Image Filtering

Two views of filtering

- Image filters in **spatial** domain
  - Filter is a mathematical operation of a grid of numbers
  - Smoothing, sharpening, measuring texture
- Image filters in the **frequency** domain
  - Filtering is a way to modify the frequencies of images
  - Denoising, sampling, image compression

Image filtering (spatial domain)

- Image filtering: compute function of local neighborhood at each position

- Really important!
  - Enhance images
    - Denoise, resize, increase contrast, etc.
  - Extract information from images
    - Texture, edges, distinctive points, etc.
  - Detect patterns
    - Template matching

Filter

- The name ***filter*** is borrowed from frequency domain processing, where "*filtering*" refers to accepting (passing) or rejecting certain frequency components.
  - A filter that passes low frequencies is called a *lowpass* filter
  - The lowpass effect is to **blur** (**smooth**) an image.
- If the operation performed on the image pixels is *linear*, then the filter is called a ***linear spatial filter***
- Otherwise, the filter is ***nonlinear***

- Image mask is a linear spatial filter

- For a mask of size m*n, we assume
  - m=2a+1
  - n=2b+1 (a, b >0)
- Our focus on filters of odd size with the smallest size 3x3

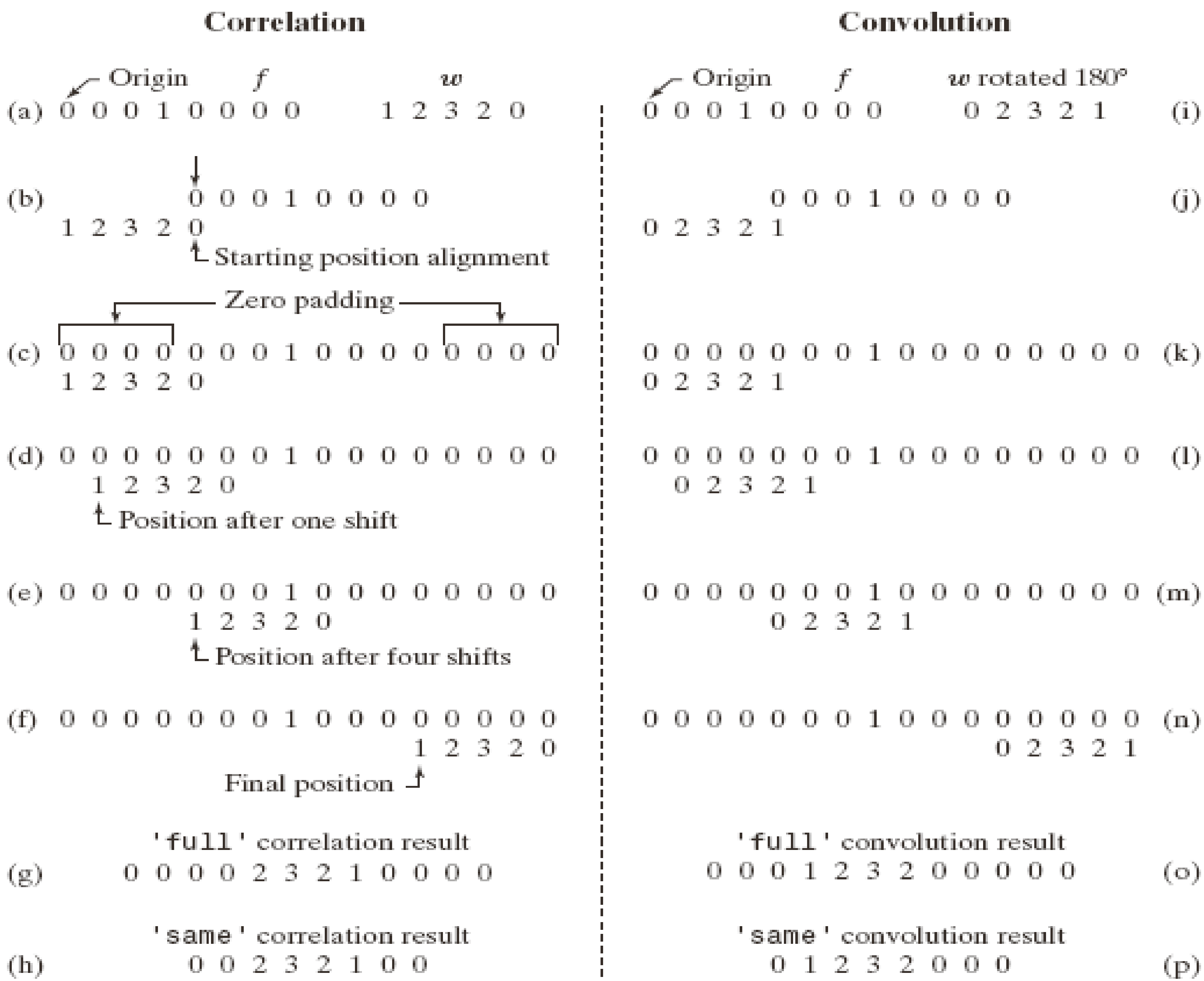
Spatial filtering expression:

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

Spatial Correlation and Convolution(相關與卷積）

- **Correlation**: the process of moving a filter mask over the image and computing the sum of products at each location.
- **Convolution**: the filter is first rotated by 180°

  - Each pixel in filter ω visits every pixel in image f
  - If there are parts of the functions that do not overlap, the solution is to pad f with enough 0s on each side to allow each pixel in ω to visit every pixel in f


Correlation
Convolution
Origin f w
(a) 0 0 0 1 0 0 0 0 1 2 3 2 0
(b) 0 0 0 1 0 0 0 0
1 2 3 2 0
Starting position alignment
Zero padding
(c) 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
1 2 3 2 0
(d) 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
1 2 3 2 0
Position after one shift
(e) 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
1 2 3 2 0
Position after four shifts
(f) 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
1 2 3 2 0
Final position
'full' correlation result
(g) 0 0 0 0 2 3 2 1 0 0 0 0
'same' correlation result
(h) 0 0 2 3 2 1 0 0
Origin f w rotated 180°
0 0 0 1 0 0 0 0 0 2 3 2 1 (i)
0 0 0 1 0 0 0 0 (j)
0 2 3 2 1
0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 (k)
0 2 3 2 1
0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 (l)
0 2 3 2 1
0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 (m)
0 2 3 2 1
0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 (n)
0 2 3 2 1
'full' convolution result
0 0 0 1 2 3 2 0 0 0 0 0 (o)
'same' convolution result
0 1 2 3 2 0 0 0 (p)

- A function that contains a single 1 with the rest being 0s is called a ***discrete unit impulse***
- Correlation of a function with a discrete unit impulse yields a rotated version of the function at the location of the impulse
- Convolution of a function with a discrete unit impulse yields a copy of the function at the location of the impulse
- We pre-rotate the filter to obtain the desired result

Origin of $f(x, y)$

0	0	0	0	0
0	0	0	0	0
0	0	1	0	0
0	0	0	0	0
0	0	0	0	0

$w(x, y)$

1	2	3
4	5	6
7	8	9

(a)

Padded f

0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0

(b)

Initial position for w

1	2	3	0	0	0	0	0	0
4	5	6	0	0	0	0	0	0
7	8	9	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0

(c)

`'full'` correlation result

0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	9	8	7	0	0	0
0	0	0	6	5	4	0	0	0
0	0	0	3	2	1	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0

(d)

`'same'` correlation result

0	0	0	0	0
0	9	8	7	0
0	6	5	4	0
0	3	2	1	0
0	0	0	0	0

(e)

Rotated w

9	8	7	0	0	0	0	0	0
6	5	4	0	0	0	0	0	0
3	2	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0

(f)

`'full'` convolution result

0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	1	2	3	0	0	0
0	0	0	4	5	6	0	0	0
0	0	0	7	8	9	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0

(g)

`'same'` convolution result

0	0	0	0	0
0	1	2	3	0
0	4	5	6	0
0	7	8	9	0
0	0	0	0	0

(h)

- **Convolution**
  - A pixel’s value is computed from its old value and the values of pixels in its vicinity （鄰近地區）.
  - More costly operations than simple point processes, but more powerful

- Mathematical definition of -discrete- **convolution**:

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

(for a mask with odd dimensions)

Correlation

- Correlation translates the mask directly to the image without flipping it.
- If the mask is symmetric (i.e., the flipped mask is the same as the original one) then the results of convolution and correlation are the same.

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

- **Non-linear filtering**
  - Linear filters have the property that the output is a linear combination of the inputs.
  - Filters which do not satisfy this property are called non-linear.

- **Normalization of mask weights**
  - The sum of weights in the convolution mask affect the overall intensity of the resulting image.
  - Many convolution masks have coefficients that sum to 1 (the convolved image will have the same average intensity as the original one).

- Some masks have negative weights and sum to 0.
- Pixels with negative values may be generated using masks with negative weights.
- Negative values are mapped to the positive range through appropriate normalization.

Generating spatial filter masks

- Generating an *m*n* linear spatial filter requires that we specify *mn* mask coefficients
- These coefficients are selected based on what the filter is supposed to do
  - All we can do with linear filtering is to implement a sum of products

Smoothing (or Low-pass) filters

- Useful for noise reduction and image blurring.
- It removes the finer details of an image.

- The idea behind smoothing filters
  - Replacing the value of every pixel in an image by the average of the intensity levels in the neighborhood defined by the filter mask, this process results in an image with reduced “sharp” transitions in intensities
  - ***Random noise*** typically consists of sharp transitions in intensity levels, the most obvious application of smoothing is noise reduction
  - ***Edges*** also are characterized by sharp intensity transitions, so averaging filters have the undesirable side effect that they blur edges

Some popular filters

1. Averaging or Mean filter

- The elements of the mask must be positive.
- The size of the mask determines the degree of smoothing.

$$\frac{1}{9} \times \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

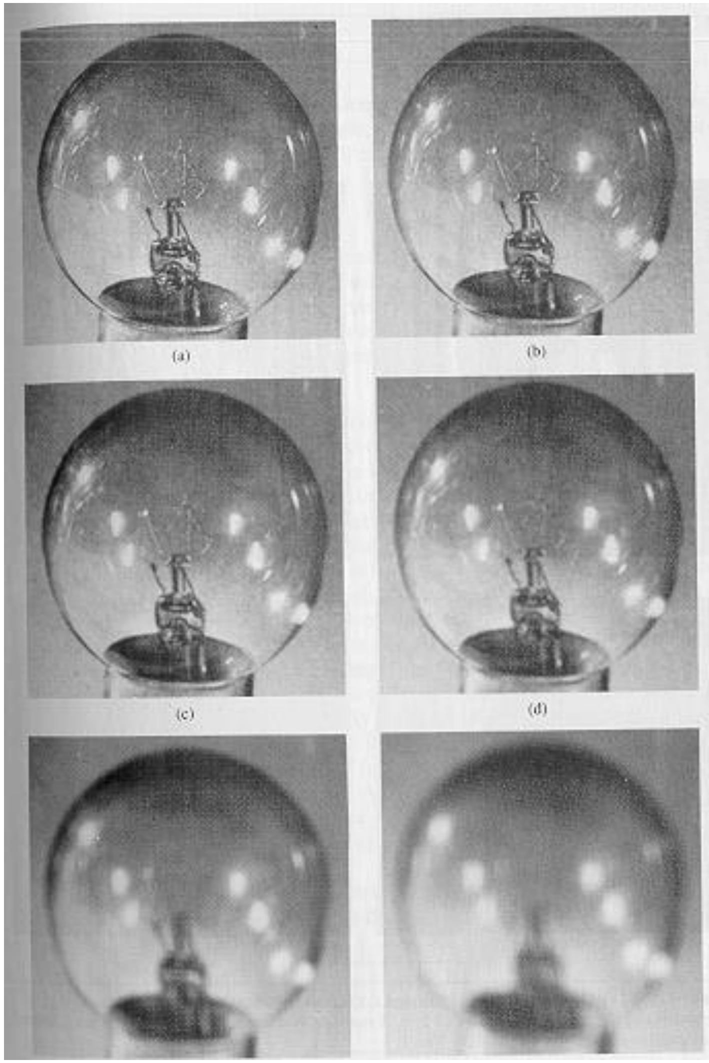
(a)

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

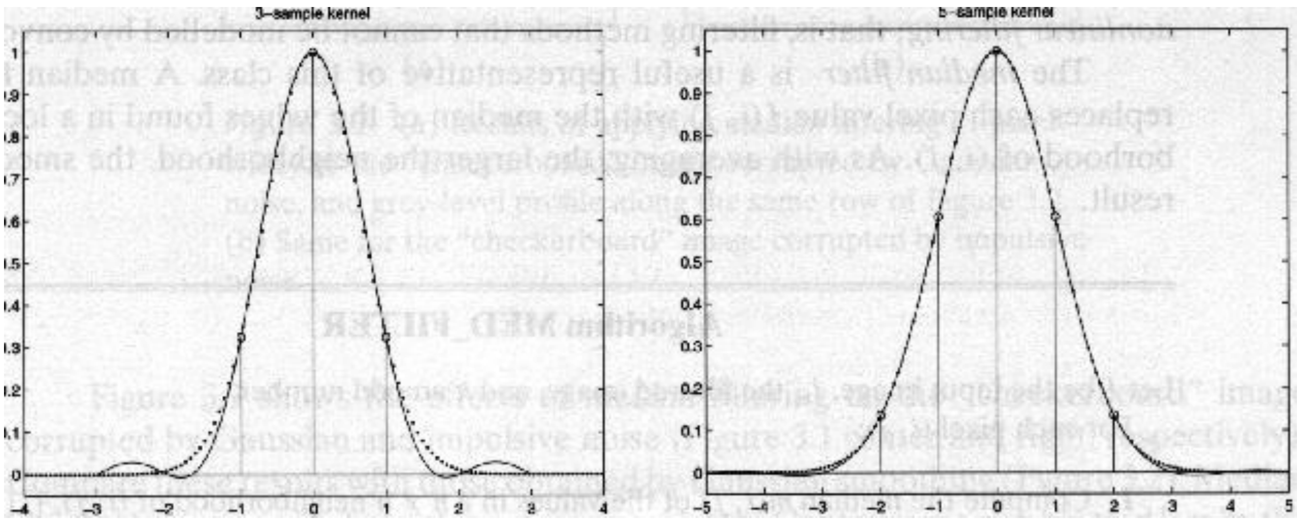
(b)

$$\frac{1}{49} \times \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

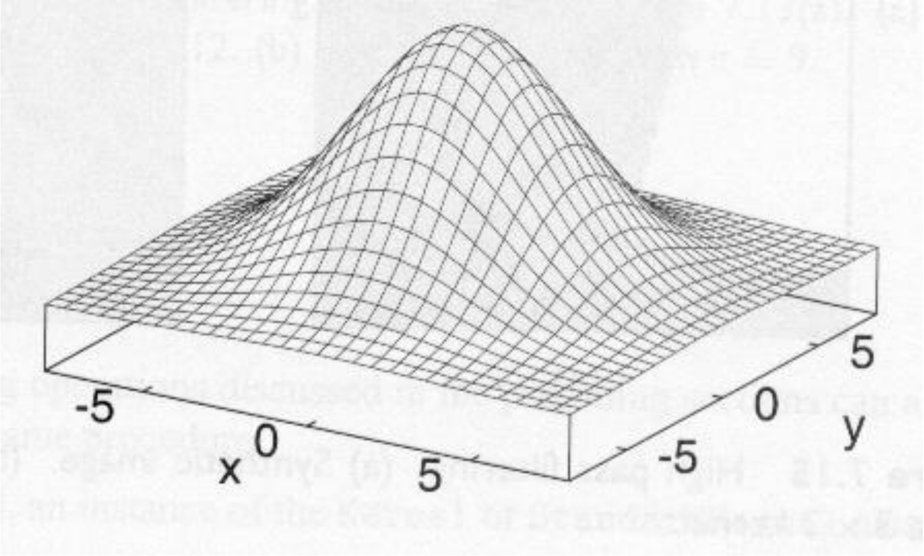
(c)



- **Gaussian (linear filter)**
  - The weights are samples from a Gaussian function.



(example using 1D Gaussian)



$$h(x, y) = e^{\left[\frac{-(x^2 + y^2)}{2\sigma^2}\right]}$$

- The Gaussian’s mask weights fall off to (almost) zero at the mask’s edges.
- Gaussian smoothing can be implemented efficiently thanks to the fact that the kernel is separable.

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

$$=\sum_{s=-a}^{a}\sum_{t=-b}^{b}e^{[\frac{-(s^2+t^2)}{2\sigma^2}]}f(x-s,y-t)$$

$$=\sum_{s=-a}^{a}e^{[\frac{-s^2}{2\sigma^2}]}\sum_{t=-b}^{b}e^{[\frac{-t^2}{2\sigma^2}]}f(x-s,y-t)$$

Separability example

2D convolution
(center location only)

$$\begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 2 & 4 & 2 \\ \hline 1 & 2 & 1 \\ \hline \end{array} * \begin{array}{|c|c|c|}\hline 2 & 3 & 3 \\ \hline 3 & 5 & 5 \\ \hline 4 & 4 & 6 \\ \hline \end{array}$$

The filter factors
into a product of 1D
filters:

$$\begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 2 & 4 & 2 \\ \hline 1 & 2 & 1 \\ \hline \end{array} = \begin{array}{|c|}\hline 1 \\ \hline 2 \\ \hline 1 \\ \hline \end{array} \times \begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline \end{array}$$

Perform convolution
along rows:

$$\begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline \end{array} * \begin{array}{|c|c|c|}\hline 2 & 3 & 3 \\ \hline 3 & 5 & 5 \\ \hline 4 & 4 & 6 \\ \hline \end{array} = \begin{array}{|c|c|c|}\hline & 11 & \\ \hline & 18 & \\ \hline & 18 & \\ \hline \end{array}$$

Followed by convolution
along the remaining column:

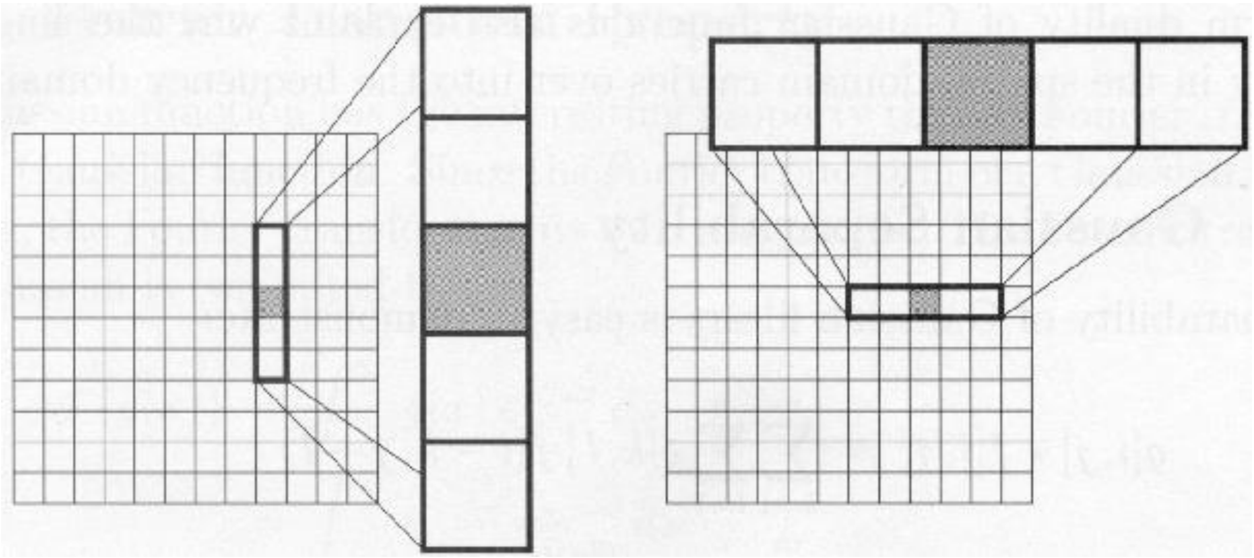
Algorithm

- To convolve an image *I* with a *n* * *n* 2D Gaussian mask *G* with $\sigma = \sigma_g$

  1. Build a 1-D Gaussian mask ***g***, of width ***n***, with $\sigma_g = \sigma_G$

  2. Convolve each column of ***I*** with ***g***, yielding a new image ***Ic***

  3. Convolve each row of ***Ic*** with ***g***



- The value of σ determines the degree of smoothing.
- As σ increases, the size of the mask must also increase if we are to sample the Gaussian satisfactorily.

7 × 7 Gaussian mask

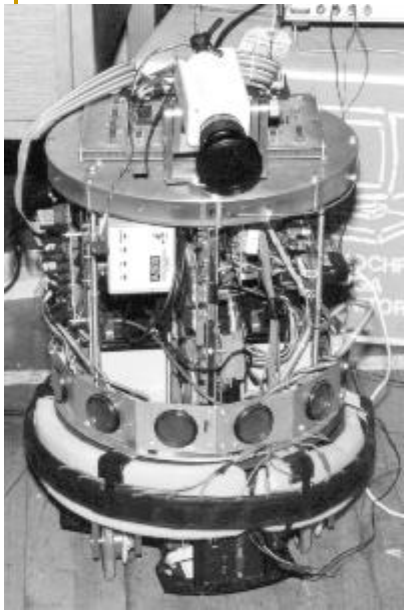
1	1	2	2	2	1	1
1	2	2	4	2	2	1
2	2	4	8	4	2	2
2	4	8	16	8	4	2
2	2	4	8	4	2	2
1	2	2	4	2	2	1
1	1	2	2	2	1	1

$\sigma = 1$

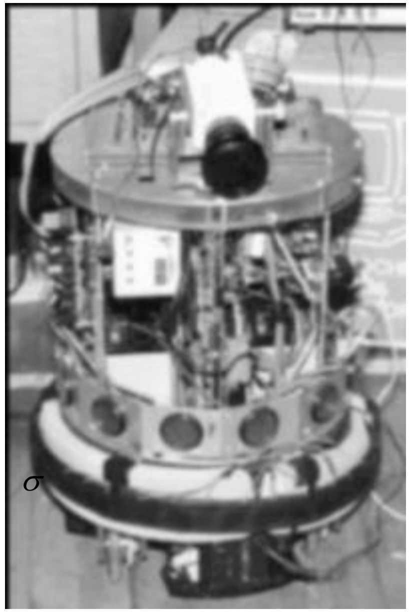
15 × 15 Gaussian mask

2	2	3	4	5	5	6	6	6	5	5	4	3	2	2
2	3	4	5	7	7	8	8	8	7	7	5	4	3	2
3	4	6	7	9	10	10	11	10	10	9	7	6	4	3
4	5	7	9	10	12	13	13	13	12	10	9	7	5	4
5	7	9	11	13	14	15	16	15	14	13	11	9	7	5
5	7	10	12	14	16	17	18	17	16	14	12	10	7	5
6	8	10	13	15	17	19	19	19	17	15	13	10	8	6
6	8	11	13	16	18	19	20	19	18	16	13	11	8	6
6	8	10	13	15	17	19	19	19	17	15	13	10	8	6
5	7	10	12	14	16	17	18	17	16	14	12	10	7	5
5	7	9	11	13	14	15	16	15	14	13	11	9	7	5
4	5	7	9	10	12	13	13	13	12	10	9	7	5	4
3	4	6	7	9	10	10	11	10	10	9	7	6	4	3
2	3	4	5	7	7	8	8	8	7	7	5	4	3	2
2	2	3	4	5	5	6	6	6	5	5	4	3	2	2

$\sigma = 3$



Original



σ = 1.0 (and kernel size 5×5).

σ = 2.0 (and kernel size 9×9).

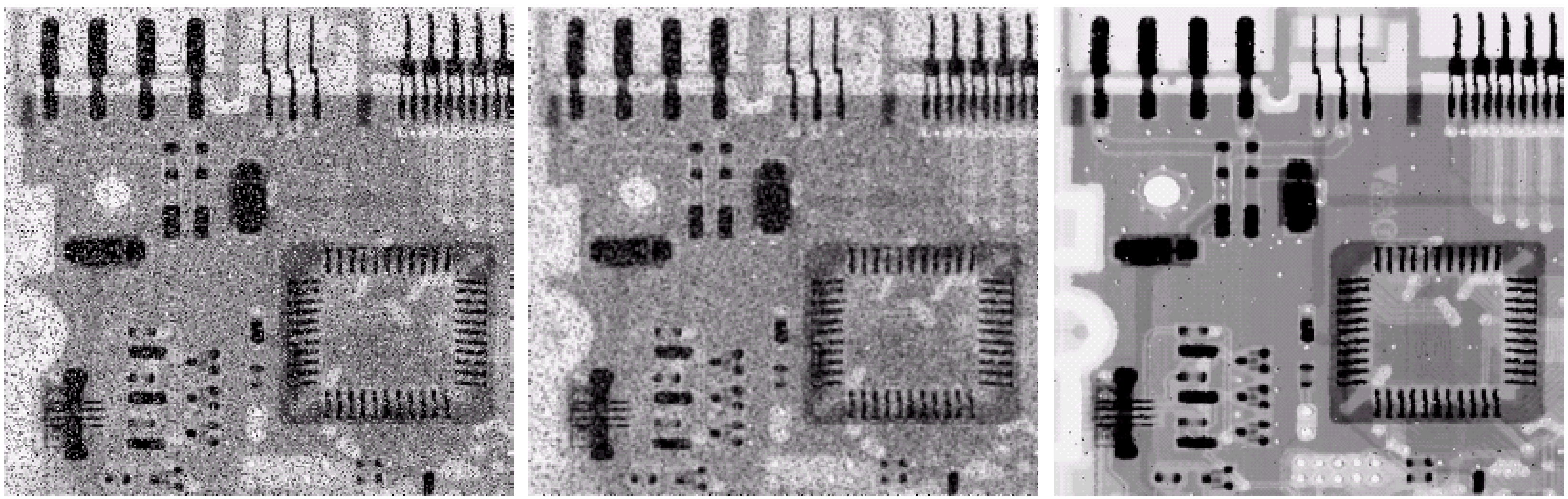
σ = 4.0 (and kernel size 15×15).

Order-statistic (non-linear) filters

- **Order-statistic filter** are nonlinear spatial filters whose response is based on ordering (ranking) the pixels contained in the image area encompassed by the filter, and then replacing the value determined by ranking result

2.Median filter (non-linear)

- Effective for removing "salt and pepper" noise (random occurrences of black and white pixels).



a b c

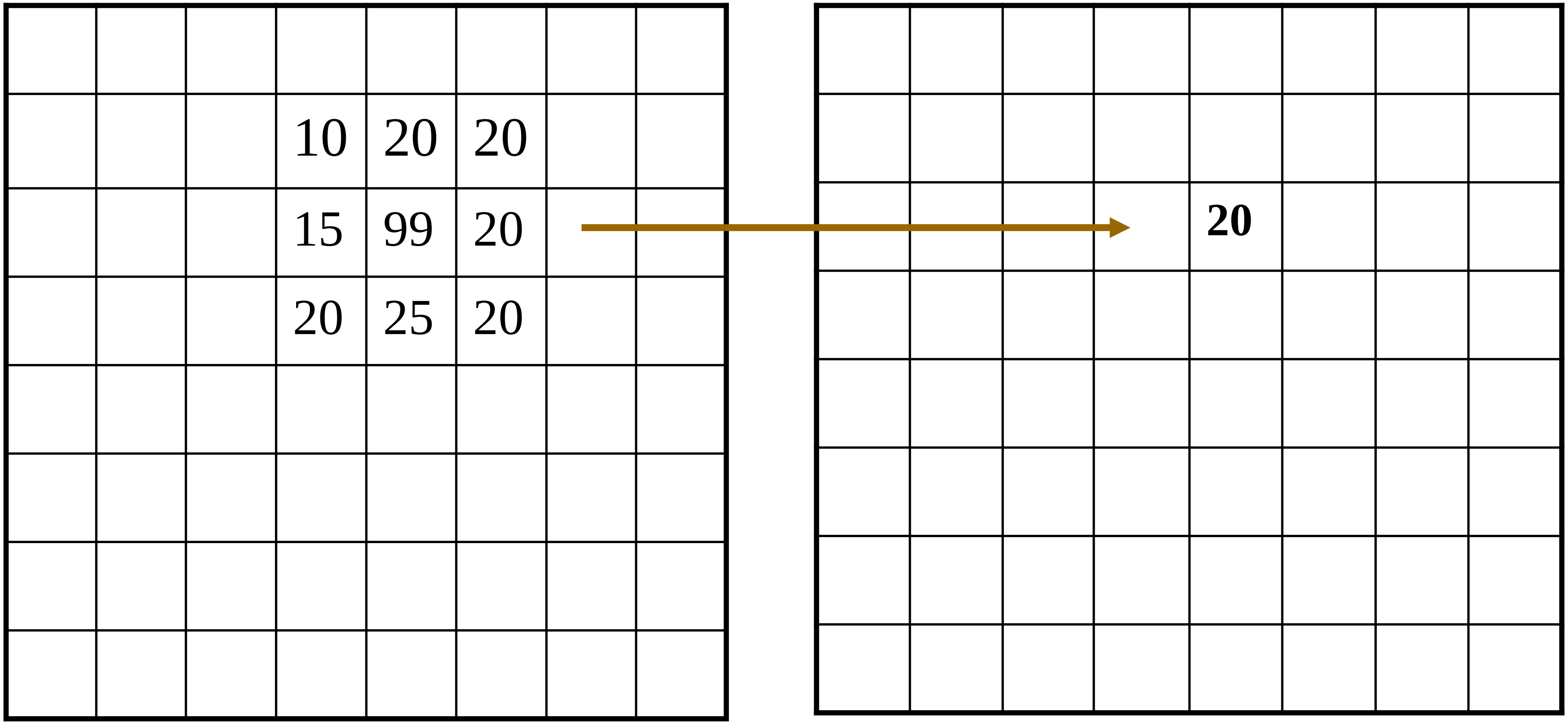
FIGURE 3.37 (a) X-ray image of circuit board corrupted by salt-and-pepper noise. (b) Noise reduction with a 3 × 3 averaging mask. (c) Noise reduction with a 3 × 3 median filter. (Original image courtesy of Mr. Joseph E. Pascente, Lixi, Inc.)

Original and with salt & pepper noise
imnoise(image, .salt & pepper.);

Local averaging
K=filter2(fspecial(.average.,3),image)/255.

Median filtered
L=medfil2(image, [3 3]);

- Replace each pixel value by the median of the gray-levels in the neighborhood of the pixels



10 20 20 15 99 20 20 15 20

Sort

10 15 20 20 20 20 20 20 99

median

- Advantage
  - Provide excellent noise-reduction capabilities, with considerably less blurring than linear smoothing filter of similar size

Summarize of smoothing (low pass filter)

- The elements of the mask must be positive.

- A larger mask size would give a greater smoothing effect.
  - Too much smoothing will eventually lead to blurring.

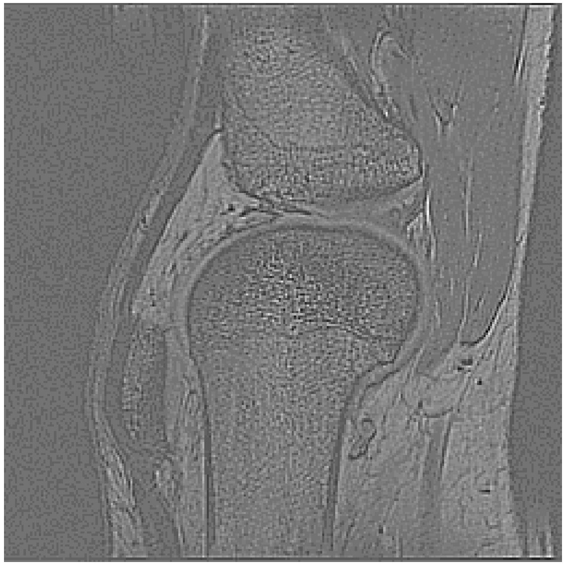
Sharpening (or High-pass)

- It is used to emphasize the fine details of an image (has the opposite effect of smoothing).
- Points of high contrast can be detected by computing intensity **differences** in local image regions.
- The weights of the mask are both positive and negative.

- Sharpening an image increases the contrast between bright and dark regions to bring out features.
- The sharpening process is basically the application of a high pass filter to an image. The following array is a kernel for a common high pass filter used to sharpen an image

$$\begin{bmatrix} -1/8 & -1/8 & -1/8 \\ -1/8 & 1 & -1/8 \\ -1/8 & -1/8 & -1/8 \end{bmatrix}$$

Original



Sharpen Filtered：Pixels that differ dramatically in contrast with surrounding pixels are brightened

- When the mask is over an area of constant or slowly varying gray level, the result of convolution will be close to zero.
- When gray level is varying rapidly within the neighborhood, the result of convolution will be a large number.
- Typically, such points form the border between different objects or scene parts (i.e., sharpening is a precursor step to edge detection).

10	10	10	10	10	10	80	80	80		
10	10	10	10	10	10	80	80	80		
10	10	10	10	10	10	80	80	80		
10	10	10	10	10	10	80	80	80		
10	10	10	10	10	10	80	80	80		

1/9 x

-1	-1	-1
-1	8	-1
-1	-1	-1

1/9 (−10 − 80 − 80 −10 + 640 −80 −10 −80 −80) = 210/9 > 0
(there is variation in the gray−levels)

1/9 (−10 − 10 − 10 −10 + 80 −10 −10 −10 −10) = 0
(there is no variation in the gray−levels)

- A similar result can be obtained in spatial domain using a high boost spatial filter.

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & x & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

- The filtering is done by convolving the mask with the image.
- The value *x* determines the amount of low-frequency information retained in the resulting image.
  - If $x = 8$ → pure highpass filter
  - If x < 8 → results in a negative of the original
  - If $x > 8$ → retain some low frequency information

- In general, the larger the value of x is, the more low-frequency information is retained.
- A larger mask will emphasize the edges more (make them wider), but help to reduce the noise effect.
- If we create an $N * N$ mask, the value for x for a highpass filter is $(N * N) - 1$.

Derivative operator

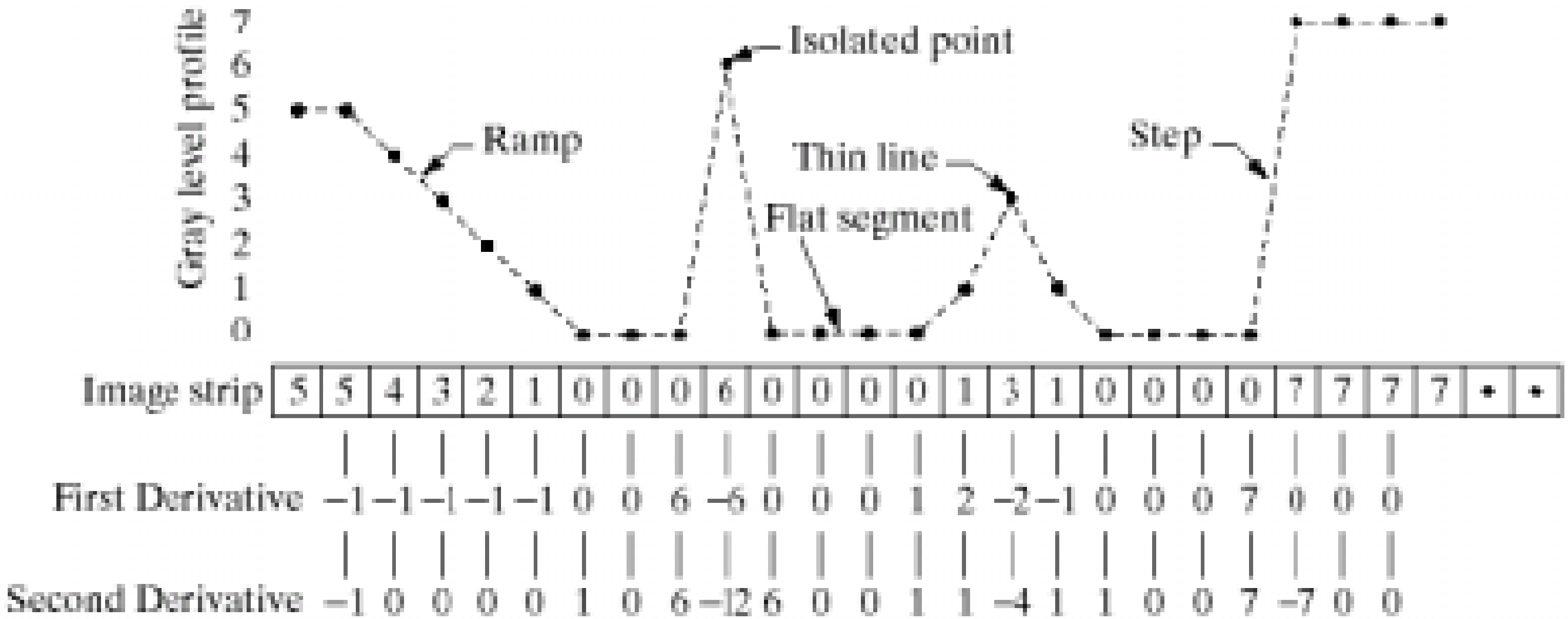
- Sharpening filters that are based on first- and second-order derivatives, respectively
- The derivatives of a digital function are defined in terms of differences

- First derivation
  - Must be zero in areas of constant intensity
  - Must be nonzero at the onset and end of an intensity step or ramp
  - Must be nonzero along ramps
- Second derivative
  - Must be zero in constant areas
  - Must be nonzero at the onset and end of an intensity step or ramp
  - Must be zero along ramps of constant slope

- A basic definition of the first-order and second-order derivative of a one-dimensional function *f(x)*

$$\frac{\partial f}{\partial x} = f(x+1) - f(x)$$

$$\frac{\partial^2 f}{\partial x^2} = f(x+1) + f(x-1) - 2f(x)$$


Gray level profile
7
6
5
4
3
2
1
0
Isolated point
Ramp
Thin line
Flat segment
Step
Image strip 5 5 4 3 2 1 0 0 0 6 0 0 0 0 1 3 1 0 0 0 0 7 7 7 7 • •
First Derivative −1 −1 −1 −1 −1 0 0 6 −6 0 0 0 1 2 −2 −1 0 0 0 7 0 0 0
Second Derivative −1 0 0 0 0 1 0 6 −12 6 0 0 1 1 −4 1 1 0 0 7 −7 0 0

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

$$\frac{\partial^2 f}{\partial x^2} = f(x+1, y) + f(x-1, y) - 2f(x, y)$$

$$\frac{\partial^2 f}{\partial y^2} = f(x, y+1) + f(x, y-1) - 2f(x, y)$$

$$\begin{aligned}&\nabla^2 f\\&= f(x+1, y) + f(x-1, y) + f(x, y+1)\\&+ f(x, y-1) - 4f(x, y)\end{aligned}$$

0	**1**	**0**
1	-4	1
0	1	0

1	**1**	**1**
1	-8	1
1	1	1

0	**-1**	**0**
-1	4	-1
0	-1	0

-1	**-1**	**-1**
-1	8	-1
-1	-1	-1

3. Unsharp masking and highboost filtering

- Unsharp masking: subtracting an unsharp (smoothed) version of an image from the original image
  - Blur the original image
  - Subtract the blurred image from the original (the resulting difference is called the mask)
  - Add the mask to the original

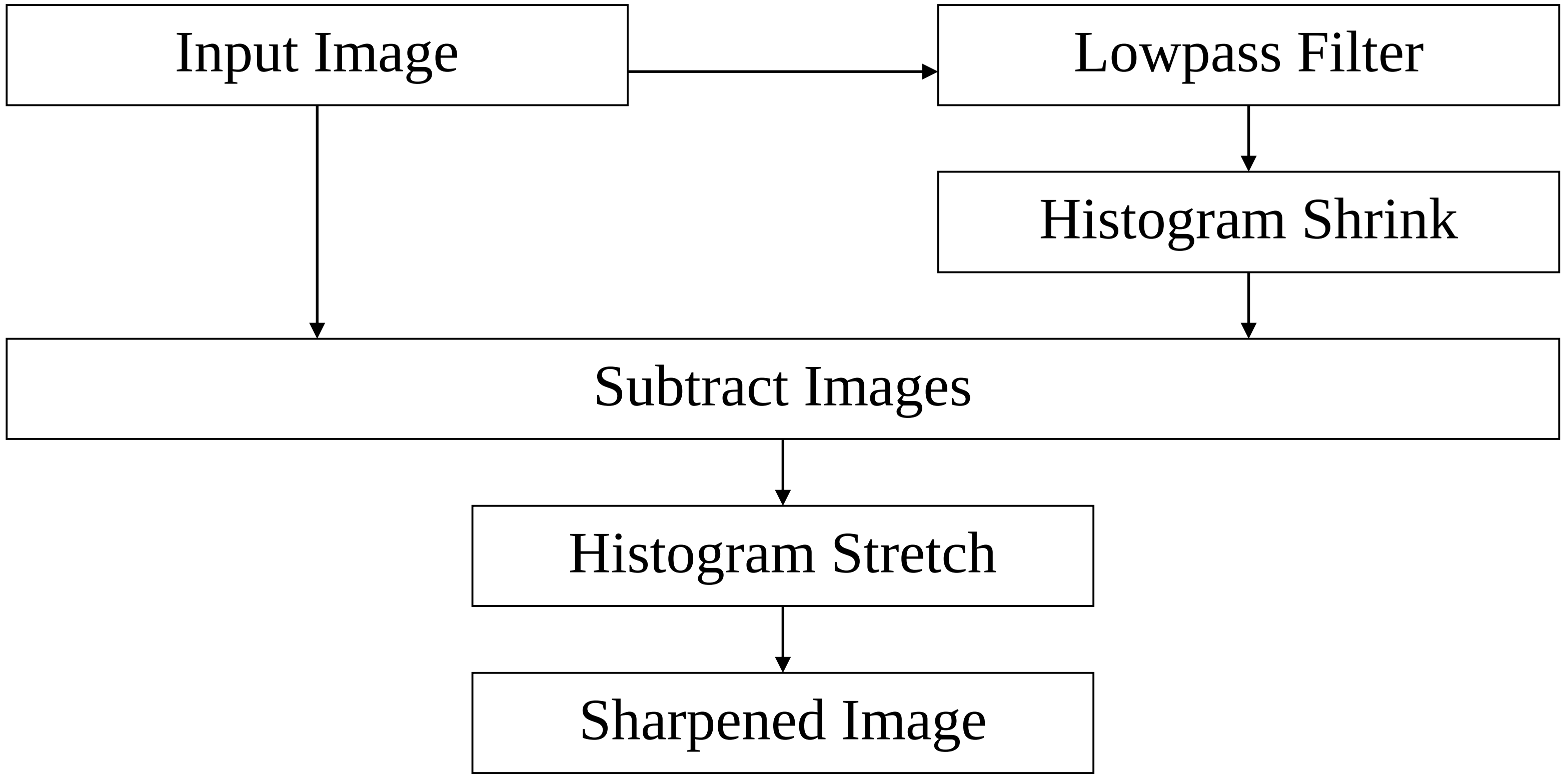
- Denote $\bar{f}(x, y)$ the blurred image

$$g_{mask}(x, y) = f(x, y) - \bar{f}(x, y)$$

$$g(x, y) = f(x, y) + k * g_{mask}(x, y)$$

  - When k=1, we have unsharp masking.
  - When k>1, the process is referred to as **highboost filtering**

- The unsharp masking enhancement algorithm is one of the more practical image sharpening methods.
- It combines many of the operations discussed before, including filtering and histogram modification.
- The flowchart of the process is shown in the next slide.


Input Image
Lowpass Filter
Histogram Shrink
Subtract Images
Histogram Stretch
Sharpened Image

- The subtraction has the visual effect of causing overshoot and undershoot around the edges, which will emphasize the edges.
- By scaling the lowpassed image with a histogram shrink, we can control the amount of edge emphasis desired.
    - To get more sharpening effect, shrink the histogram less.

Original image

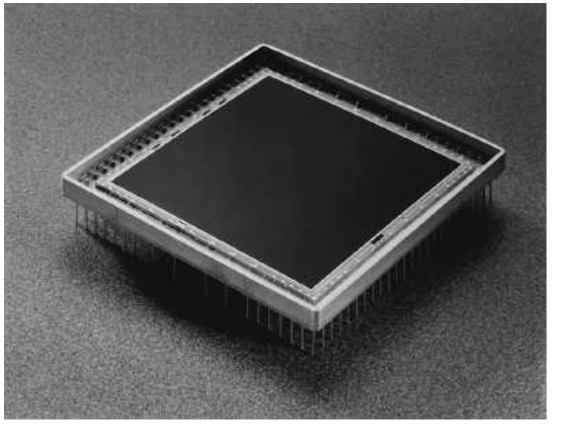
Result of unsharp masking with lower limit = 0, upper limit = 100

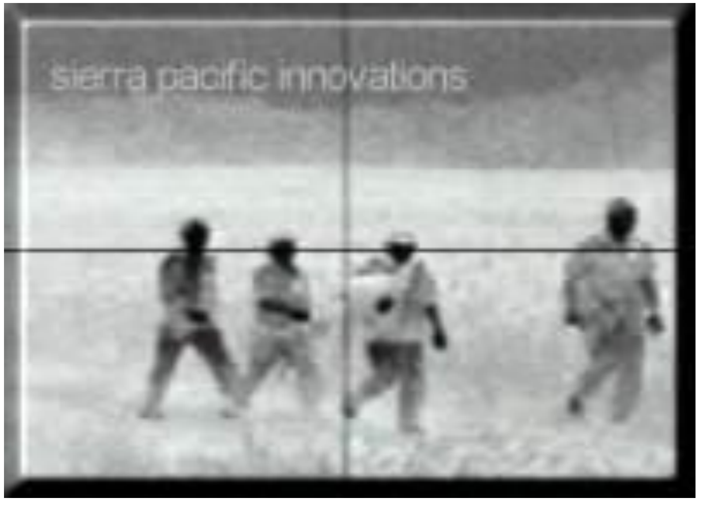
Result of unsharp masking with lower limit = 0, upper limit = 150

Result of unsharp masking with lower limit = 0, upper limit = 200

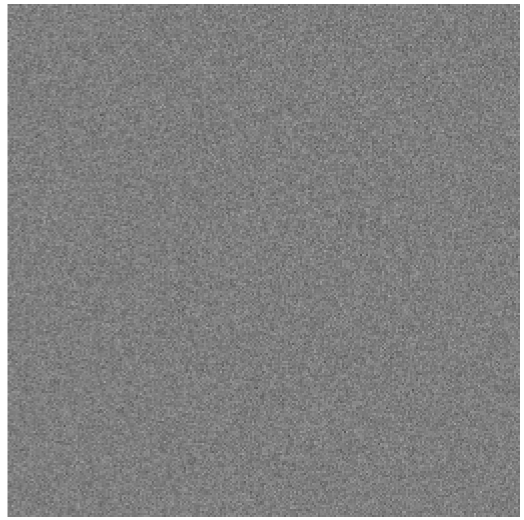
Noise

- Source of noise = CCD chip.
- Electronic signal fluctuations in detector.
- Caused by thermal energy.
- Worse for infra-red sensors.





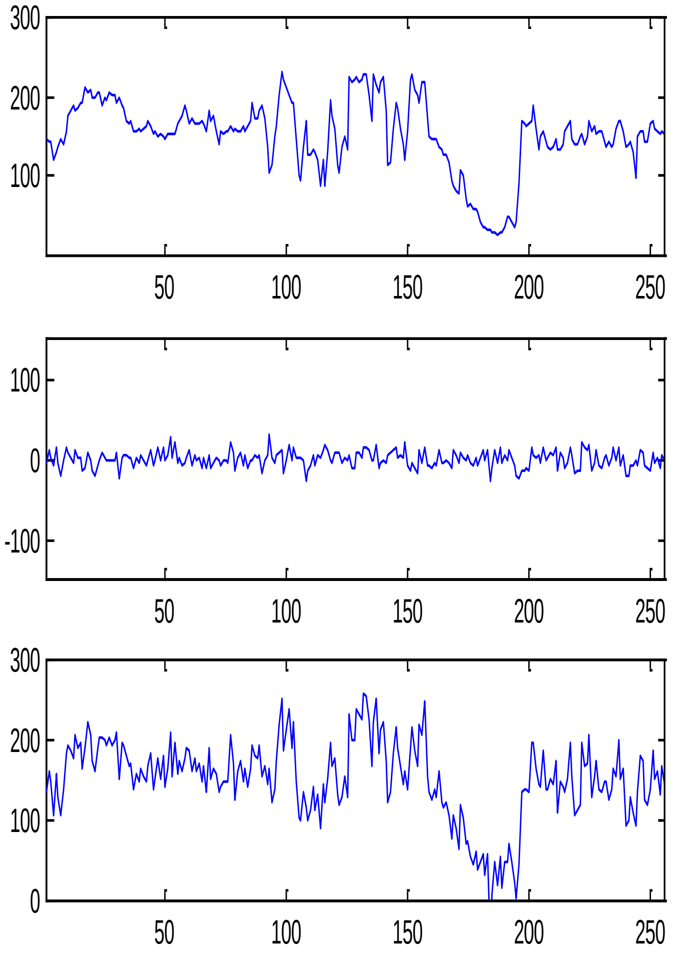
image



noise

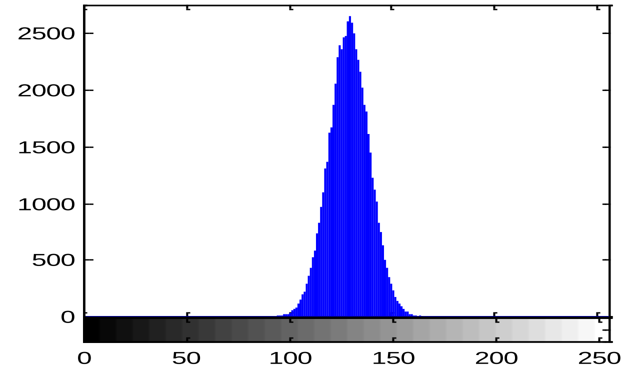
‘grainy’
image

- Plot of image brightness.
- Vertical slice through image.
- Noise is additive.
- Noise fluctuations are rapid, ie, high frequency.



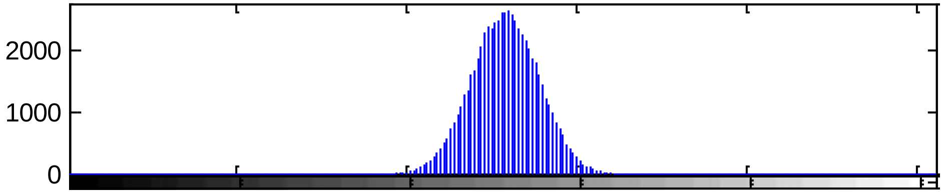
Noise Histogram

- Plot noise histogram
- Histogram is called normal or Gaussian (distribution)
- Mean (noise) μ = 0
- Standard deviation σ
- *i* is the grey level.

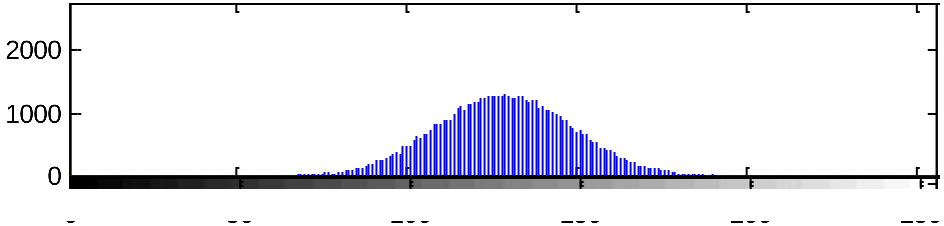


$$f(i) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{1}{2}\left(\frac{i-\mu}{\sigma}\right)^2\right]$$

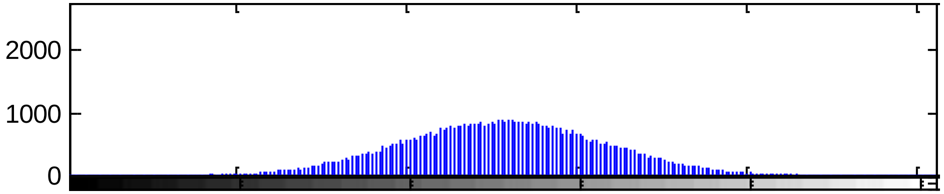
σ=10



σ=20

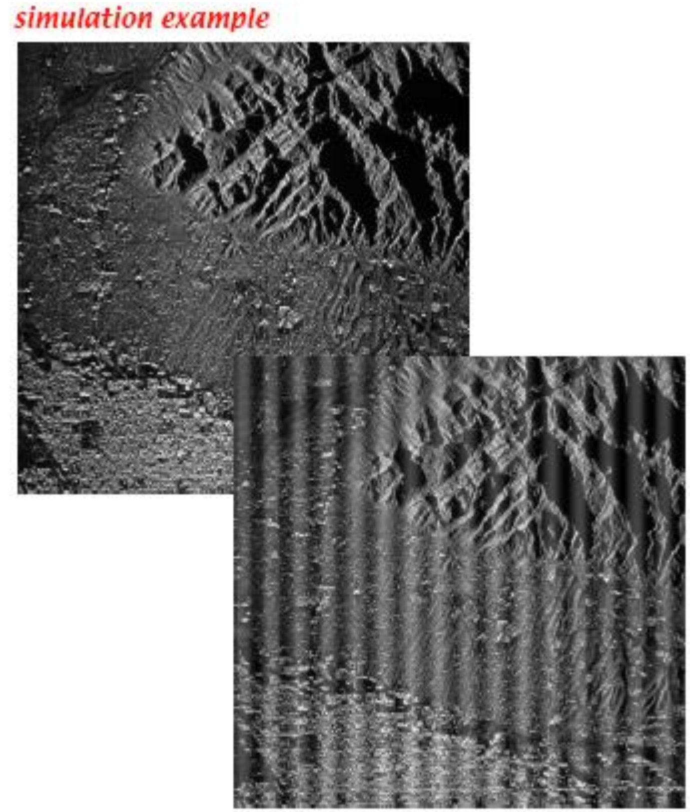


σ=30



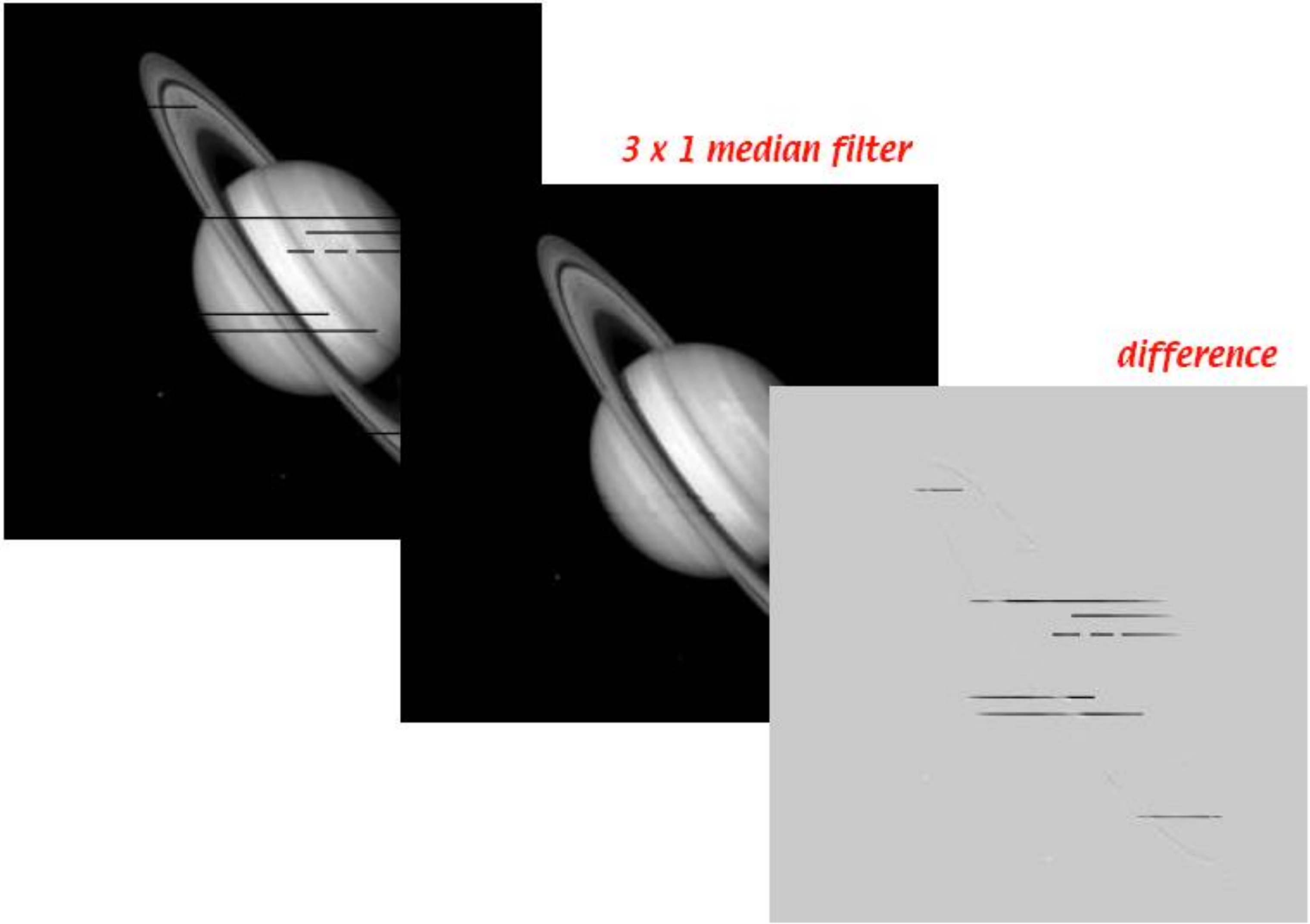
Structured Noise

- ***Periodic, stationary***
  - Noise has fixed amplitude, frequency and phase
  - Commonly caused by interference between electronic components



- ***Periodic, nonstationary***
  - Noise parameters (amplitude, frequency, phase) vary across the image
  - Intermittent （斷斷續續） interference between electronic components

Line Drop Removal



- **Gaussian noise** is statistical noise that has its probability density function equal to that of the normal distribution, which is also known as the Gaussian distribution.
- A special case is **white Gaussian noise**, in which the values at any pairs of times are statistically independent (and uncorrelated).
- In applications, Gaussian noise is most commonly used as additive white noise to yield additive white Gaussian noise

- **Salt and pepper noise** is a form of noise typically seen on images.
- It represents itself as randomly occurring white and black pixels.

- The **grain** of photographic film is a signal-dependent noise, is uniformly distributed (equal number per area)