window of size mxn, centered at point (x,y).

ref(1) pages 231-233

Arithmetic mean filter

- Compute the average value of the corrupted image r(i,j) in the area defined by $A_{x,y}$.
- The value of the restored image s at any point (x,y)

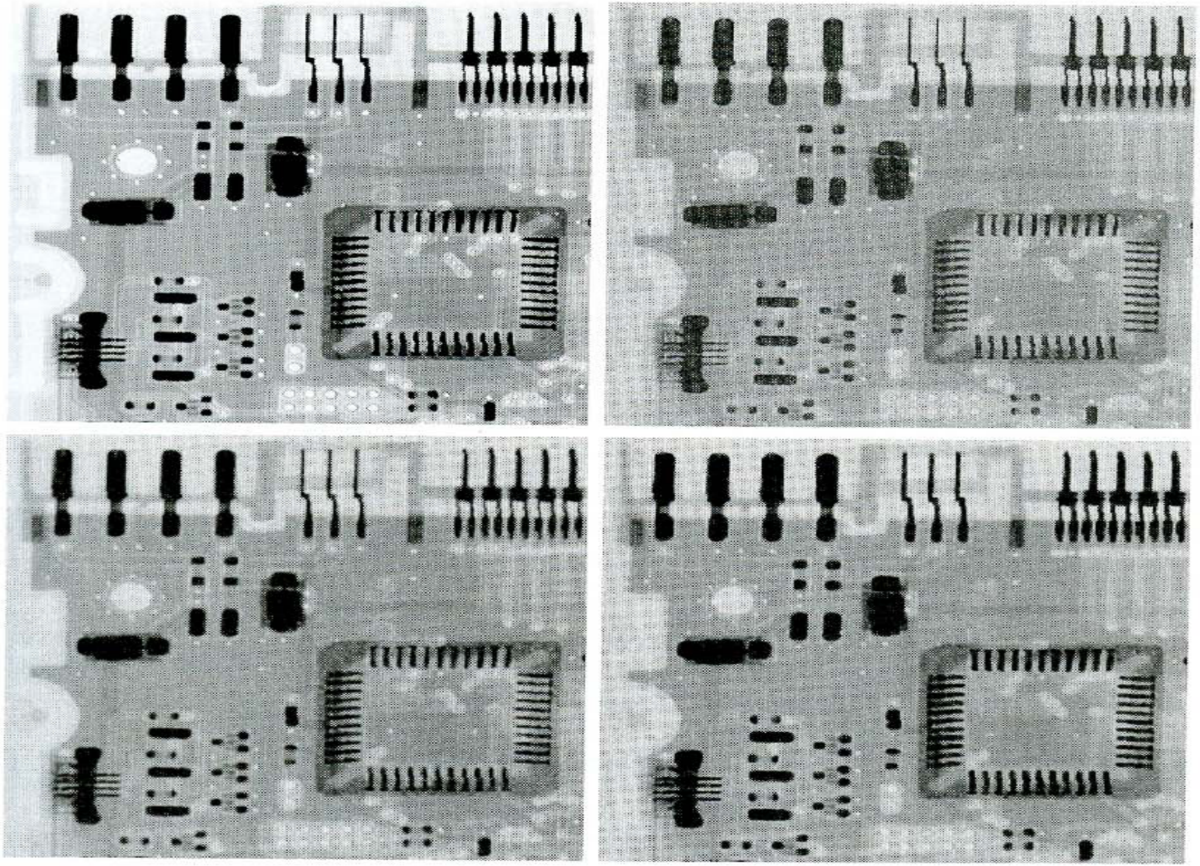
$$s(x,y) = \frac{1}{mn} \sum_{(i,j) \in A_{x,y}} r(i,j)$$

Note: Using a convolution mask in which all coefficients have value 1/mn. Noise is reduced as a result of blurring.

Geometric mean filter

- Using a geometric mean filter is given by the expression

$$s(x,y) = \left[\prod_{(i,j)\in A_{xy}} r(i,j) \right]^{\frac{1}{mn}}$$



a	b
c	d

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

Harmonic mean filter

- The harmonic mean filter operation is given by the expression

$$s(x,y) = \frac{mn}{\sum_{(i,j)\in A_{xy}} \frac{1}{r(i,j)}}$$

Contraharmonic mean filter

- The contraharmonic mean filter operation is given by the expression

$$s(x,y) = \frac{\sum_{(i,j)\in A_{xy}} r(i,j)^{Q+1}}{\sum_{(i,j)\in A_{xy}} r(i,j)^{Q}}$$

Where Q is called the order of the filter. This filter is well suited for reducing or virtually eliminating the effects of salt-and-pepper noise.

a b
c d

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