CHAPTER 3

AREA PROCESSES

Area processes use the input pixel as well as the pixels around it to generate a new output pixel.

Convolution

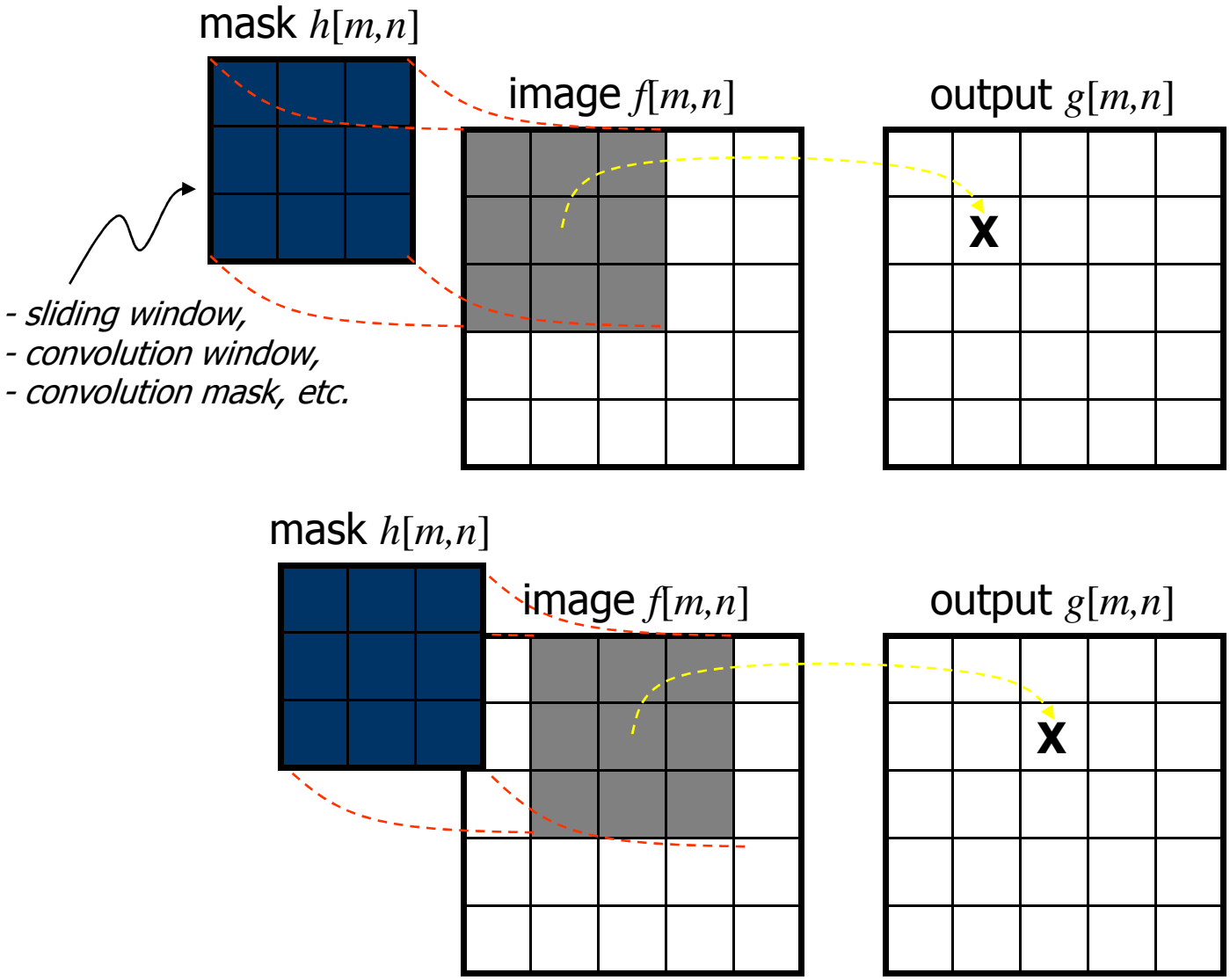
- Mathematical Formulation

$$g(x,y) = h(x,y) * f(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(x-x', y-y')\, f(x', y')\, dx'\, dy'$$

$$g[m,n] = h[m,n] * f[m,n] = \sum_{m'=-\infty}^{\infty}\sum_{n'=-\infty}^{\infty} h[m-m', n-n']\, f[m', n']$$

- Convolution in image processing consists of the following processes:
  - Overlay the mask $h[m,n]$ on the image $f[m,n]$
  - Multiply the coincident terms
  - Sum all the results
  - Move to the next pixel, across the entire image

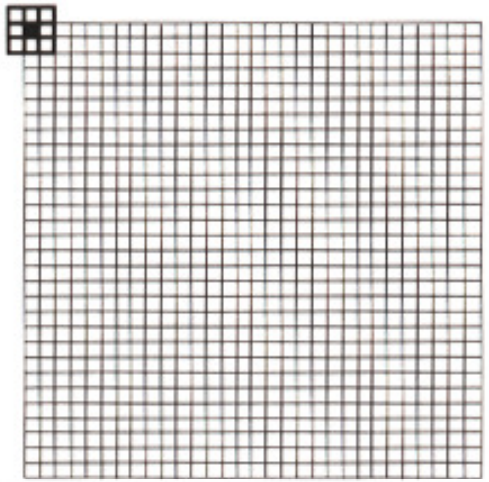
Convolution



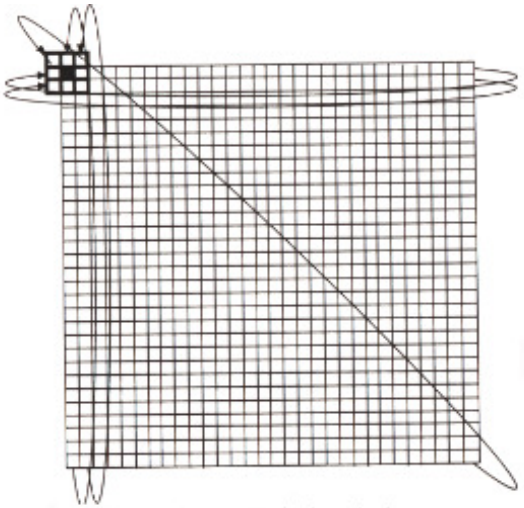
Convolution

- How to treat the image borders
  - Zero padding: Treat the empty cells in the convolution window as zeros.
  - Skipping border pixels: Start convolving with the pixel at (1,1) instead of the pixel at (0,0).
  - Enlarging image before convolving: Duplicate border pixels.
  - Wrapping around: Assume the image is periodic in both horizontal and vertical axes.

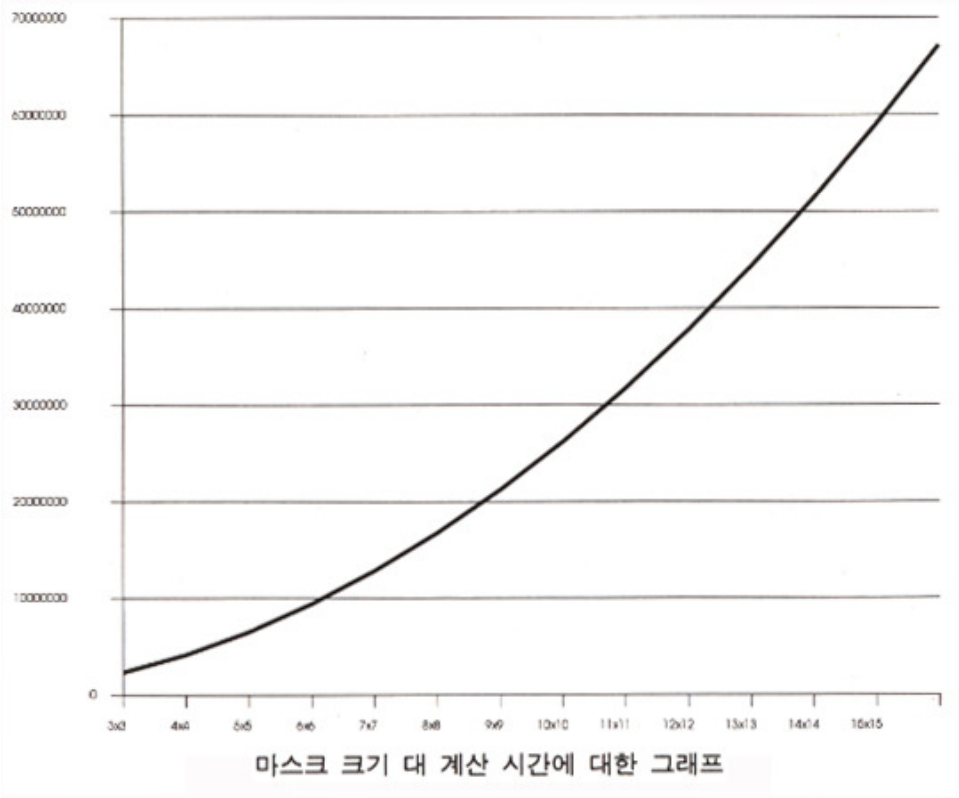
Convolution



영상을 중첩한 회선 마스크



에지 주위를 둘러싼 회선 마스크



마스크 크기 대 계산 시간에 대한 그래프

Convolution

- Separable Masks
  - Separable function: $f(x,y)=g(x)h(y)$
  - The convolution can be performed by executing two convolutions with 1-D masks.
  - Separable functions reduce the number of computations required when using large masks.

$$\underbrace{\begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}}_{9 \text{ multiplications}} = \underbrace{\begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix} \begin{pmatrix} 1 & 2 & 1 \end{pmatrix}}_{6 \text{ multiplications}}$$

  - Savings increase as mask sizes grow.

Convolution

- Embossing
  - The coefficients of the mask have a center weight of 0 and sum to 0.
  - The key feature is the canceling coefficients about the center coefficients.

-1	0	0
0	0	0
0	0	1


original image


embossed image


embossed and inverted


embossed and rescaled

Blurring

- Blurring (Lowpass Spatial Filtering)
  - Removes the finer details of an image.
  - Reduces high-frequency noise.

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \qquad \frac{1}{25}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

  - Blurring is simply neighborhood averaging.
  - Averaging tends to remove extreme values from a group.

Blurring

original image

3X3 blurring

5X5 blurring

7X7 blurring

Blurring

original image with noise

3X3 blurring

5X5 blurring

7X7 blurring

Gaussian Smoothing

- Gaussian Smoothing Filters

$$G(x, y) = G(x)G(y) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}} \times \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{y^2}{2\sigma^2}} = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

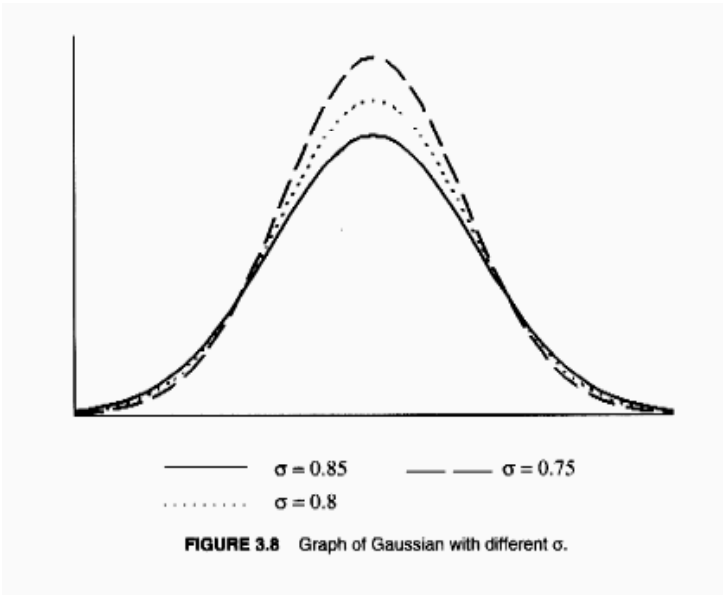
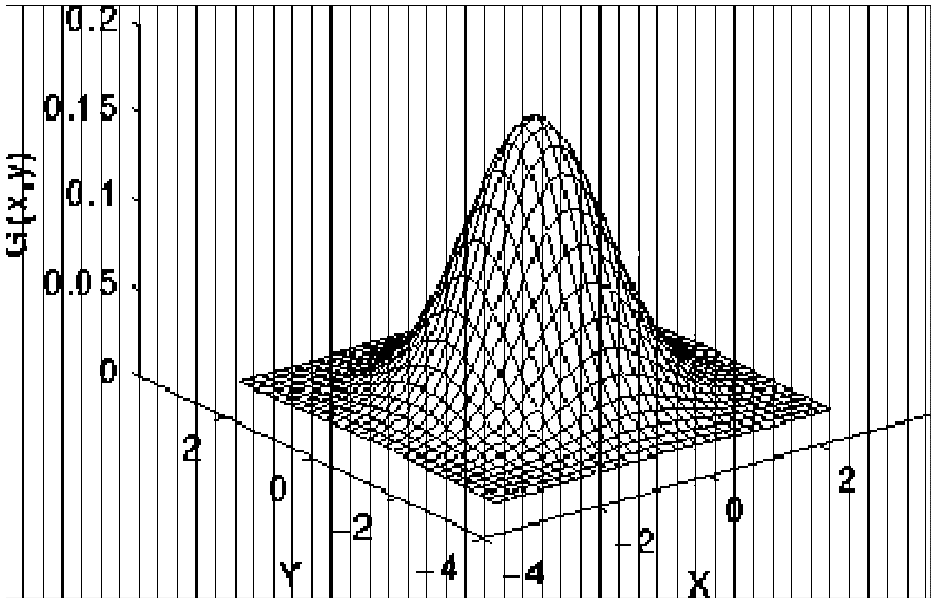


FIGURE 3.8 Graph of Gaussian with different σ.



Gaussian Smoothing

- Mask Approximation for Gaussian Smoothing

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

$\frac{1}{16}$

1	2	1
2	4	2
1	2	1

  - Size: variable with σ (standard deviation)
  - 5x5 when σ=1.0 (15x15 when σ=2.0)
  - Separate convolutions reduce computations.

original image

Gaussian filtering

3X3 blurring

Sharpening

- Sharpening has the opposite affect of blurring.
- Emphasizes the details in an image.
- The convolution mask has a positive coefficient in its center and mostly negative coefficients around the outer edge.

(a)

0	-1	0
-1	5	-1
0	-1	0

(b)

-1	-1	-1
-1	9	-1
-1	-1	-1

(c)

1	-2	1
-2	5	-2
1	-2	1

original image

Sharpening

original image

sharpend by (a)

sharpend by (b)

sharpend by (c)

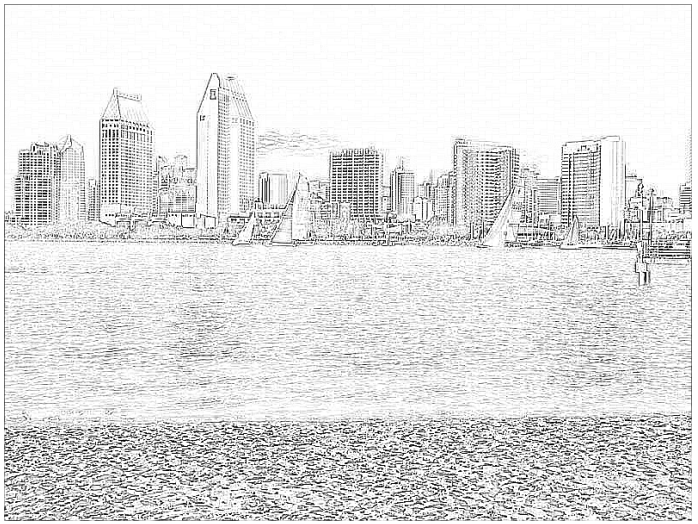
Sharpening

- Sharpening is based on the highpass filter.
- A common highpass filter mask:

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

Note: coefficients sum to 0

original image



highpass-filtered image

Sharpening

- Unsharp Masking (a method of highpass filtering)
  - [highpass] = [original] – [lowpass]
- High-Boost Filtering
  - [High Boost] = α [original] – [lowpass]

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & \alpha & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} = \frac{1}{9}\begin{bmatrix} -1 & -1 & -1 \\ -1 & w & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

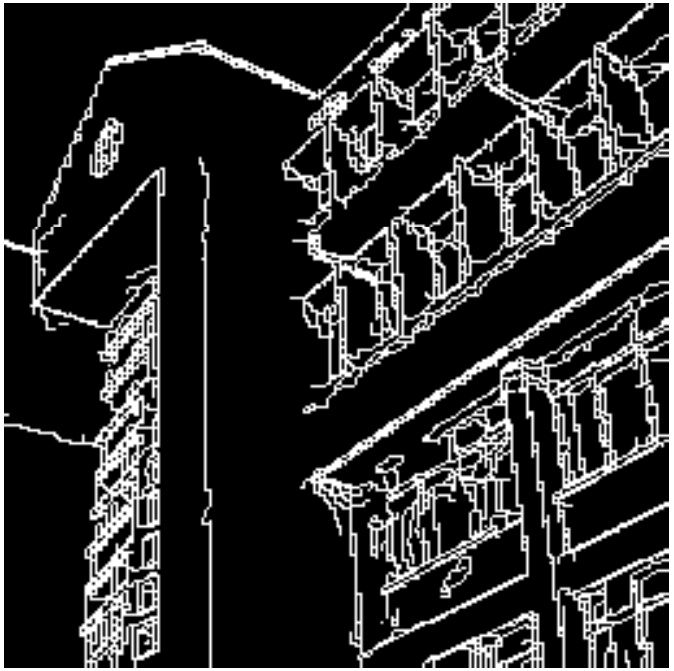
$$\alpha - \frac{1}{9} = \frac{w}{9} \quad \text{or} \quad w = 9\alpha - 1$$

  - α=1: highpass filter
  - α>1: a fraction of original is added back to highpass result

Sharpening

Edge Detection

- Convert a 2D image into a set of curves
  - Extracts salient features of the scene
  - More compact than pixels



- How can you tell that a pixel is on an edge?

Edge Detection

- Intensity changes rapidly around edges.
- Different edge profiles:

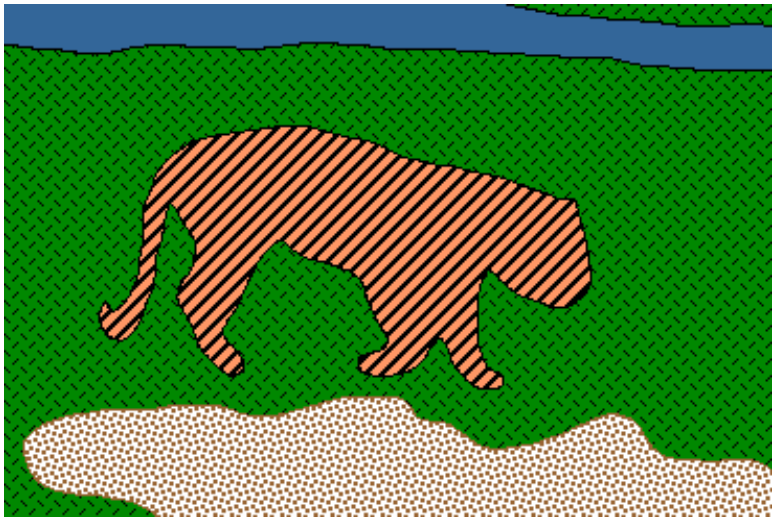
roof edge line edge step edge ramp edge

- The edges tell:
  - Where objects are
  - Their shape and size
  - Something about their texture

Edge Detection

- The applications include image segmentation and registration:
  - Image segmentation: groups pixels into regions to determine an image's composition.

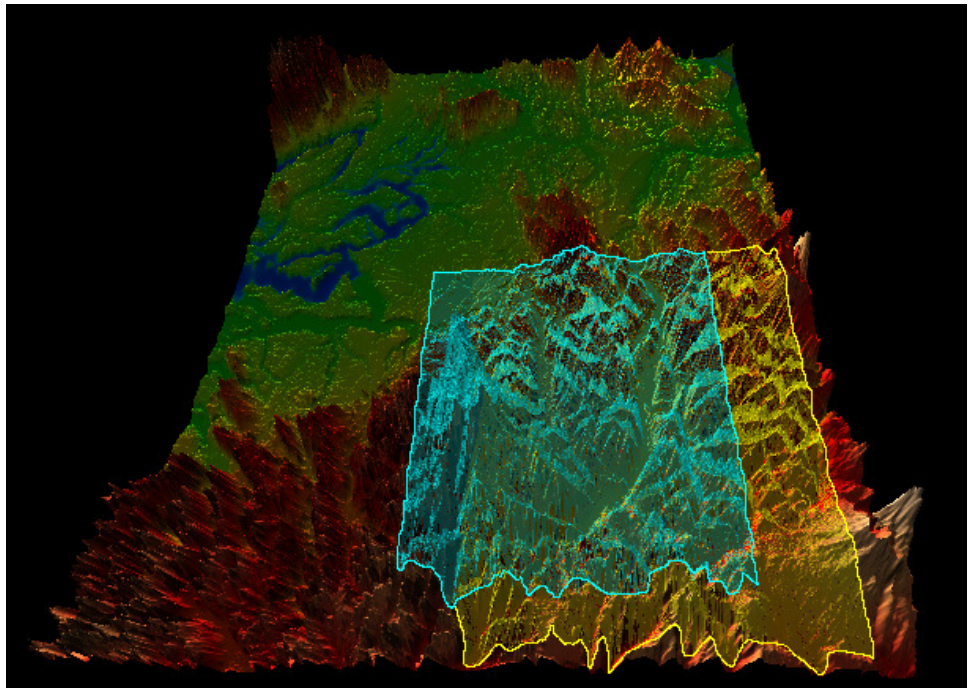
original image



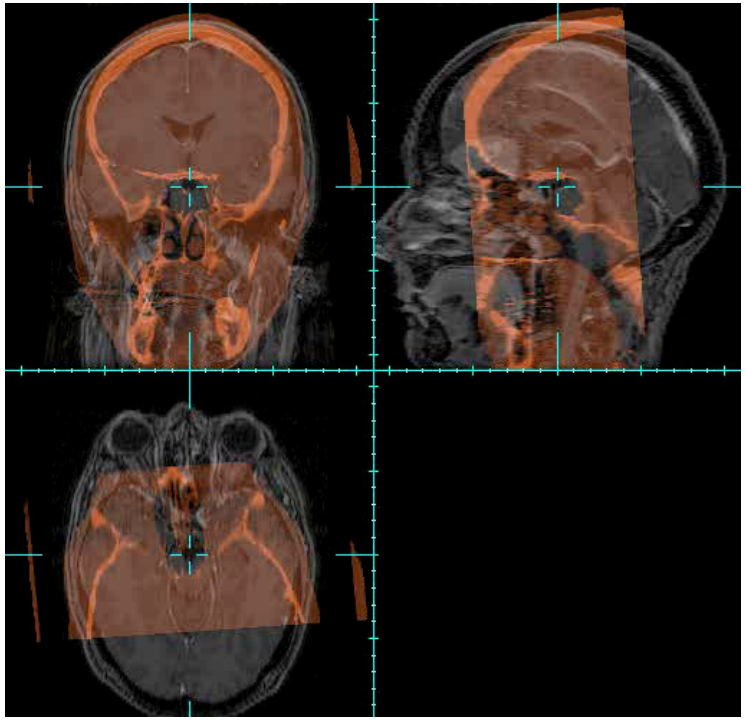
segmented image

Edge Detection

- Image registration: aligns two images that may have been acquired at separate times or from different sensors.



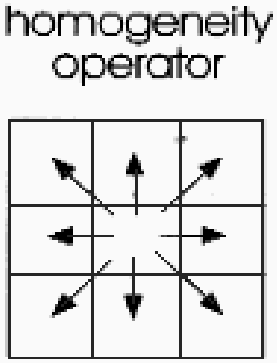
range image registration

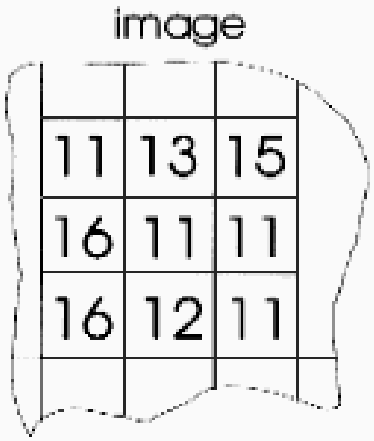


medical image registration

Simple Edge Detectors

- Homogeneity Operator
  - Subtract each 8 surrounding pixels from the center pixel of a 3x3 window and choose the maximum of the absolute value.





new pixel = maximum of { |11-11| |11-13| |11-15|
|11-16| |11-11|
|11-16| |11-12| |11-11| } = 5

FIGURE 3.12 How the homogeneity operator works.

Simple Edge Detectors

- Difference Edge Detector
  - Subtracts four pixels:

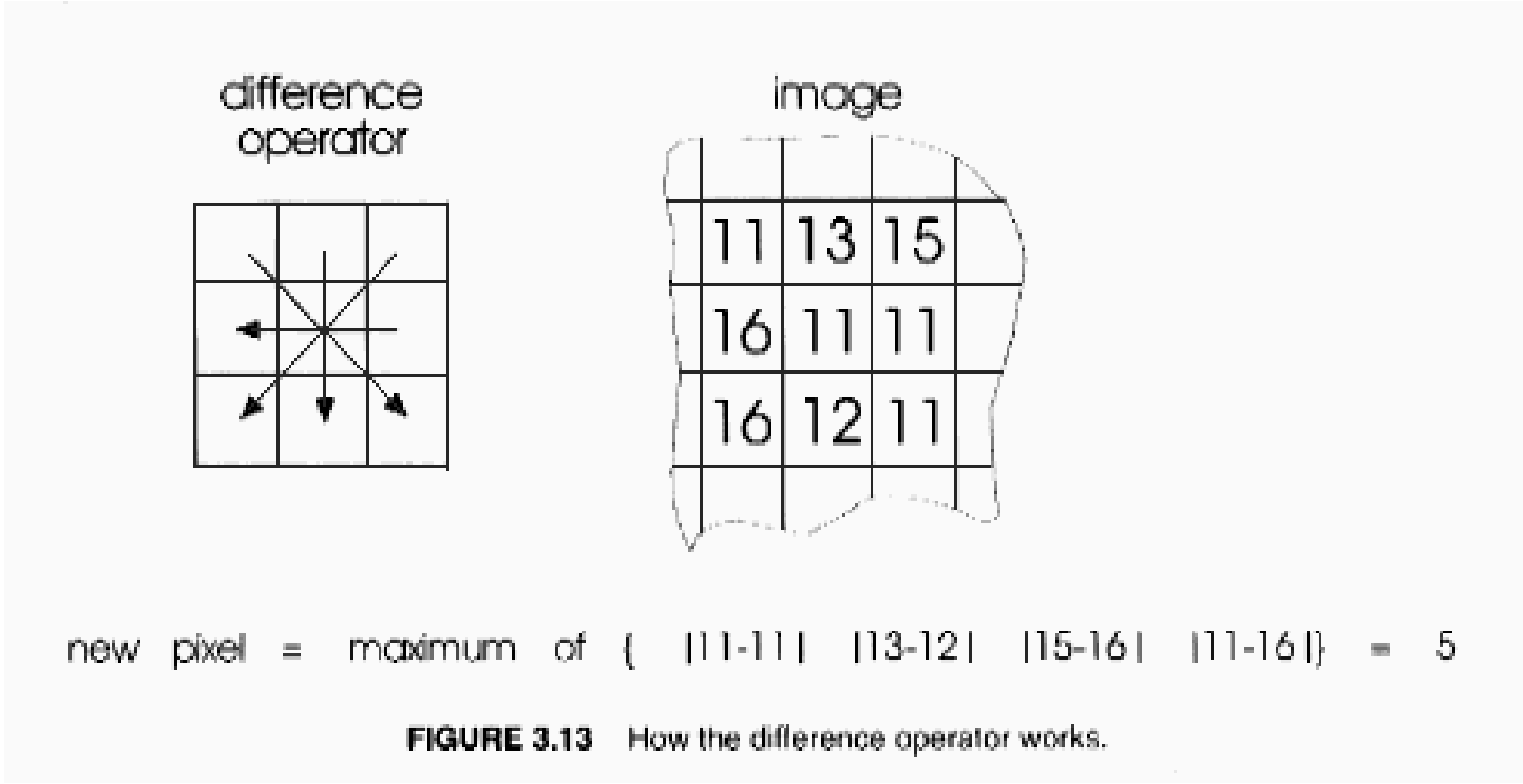
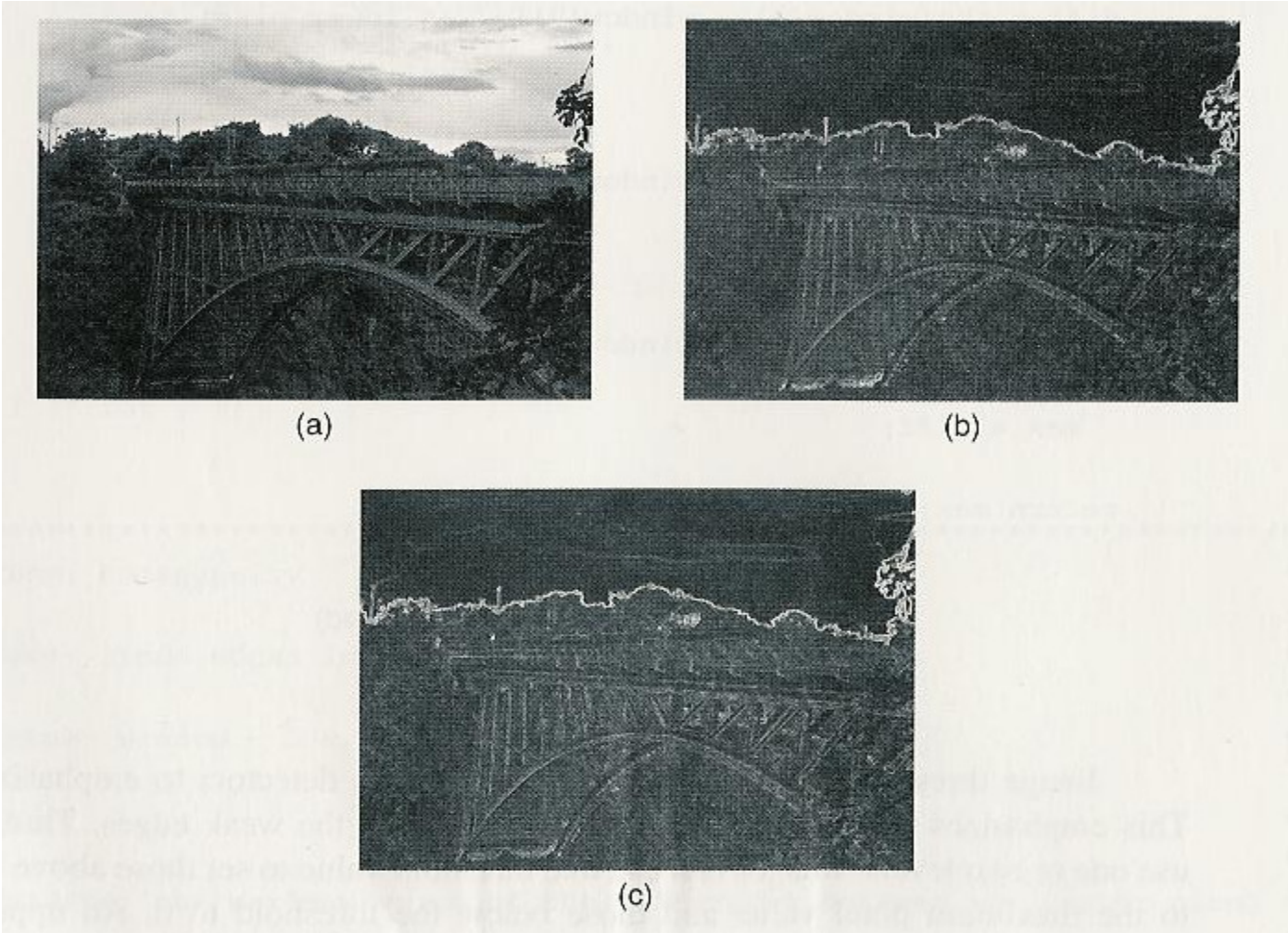


FIGURE 3.13 How the difference operator works.

Simple Edge Detectors



(a) Original image, (b) edge map of homogeneity operator,

(c) Edge map of difference operator.

Simple Edge Detectors

- Thresholding an edge map
  (Emphasize the strong edges and deemphasize the weak edges)

  - single threshold
    - g(x) = 255 if x >= threshold
    - 0 otherwise
  - double threshold
    - g(x) = 255 if x >= high_threshold
    - 0 if x <= low_threshold
    - x otherwise

Edge Detection

- Edge is Where Change Occurs
  - Change is measured by derivative in 1D
  - Biggest change, derivative has maximum magnitude
  - Or 2nd derivative is zero.

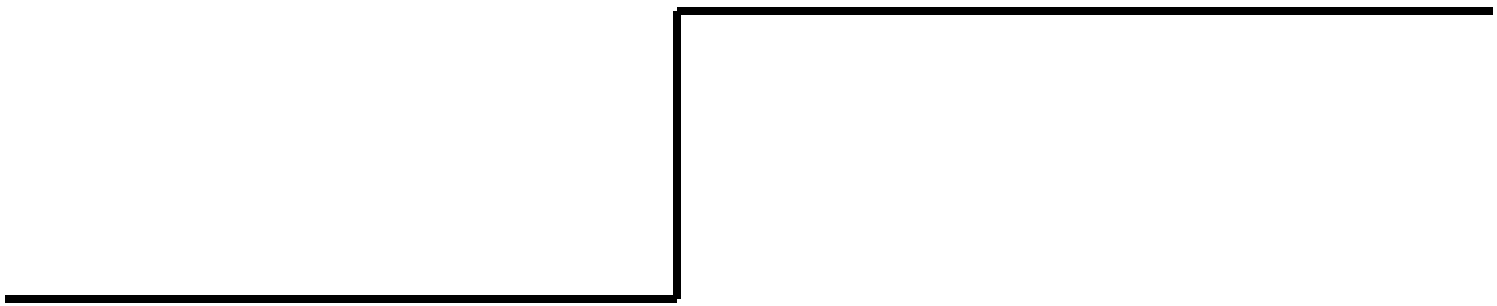
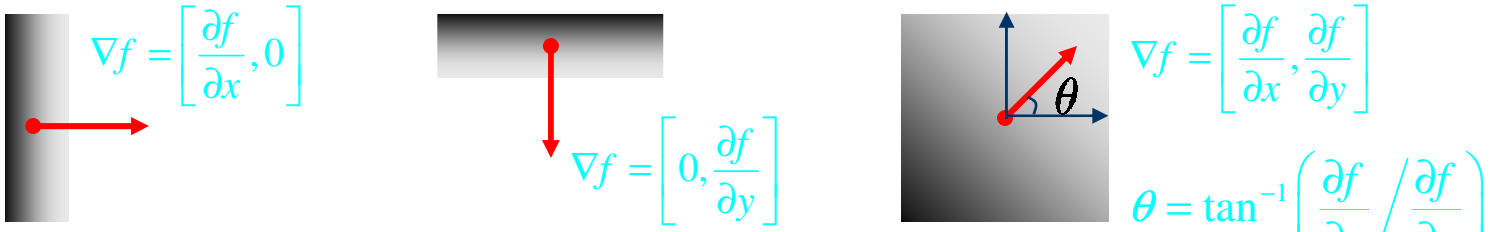


Image Gradient

- The gradient of an image:
  edge is where change occurs



The gradient points in the direction of most rapid change in intensity

- The edge strength is given by the gradient magnitude

$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

The Discrete Gradient

- How can we differentiate a digital image $f(x,y)$?
  - Continuous gradient:

$$\frac{\partial f(x,y)}{\partial x} = \lim_{\Delta x \to 0} \frac{f(x+\Delta x, y) - f(x,y)}{\Delta x}$$

$$\frac{\partial f(x,y)}{\partial y} = \lim_{\Delta y \to 0} \frac{f(x, y+\Delta y) - f(x,y)}{\Delta y}$$

  - Discrete gradient:

$$\frac{\partial f(x,y)}{\partial x} \approx f[x+1, y] - f[x,y]$$

$$\frac{\partial f(x,y)}{\partial y} \approx f[x, y+1] - f[x,y]$$

Common Gradient Operators (Masks)

- Roberts

$$\begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Sensitive to noise

- Prewitt

$$\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \qquad \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

Sensitive to vertical/horizontal edges

- Sobel

$$\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \qquad \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Sensitive to diagonal edges

- Frei-Chen

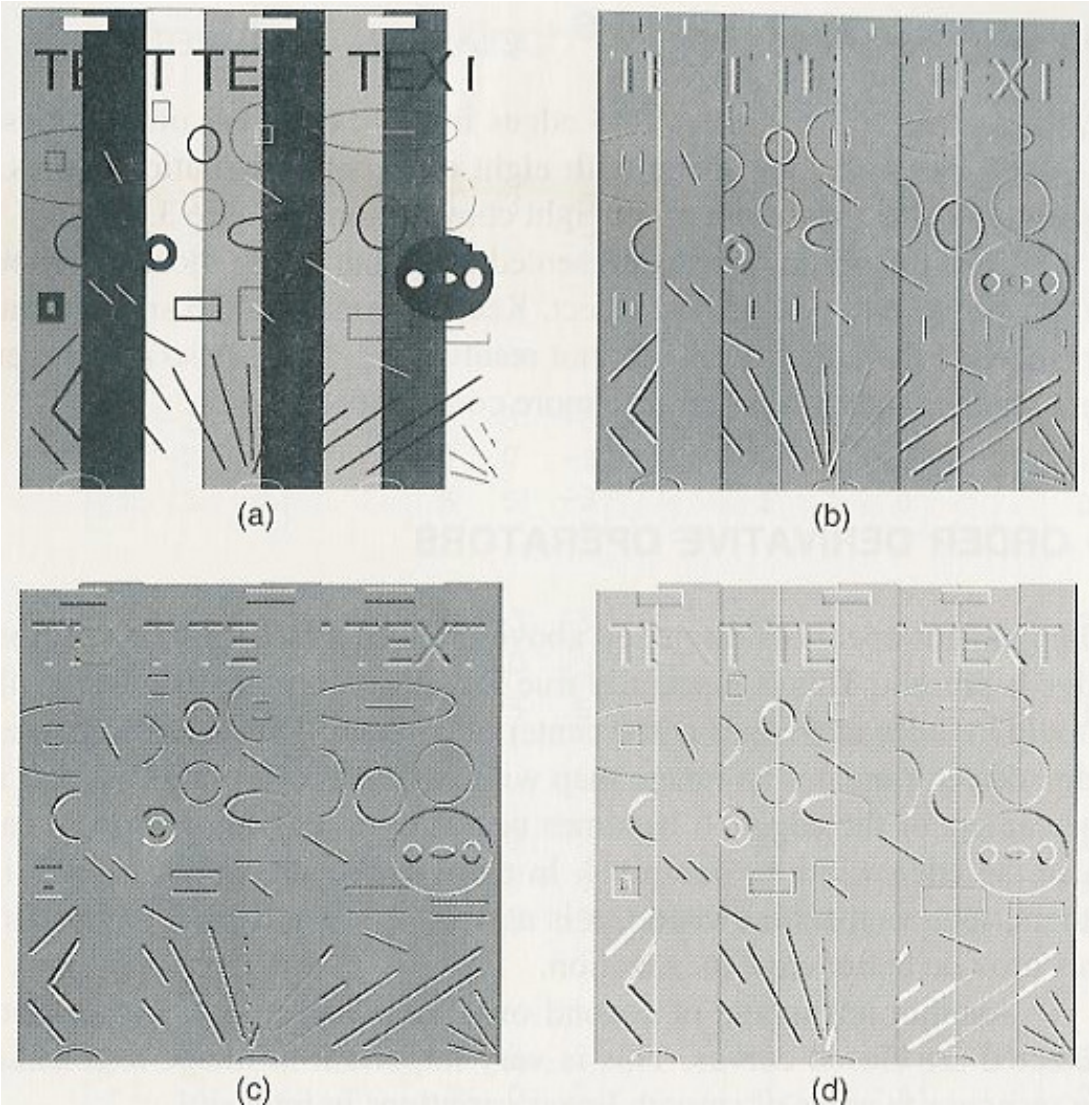
$$\begin{bmatrix} 1 & 0 & -1 \\ \sqrt{2} & 0 & -\sqrt{2} \\ 1 & 0 & -1 \end{bmatrix} \qquad \begin{bmatrix} -1 & -\sqrt{2} & -1 \\ 0 & 0 & 0 \\ 1 & \sqrt{2} & 1 \end{bmatrix}$$

Sobel

- Sobel Operation

```
for (i=1; i<nrow-1; i++) {
  for (j=1; j<ncol-1;j++) {
     hr = f[i-1][j-1] + f[i][j-1]*2 + f[i+1][j-1] - f[i-1][j+1] - f[i][j+1]*2 - f[i+1][j+1];
     hc = -f[i-1][j-1] – f[i-1][j]*2 – f[i-1][j+1] + f[i+1][j-1] + f[i+1][j]*2 + f[i+1][j+1];
     g[i][j] = sqrt(hr*hr + hc*hc);
  }
}
```

Prewitt



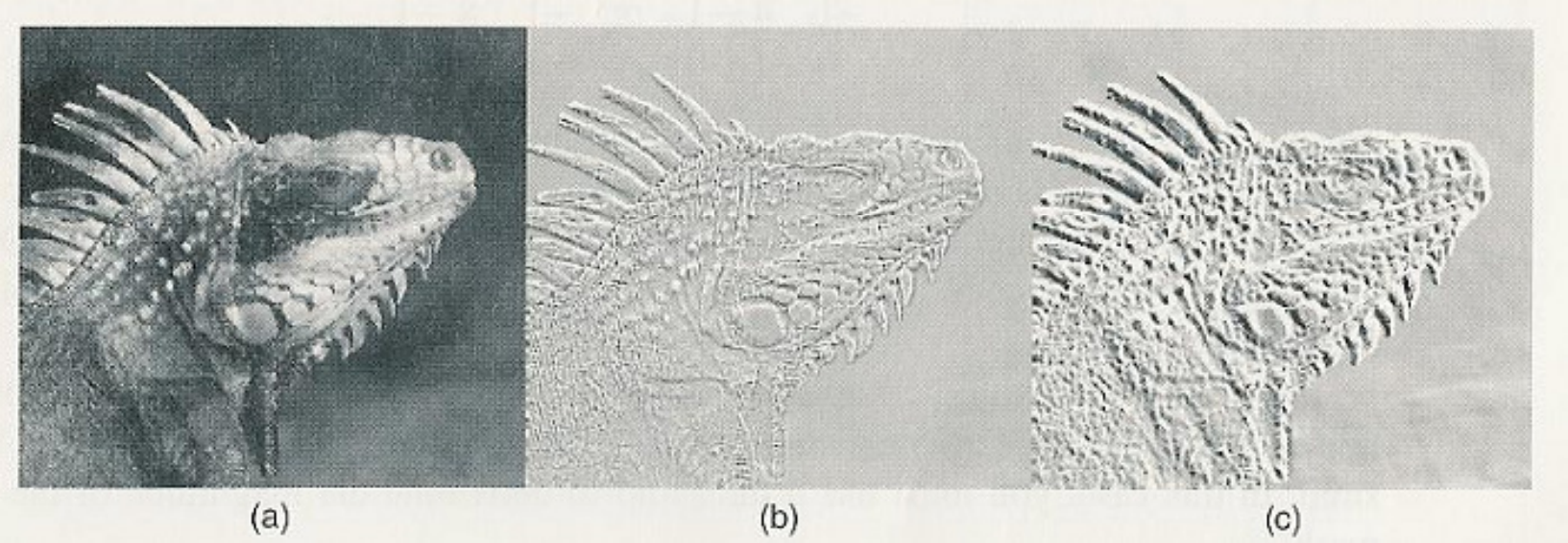
(a) Original image

(b) $\frac{\partial f}{\partial x}$

(c) $\frac{\partial f}{\partial y}$

(d) $\sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$

Roberts vs. Prewitt



Original image Processed with Roberts Processed with Prewitt

Compass Gradient Operator

- Compass Gradient Operator
  - Find edges in eight different orientations.
  - Convolve an image with eight different masks.
  - Prewitt, Kirsch, Robinson 3-level, Robinson 5-level

- The Effects of Convolution
  - Smaller masks are sensitive to noise
  - Larger masks cannot resolve fine detail, and also requires more computations.

Compass Gradient Operator

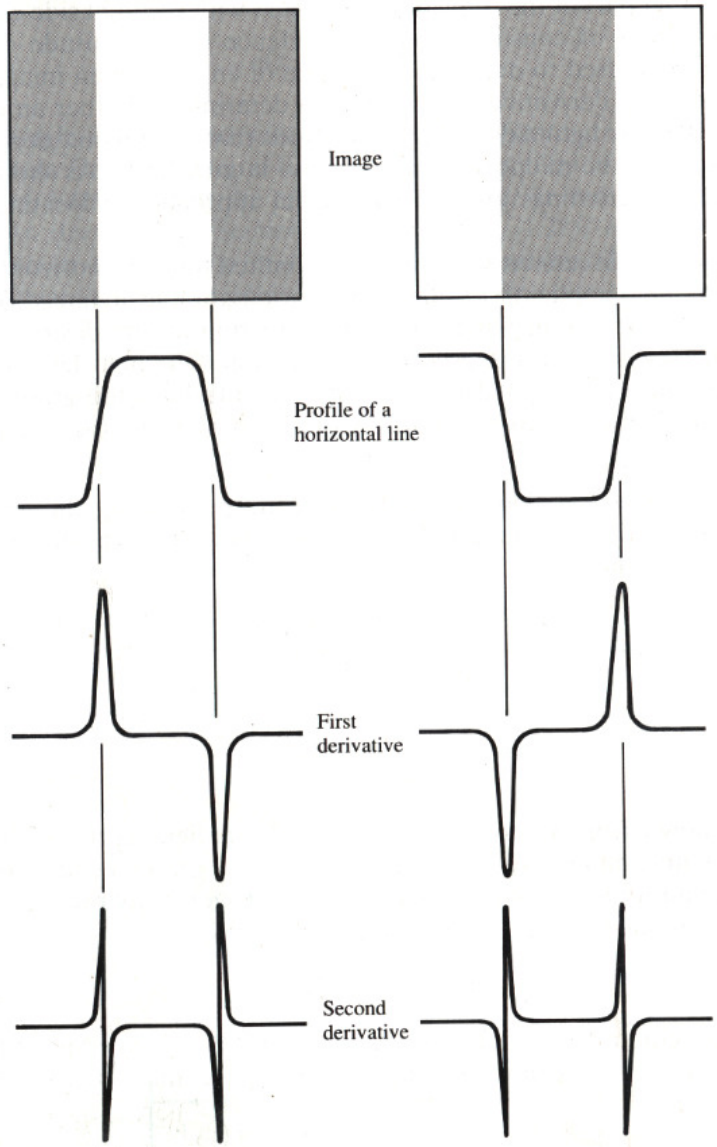
Prewitt	Kirsch	Robinson 3-level	Robinson 5-level
$\begin{bmatrix} 1 & 1 & -1 \\ 1 & -2 & -1 \\ 1 & 1 & -1 \end{bmatrix}$	$\begin{bmatrix} 5 & -3 & -3 \\ 5 & 0 & -3 \\ 5 & -3 & -3 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$
$\begin{bmatrix} 1 & -1 & -1 \\ 1 & -2 & -1 \\ 1 & 1 & -1 \end{bmatrix}$	$\begin{bmatrix} -3 & -3 & -3 \\ 5 & 0 & -3 \\ 5 & 5 & -3 \end{bmatrix}$	$\begin{bmatrix} 0 & -1 & -1 \\ 1 & 0 & -1 \\ 1 & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & -1 & -2 \\ 1 & 0 & -1 \\ 2 & 1 & 0 \end{bmatrix}$
$\begin{bmatrix} -1 & -1 & -1 \\ 1 & -2 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	$\begin{bmatrix} -3 & -3 & -3 \\ -3 & 0 & -3 \\ 5 & 5 & 5 \end{bmatrix}$	$\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & -1 \end{bmatrix}$
$\begin{bmatrix} -1 & -1 & 1 \\ -1 & -2 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	$\begin{bmatrix} -3 & -3 & -3 \\ -3 & 0 & 5 \\ -3 & 5 & 5 \end{bmatrix}$	$\begin{bmatrix} 1 & 1 & -1 \\ 1 & -2 & -1 \\ 1 & 1 & -1 \end{bmatrix}$	$\begin{bmatrix} -2 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{bmatrix}$
$\begin{bmatrix} -1 & 1 & 1 \\ -1 & -2 & 1 \\ -1 & 1 & 1 \end{bmatrix}$	$\begin{bmatrix} -3 & -3 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & 5 \end{bmatrix}$	$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$
$\begin{bmatrix} 1 & 1 & 1 \\ -1 & -2 & 1 \\ -1 & -1 & 1 \end{bmatrix}$	$\begin{bmatrix} -3 & 5 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & -3 \end{bmatrix}$	$\begin{bmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ -1 & -1 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{bmatrix}$
$\begin{bmatrix} 1 & 1 & 1 \\ 1 & -2 & 1 \\ -1 & -1 & -1 \end{bmatrix}$	$\begin{bmatrix} 5 & 5 & 5 \\ -3 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix}$	$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$	$\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$
$\begin{bmatrix} 1 & 1 & 1 \\ -1 & -2 & 1 \\ -1 & -1 & 1 \end{bmatrix}$	$\begin{bmatrix} 5 & 5 & -3 \\ 5 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix}$	$\begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -1 \end{bmatrix}$	$\begin{bmatrix} 2 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -2 \end{bmatrix}$

Second Order Derivative Operators

- Second Derivative Operators
  - Better edge localization
  - Laplacian operator
  - The resulting image exhibits a sign change at image edges. (zero crossing)

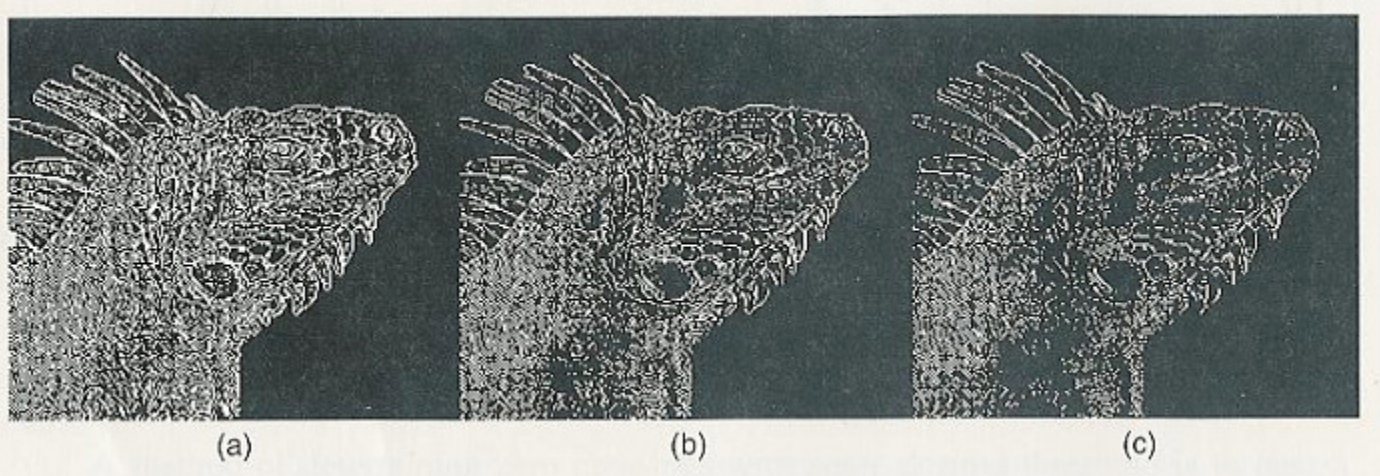
$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \quad \text{or} \quad \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

- Disadvantage of Laplacian
  - Sensitive to noise
  - Produce double edges
  - Unable to detect edge direction



Zero Crossing

- Simple zero crossing may indicate more edges than desired.
- Use a threshold to determine the magnitude of the edge to mark.
  - Pass a 3x3 window determining the maximum and minimum values within that window.

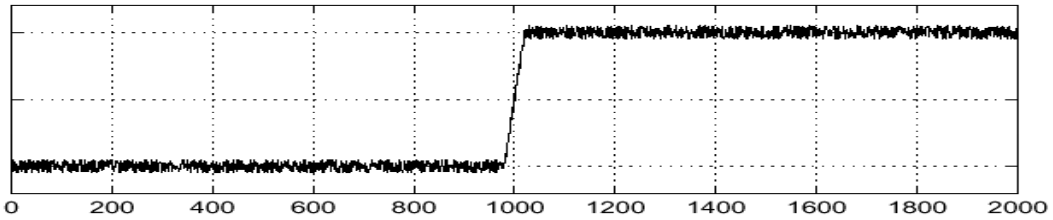


Different thresholds: (a) 20; (b) 48; (c) 72.

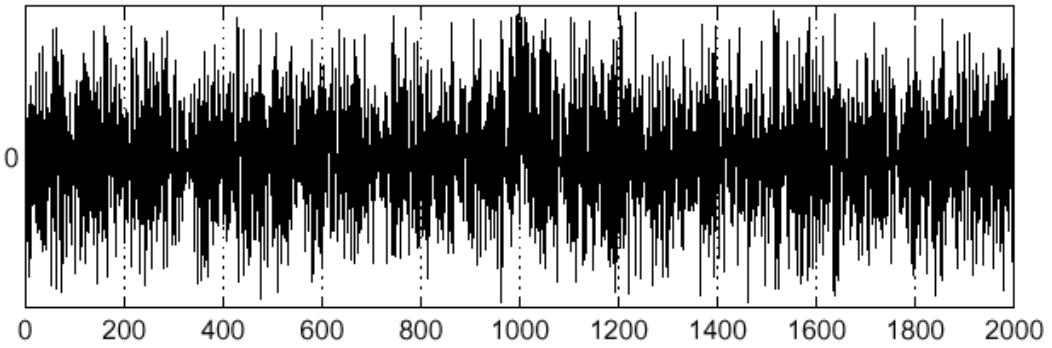
Effects of Noise

- Consider a single row or column of the image
  - Plotting intensity as a function of position gives a signal

$f(x)$



$\frac{\partial f}{\partial x}$



Where is the edge?

Effects of Noise

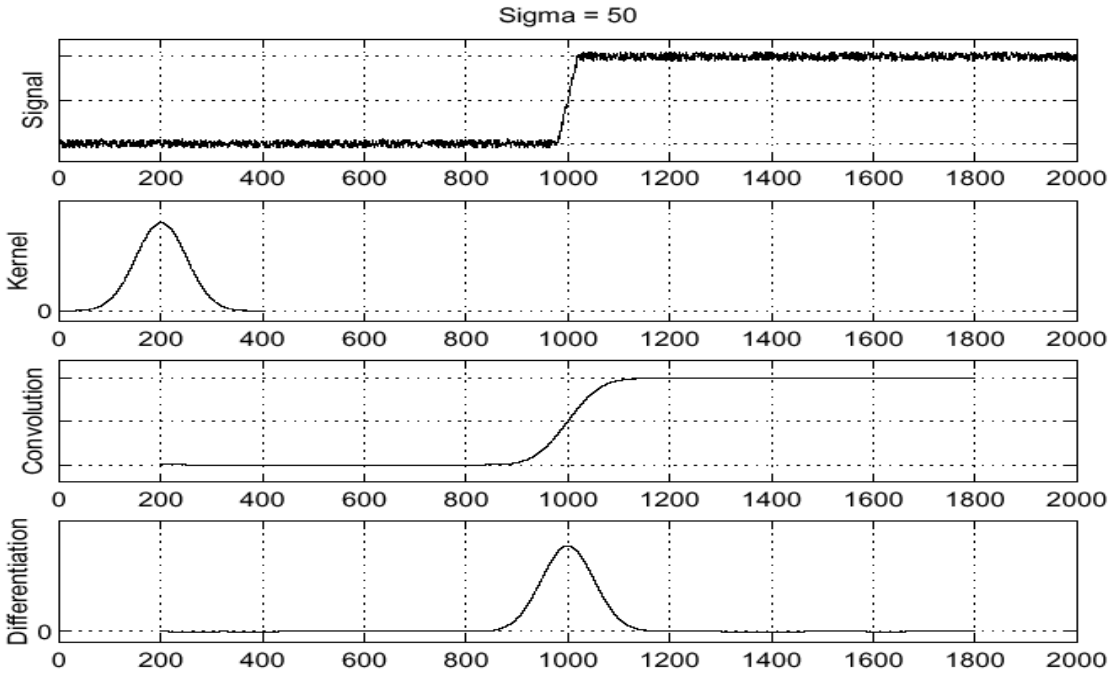
- Solution: smooth first

$f(x)$

$h(x)$

$g(x) = h(x) * f(x)$

$\frac{\partial g(x)}{\partial x}$



Where is the edge? ➡ Look for peaks in

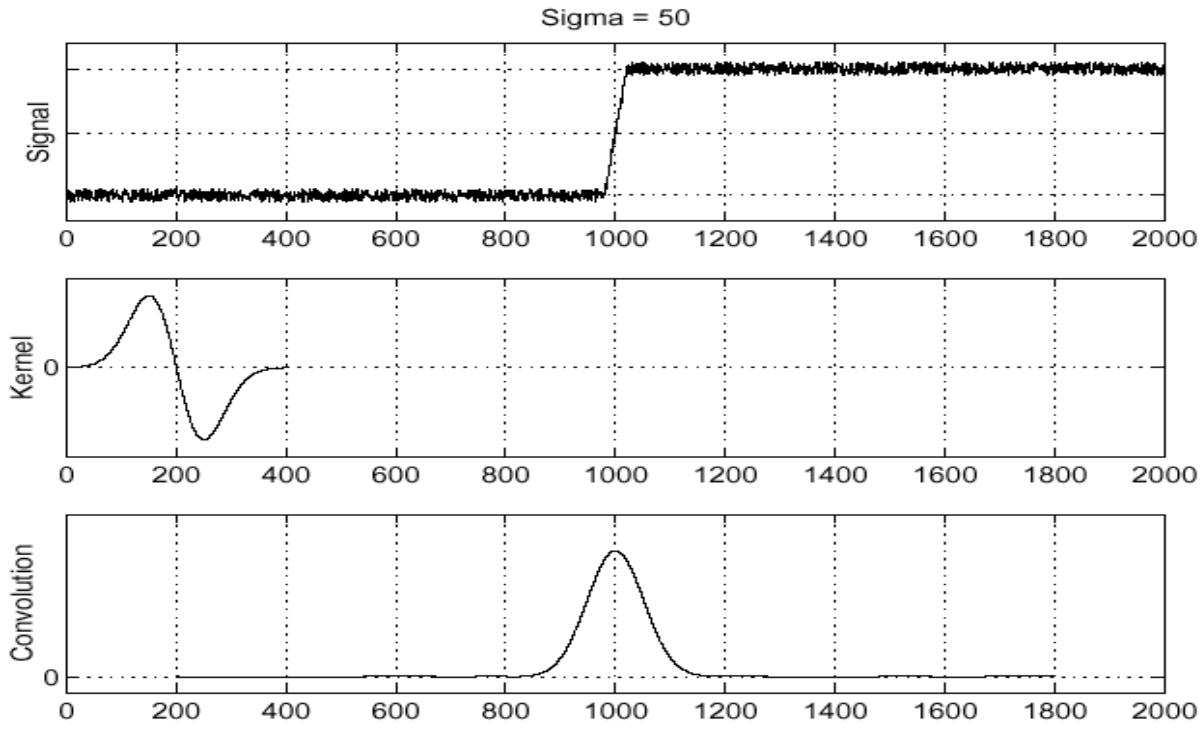
Effects of Noise

- Derivative of Convolution: $\frac{\partial h(x) * f(x)}{\partial x} = \frac{\partial h(x)}{\partial x} * f(x)$

$f(x)$

$\frac{\partial h(x)}{\partial x}$

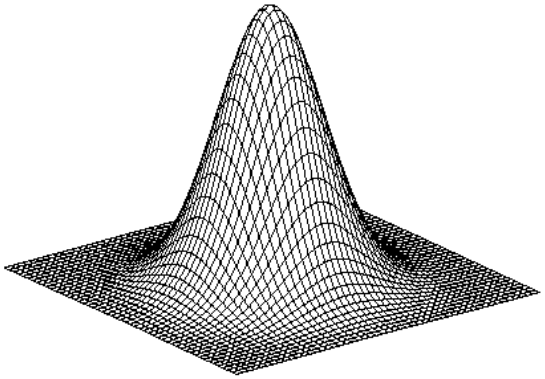
$\frac{\partial h(x)}{\partial x} * f(x)$



Laplacian of Gaussian (LoG)

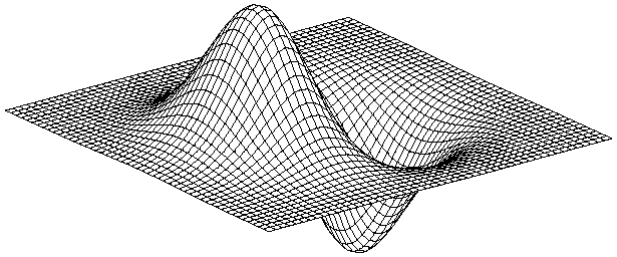
- Gaussian smoothing before applying Laplacian

$$LoG(x, y) = \frac{1}{\pi\sigma^4}\left[1 - \frac{(x^2 + y^2)}{2\sigma^2}\right]e^{\frac{-(x^2+y^2)}{2\sigma^2}}$$



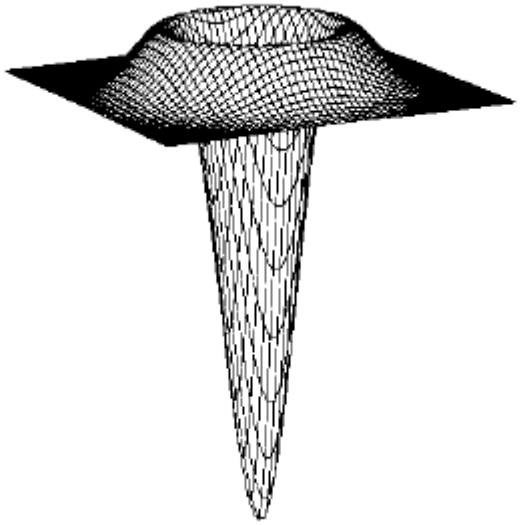
Gaussian

$$h(x, y) = \frac{1}{2\pi\sigma^2}e^{-\frac{x^2+y^2}{2\sigma^2}}$$



Derivative of Gaussian

$$\frac{\partial h(x, y)}{\partial x}$$



Laplacian of Gaussian

$$\nabla^2 h(x, y)$$

Laplacian of Gaussian (LoG)

- The wider the LoG function, the wider the edge that will be detected.
- The greater the value of σ, the wider the convolution mask necessary.

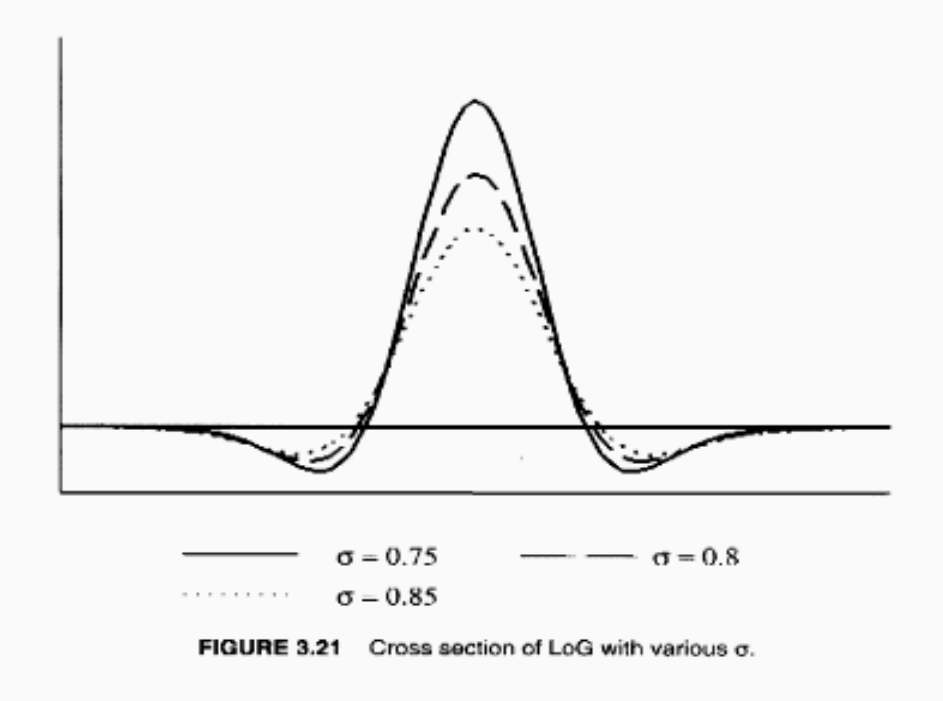


FIGURE 3.21 Cross section of LoG with various σ.

0	1	1	2	2	2	1	1	0
1	2	4	5	5	5	4	2	1
1	4	5	3	0	3	5	4	1
2	5	3	-12	-24	-12	3	5	2
2	5	0	-24	-40	-24	0	5	2
2	5	3	-12	-24	-12	3	5	2
1	4	5	3	0	3	5	4	1
1	2	4	5	5	5	4	2	1
0	1	1	2	2	2	1	1	0

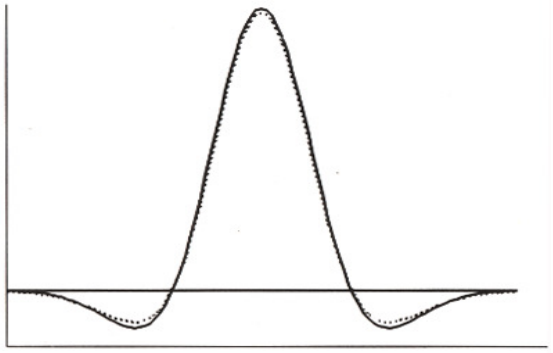
LoG Mask (σ=1.4)

Difference of Gaussians (DoG)

- The LoG ooperator requires large computation.
- Approximate LoG using two Gaussians:

$$DoG(x,y) = \frac{1}{2\pi\sigma_1^2} e^{\frac{-(x^2+y^2)}{2\sigma_1^2}} - \frac{1}{2\pi\sigma_2^2} e^{\frac{-(x^2+y^2)}{2\sigma_2^2}}, \text{ where } \sigma_1 > \sigma 2$$

- Can specify the width of edges to detect by varying the values of σ_1 and σ_2.
- Most similar to the LoG when $\sigma_1 = 1.6\sigma_2$.



LoG와 DoG 함수

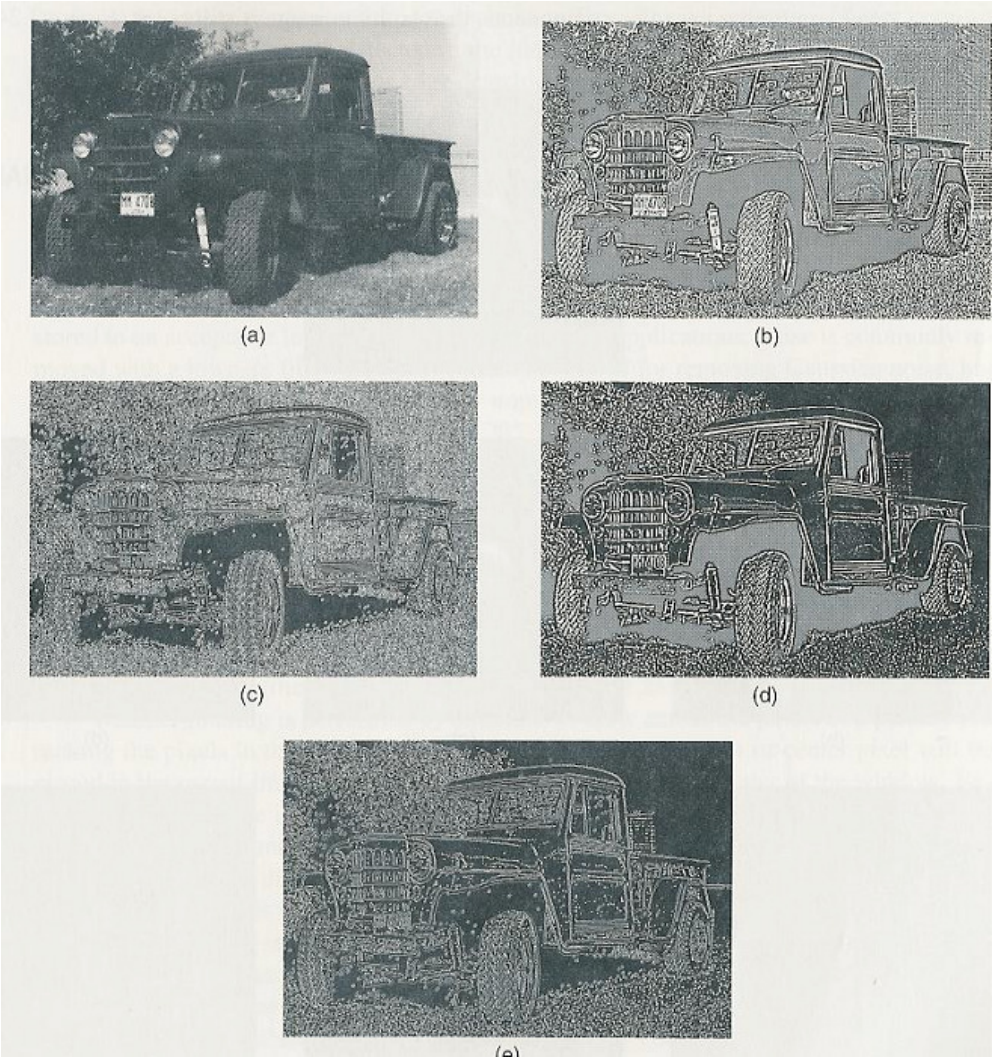
7×7 마스크

0	0	−1	−1	−1	0	0
0	−2	−3	−3	−3	−2	0
−1	−3	5	5	5	−3	−1
−1	−3	5	16	5	−3	−1
−1	−3	5	5	5	−3	−1
0	−2	−3	−3	−3	−2	0
0	0	−1	−1	−1	0	0

9×9 마스크

0	0	0	−1	−1	−1	0	0	0
0	−2	−3	−3	−3	−3	−3	−2	0
0	−3	−2	−1	−1	−1	−2	−3	0
−1	−3	−1	9	9	9	−1	−3	−1
−1	−3	−1	9	19	9	−1	−3	−1
−1	−3	−1	9	9	9	−1	−3	−1
0	−3	−2	−1	−1	−1	−2	−3	0
0	−2	−3	−3	−3	−3	−3	−2	0
0	0	0	−1	−1	−1	0	0	0

Difference of Gaussians (DoG)



(a) Original (b) 7x7 DoG (c) zero-crossings detected
(d) 9x9 DoG (e) zero-crossings detected.

- The image processed with the 9x9 mask shows wider edges.
- Notice that the edges in the edge maps are shown as closed curves.
  (A characteristic of edge detectors based on the Laplacian of Gaussain smoothing function.)

Color Edge Detection

- Type I
  - Detect the discontinuity in an image's luminance.
  - Edge detection is performed on the intensity channel of a color image in HSI space.

- Type II
  - Detect the discontinuity in RGB channels.
  - Edge detection is performed on each color component.
  - Color components can also be summed to create the gray scale edge map:

$$G(x, y) = \sqrt{G_{red}^2 + G_{green}^2 + G_{blue}^2}$$

Median Filtering

- Advantage:
  - Suited for removing impulse noise while preserving sharp edges and details.
  - Note: Lowpass filtering is suited for removing Gaussian noise, but not impulse noise.

- Three steps for median filtering:
  - Sort gray values of the pixels within a window
  - Find the median value
  - Replace the center pixel value with the median

Median Filtering

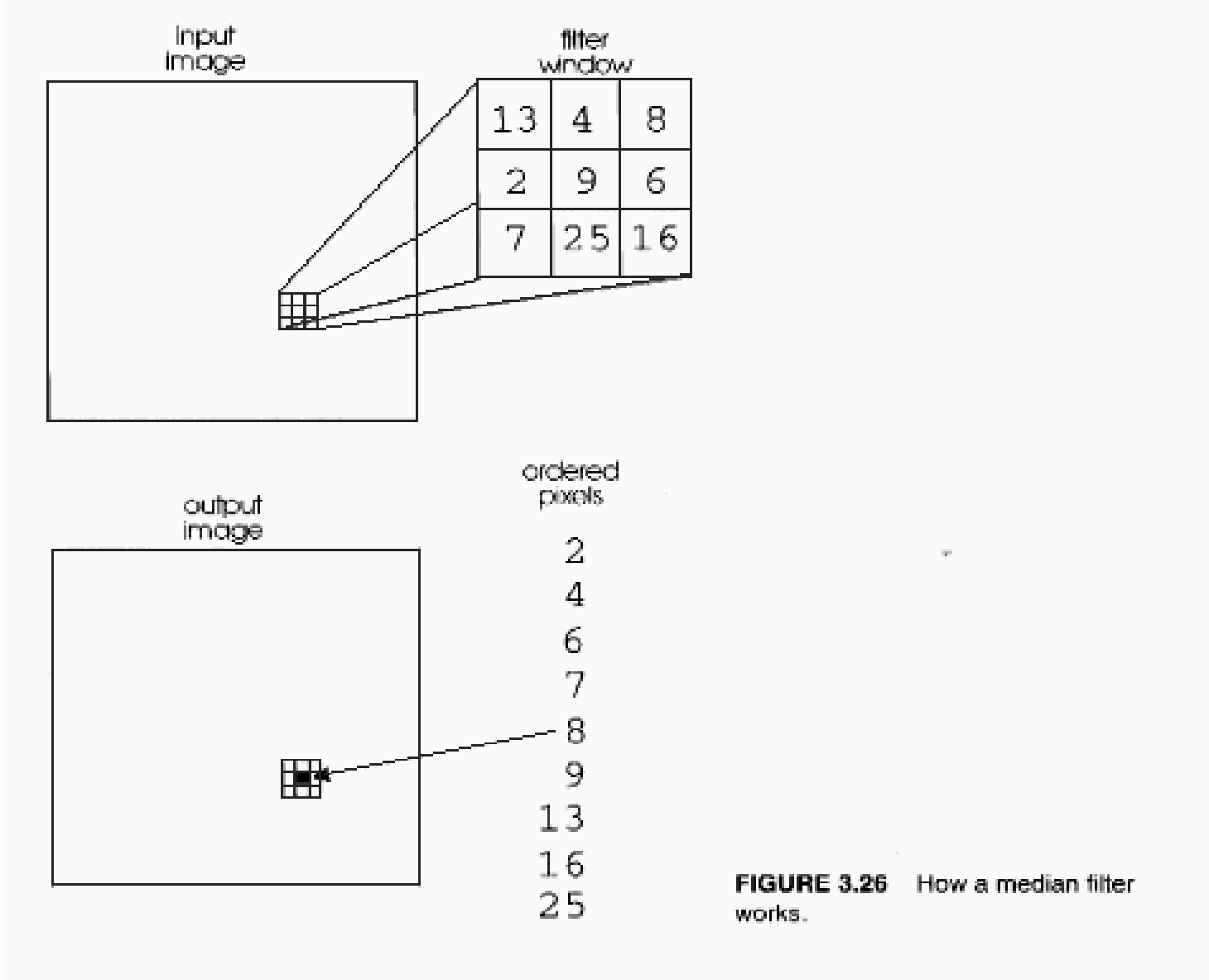


FIGURE 3.26 How a median filter works.

Median Filtering

- Median filter windows:

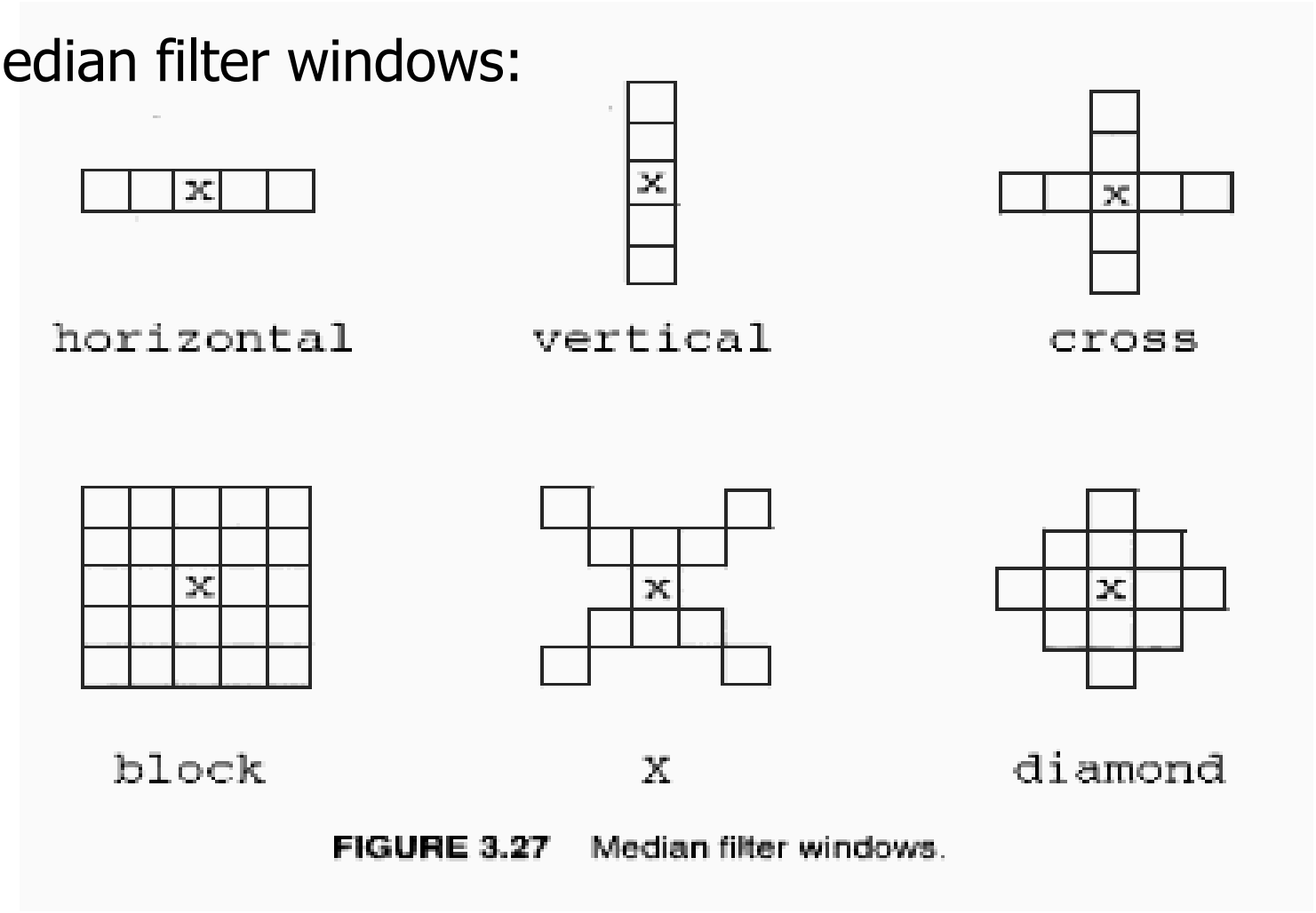


FIGURE 3.27 Median filter windows.

Median Filtering

Impulse noise

Gaussian filtered

Median filtered

Gaussian noise

Gaussian filtered

Median filtered

Median Filtering

Gaussian noise

3x3 Gaussian filtered

5x5 Gaussian filtered

7x7 Gaussian filtered

3x3 Median filtered

5x5 Median filtered

7x7 Median filtered

Color Median Filtering

- The problem with median filtering each color component separately:

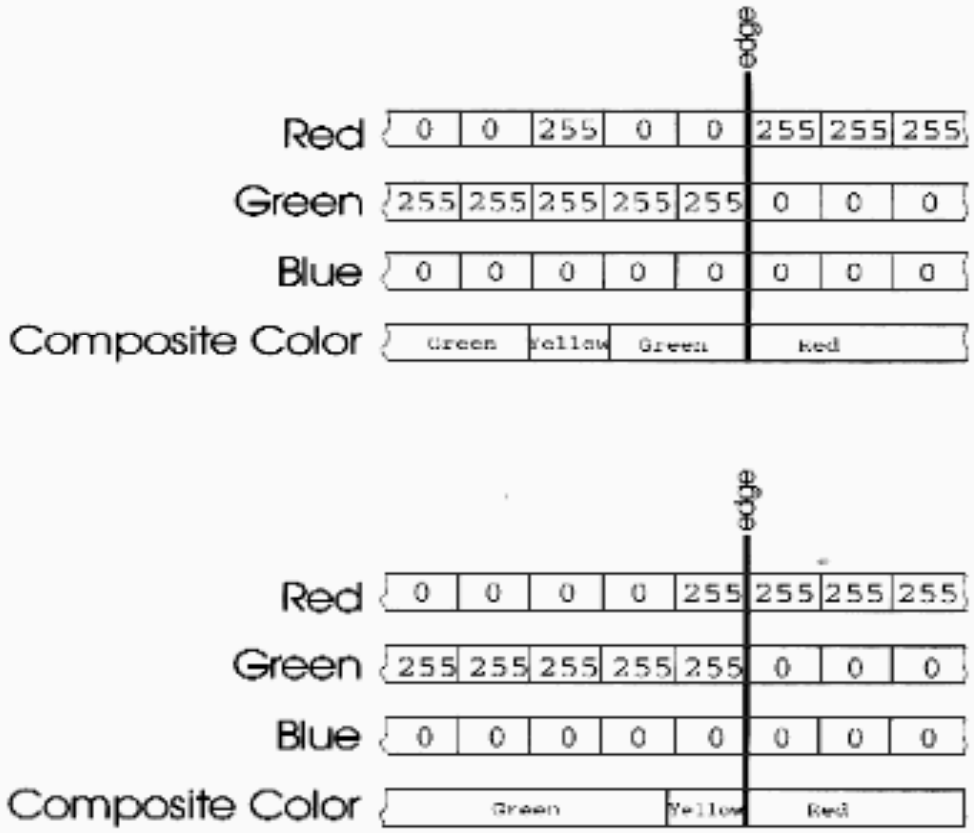


FIGURE 3.29 (a) Original RGB channels with yellow spike; (b) separate channels median filtered.

Color Median Filtering

- A unique property of median filtering:

$$\sum_{i=1}^{N} \left| x_{med} - x_i \right| \leq \sum_{i=1}^{N} \left| y - x_i \right|,$$

where N is the number of elements in the set, y an arbitrary value in the set, and x_{med} the median value.

- Break up x and y in the unique property into red, green, blue components ($x_i = \{red_i, green_i, blue_i\}$) :

$$Dist_i = \sum_{i=1}^{N} \left| red_i - red_j \right| + \sum_{i=1}^{N} \left| green_i - green_j \right| + \sum_{i=1}^{N} \left| blue_i - blue_j \right|,$$

where i is the pixel being processed and j represents the other pixel samples.

- The pixel x_i with smallest $Dist_i$ is the filter output.

Minimum/Maximum Filtering

- Replace center pixel with min or max window value.
- Minimum filter removes white spikes.
- Maximum filter removes dark spikes.
- Cascade of max and min filters remove white & dark spikes ("salt and pepper" noise)
  - Closing filter: max+min filter - removes dark spike without altering original.
  - Opening filter: min+max filter – removes white spike without altering original.