Recitation 4

Convolution

Agenda

1. Convolution – 1D and 2D
2. Image Derivatives
3. Fourier Transform
4. Sharpening
5. Edge Detection

Convolution – 1D and 2D

Image Derivatives

Fourier Transform

Sharpening

Edge Detection

1D Convolution

$$h(x) = (f * g)(x) = \sum_{i=0}^{N-1} f(i)g(x - i)$$

$$f = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \end{pmatrix} \qquad g = \begin{pmatrix} 0 & 0 & 1 & -1 & 0 \end{pmatrix}$$

Example

$$f * g$$

$$\begin{array}{c} 0 \\ \downarrow \\ f = (0 \quad 0 \quad 1 \quad 0 \quad 0) \end{array}$$

$$\begin{array}{c} g = (0 \quad 0 \quad 1 \quad -1 \quad 0) \\ \uparrow \\ 0 \end{array}$$

$$h(x) = (f * g)(x) = \sum_{i=0}^{N-1} f(i)g(x-i)$$

Example

$$f = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

$$g = \begin{pmatrix} 0 & -1 & 1 & 0 & 0 \end{pmatrix}$$

$$h(x) = (f * g)(x) = \sum_{i=0}^{N-1} f(i)g(x-i)$$

Example

$$f = (0 \quad 0 \quad 1 \quad 0 \quad 0)$$

$$g = (0 \quad -1 \quad 1 \quad 0 \quad 0)$$

$$h(0) = \sum_{i=0}^{N-1} f(i)g(-i) = f(0) \cdot g(0) + f(1) \cdot g(-1) + f(2) \cdot g(-2) \ldots$$

$$\boxed{h(x) = (f * g)(x) = \sum_{i=0}^{N-1} f(i)g(x-i)}$$

Example

$$f = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

$$g = \begin{pmatrix} 0 & -1 & 1 & 0 & 0 \end{pmatrix}$$

$$h(1) = \sum_{i=0}^{N-1} f(i)g(1-i) = f(0) \cdot g(1) + f(1) \cdot g(0) + f(2) \cdot g(-1) \ldots$$

$$\boxed{h(x) = (f * g)(x) = \sum_{i=0}^{N-1} f(i)g(x-i)}$$

Example

$$f = (0 \quad 0 \quad 1 \quad 0 \quad 0)$$

$$g = (0 \quad -1 \quad 1 \quad 0 \quad 0)$$

$$h(2) = \sum_{i=0}^{N-1} f(i)g(2-i) = f(0) \cdot g(2) + f(1) \cdot g(1) + f(2) \cdot g(0) \ldots$$

$$\boxed{h(x) = (f * g)(x) = \sum_{i=0}^{N-1} f(i)g(x-i)}$$

Example

$$\begin{array}{rccc} & 0 & & \\ & \downarrow & & \\ f = (& 1 & 0 & 1) \\ & & & \\ g = (& 0 & 2 & 1) \\ & \uparrow & & \\ & 0 & & \end{array}$$

$$h(x) = (f * g)(x) = \sum_{i=0}^{N-1} f(i)g(x-i)$$

Example

$$f = \begin{pmatrix} 1 & 0 & 1 \end{pmatrix}$$

$$g = \begin{pmatrix} 1 & 2 & 0 \end{pmatrix}$$

$$\boxed{h(x) = (f * g)(x) = \sum_{i=0}^{N-1} f(i)g(x-i)}$$

Example

$$f = (1 \quad 0 \quad 1)$$

$$g = (1 \quad 2 \quad 0)$$

$$h = (\overset{\overset{0}{\downarrow}}{0}$$

$$h(x) = (f * g)(x) = \sum_{i=0}^{N-1} f(i)g(x-i)$$

Example

$$f = (1 \quad 0 \quad 1)$$

$$g = (1 \quad 2 \quad 0)$$

$$h = (0 \quad \overset{\overset{1}{\downarrow}}{2}$$

$$\boxed{h(x) = (f * g)(x) = \sum_{i=0}^{N-1} f(i)g(x-i)}$$

Example

$$f = (1 \quad 0 \quad 1)$$

$$g = (1 \quad 2 \quad 0)$$

$$h = (0 \;\; 2 \;\; \overset{\overset{2}{\downarrow}}{1}$$

$$h(x) = (f * g)(x) = \sum_{i=0}^{N-1} f(i)g(x-i)$$

Example

$$f = (1 \quad 0 \quad 1)$$

$$g = (1 \quad 2 \quad 0)$$

$$h = (0 \;\; 2 \;\; 1 \;\; \overset{\overset{3}{\downarrow}}{2}$$

$$\boxed{h(x) = (f * g)(x) = \sum_{i=0}^{N-1} f(i)g(x-i)}$$

Example

$$f = (1 \quad 0 \quad 1)$$

$$g = (1 \quad 2 \quad 0)$$

$$h = (0 \;\; 2 \;\; 1 \;\; 2 \;\; \overset{\overset{4}{\downarrow}}{1})$$

$$\boxed{h(x) = (f * g)(x) = \sum_{i=0}^{N-1} f(i)g(x-i)}$$

Example

$$f = (1 \quad 0 \quad 1)$$
$$g = (1 \quad 2 \quad 0)$$

$$f = (1 \quad 0 \quad 1)$$
$$g = (1 \quad 2 \quad 0)$$

$$h(-1) = h(5) = 0$$

(Ignore)

$$h(x) = (f * g)(x) = \sum_{i=0}^{N-1} f(i)g(x-i)$$

Example

$$\begin{array}{c} \\ \\ f = (1 \end{array} \begin{array}{c} 0 \\ \downarrow \\ 0 \end{array} \begin{array}{c} \\ \\ 1) \end{array}$$

$$\begin{array}{c} g = (0 \\ \\ \\ \end{array} \begin{array}{c} 2 \\ \uparrow \\ 0 \end{array} \begin{array}{c} 1) \\ \\ \\ \end{array}$$

$$\begin{array}{cccccc} & -2 & & 0 & & 2 \\ & \downarrow & & \downarrow & & \downarrow \\ h = (& 0 & 2 & 1 & 2 & 1) \end{array}$$

The same values as before!

$$h(x) = (f * g)(x) = \sum_{i=0}^{N-1} f(i)g(x-i)$$

Convolution Properties

Commutative $$f * g = g * f$$

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

Distributive $$f * (g + h) = f * g + f * h$$

A convolution is a **linear operator**: it can be represented in matrix form as

$$f * g \equiv Gf$$

Convolution Matrix

$$f = (1 \quad 0 \quad 1) \qquad g = (0 \quad 2 \quad 1)$$

$$f * g \equiv Gf = \begin{bmatrix} 0 & 0 & 0 \\ 2 & 0 & 0 \\ 1 & 2 & 0 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 2 \\ 1 \\ 2 \\ 1 \end{bmatrix} \Rightarrow (0 \quad 2 \quad 1 \quad 2 \quad 1)$$

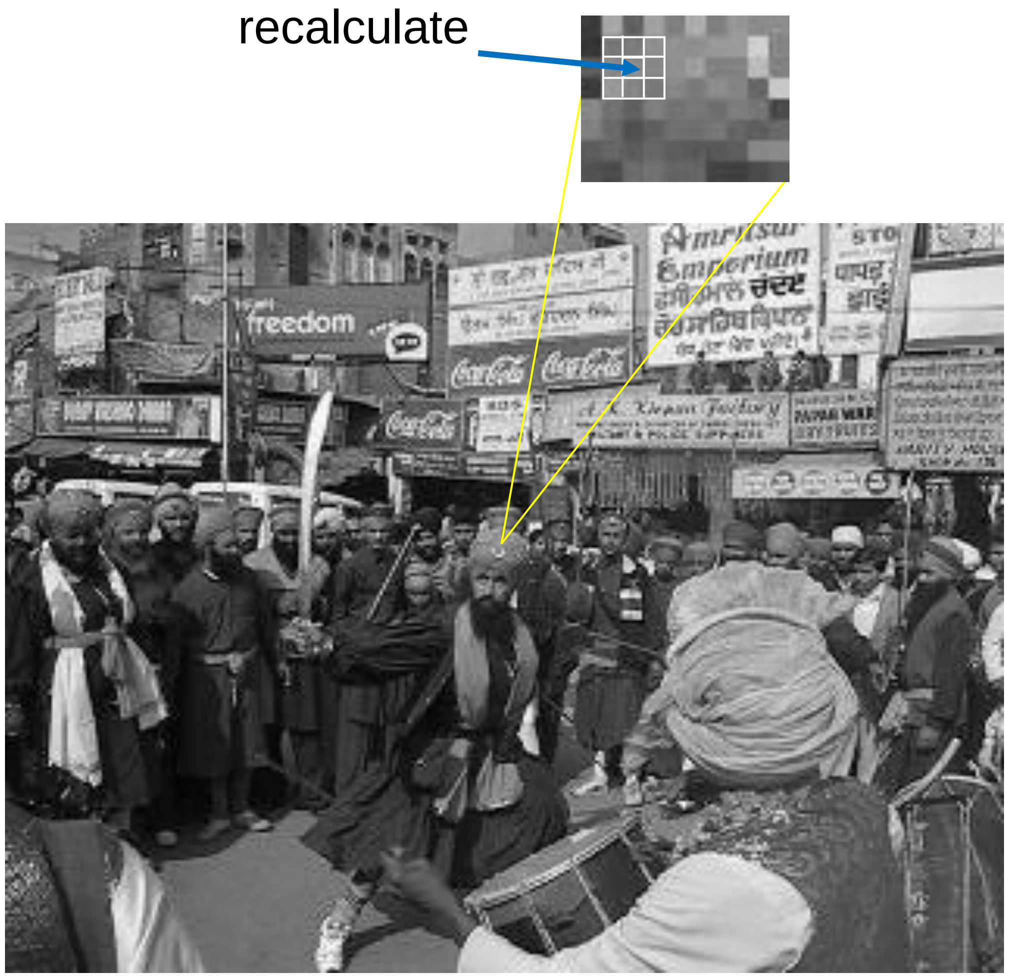
2D Convolution

$$f(x, y) * g(x, y) = \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} f(x-i, y-j)g(i, j)$$

Image

Filter, Kernel

Application: Spatial Filtering



$w(-1,-1)$	$w(-1,0)$	$w(-1,1)$
$w(0,-1)$	$w(0,0)$	$w(0,1)$
$w(1,-1)$	$w(1,0)$	$w(1,1)$

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

$$g(30,78) = \sum_{i=-a}^{a} \sum_{j=-b}^{b} w(i,j) f(30-i, 78-j)$$

Smoothing by Spatial Filtering

1/9 *

1	1	1
1	1	1
1	1	1

Smoothing by Spatial Filtering

1/16 *

1	2	1
2	4	2
1	2	1

Smoothing by Spatial Filtering

Using a larger kernel

What will happen?

	0	0	0
*	0	1	0
	0	0	0

Identity

$$* \quad \begin{array}{|c|c|c|} \hline 0 & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array}$$

What will happen?

	0	0	0
*	1	0	0
	0	0	0

Shifting

	0	0	0
*	1	0	0
	0	0	0

Convolution – 1D and 2D

Image Derivatives

Fourier Transform

Sharpening

Edge Detection

Derivative

$$\frac{\partial}{\partial rows} f(i,j) = \frac{\partial}{\partial i} f(i,j) = \lim_{\epsilon \to 0} \frac{f(i,j) - f(i-\epsilon,j)}{\epsilon}$$

columns

rows

$\epsilon = 1$ pixel

1st Derivative Approximation

$$\frac{\partial}{\partial rows} f(i,j) \cong f(i,j) - f(i-1,j)$$

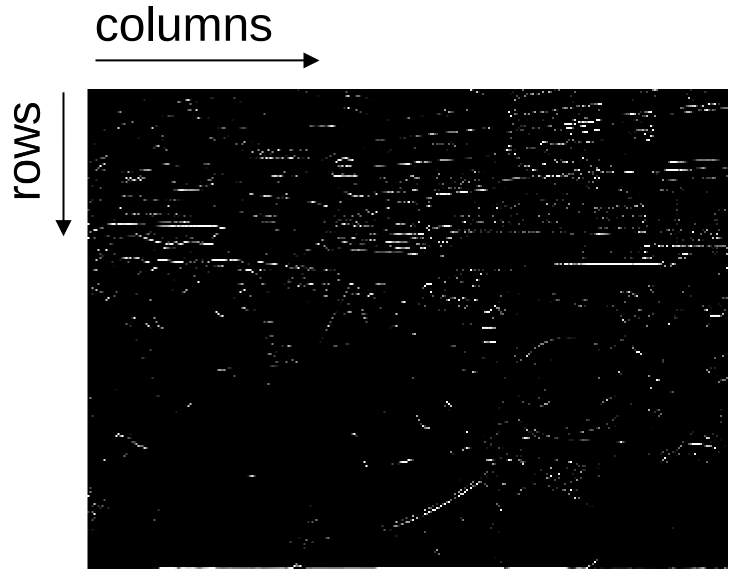
columns

rows

Implement:
convolution with $\begin{pmatrix} 1 \\ -1 \end{pmatrix}$

1^{st} Derivative Approximation

$$\frac{\partial}{\partial rows} f(i,j) \cong f(i,j) - f(i-1,j)$$



Implement: convolution with $\begin{pmatrix} 1 \\ -1 \end{pmatrix}$

1st Derivative Approximation

$$\frac{\partial}{\partial cols} f(i,j) \cong f(i,j) - f(i,j-1)$$

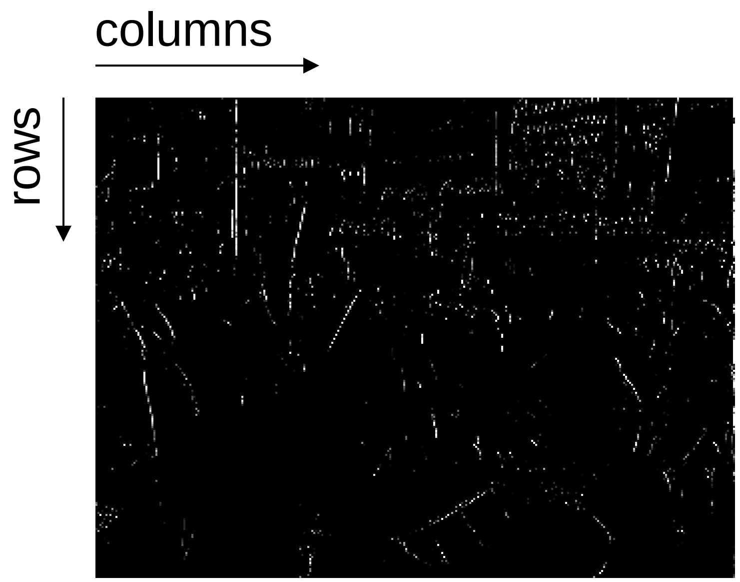
columns

rows

Implement convolution with $\begin{pmatrix} 1 & -1 \end{pmatrix}$

1st Derivative Approximation

$$\frac{\partial}{\partial cols} f(i,j) \cong f(i,j) - f(i,j-1)$$



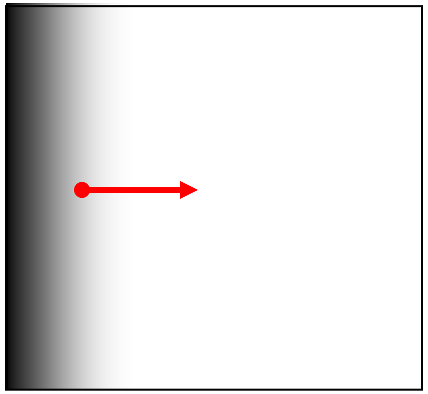
Implement convolution with $\begin{pmatrix} 1 & -1 \end{pmatrix}$

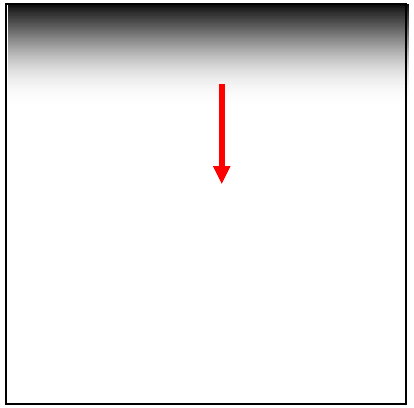
The Gradient

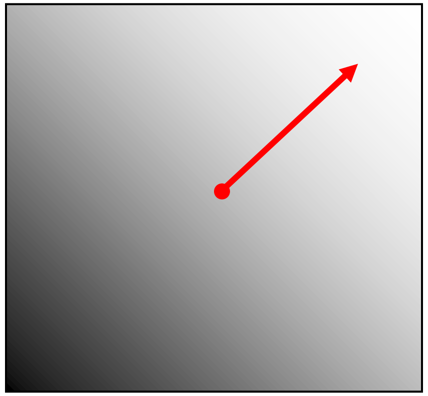
The vector of partial derivatives.
Points in the direction of most rapid change in intensity

$$\nabla f = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right)$$

$$\nabla f = \left[\frac{\partial f}{\partial x}, 0 \right] \qquad \nabla f = \left[0, \frac{\partial f}{\partial y} \right] \qquad \nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right]$$







big gradient values = big changes

The Gradient - Properties

Magnitude

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

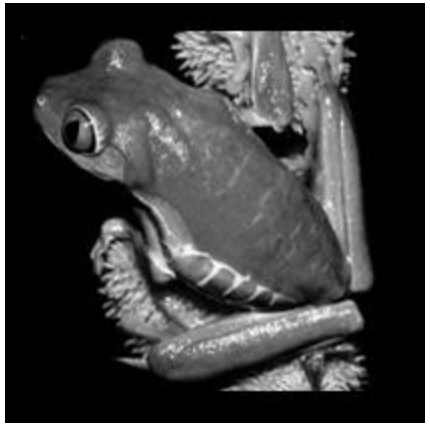
Direction (orientation angle)

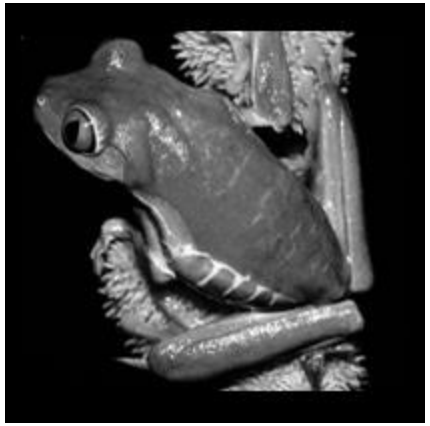
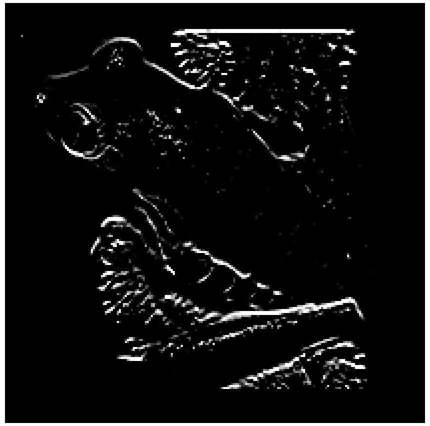
$$\alpha = atan2\left(\frac{\partial f}{\partial y}, \frac{\partial f}{\partial x}\right)$$

Directional derivative

$$\cos(\alpha)\frac{\partial f}{\partial x} + \sin(\alpha)\frac{\partial f}{\partial y}$$

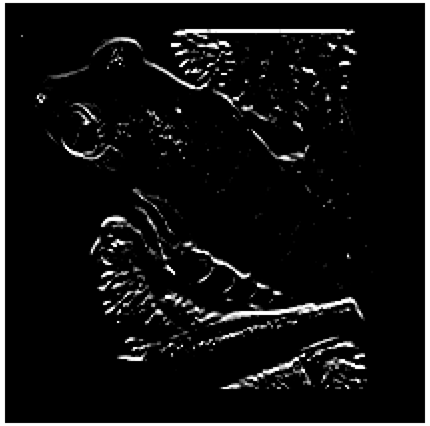
Example

$I_X =$  $* \begin{pmatrix} -1 & 0 & 1 \end{pmatrix} =$ 

$I_Y =$  $* \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix} =$ 

Example

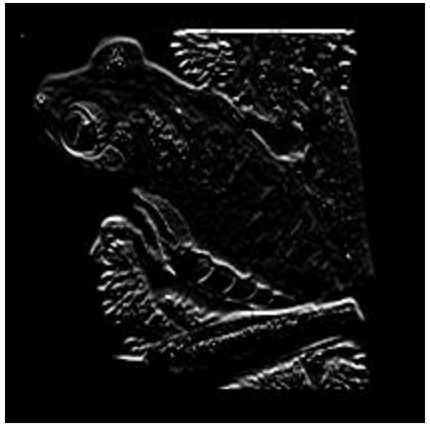
$$\nabla f = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right) =$$

( , )

Example: Gradient Magnitude vs. Derivatives

$I_X =$

$I_Y =$



$\sqrt{I_X^2 + I_Y^2} =$

2^{nd} Derivative Approximation

$$\frac{\partial^2}{\partial x^2} f(i,j) \cong f(i-1,j) + f(i+1,j) - 2f(i,j)$$

Implement convolution with $\begin{pmatrix} 1 & -2 & 1 \end{pmatrix}$

Check that: $\begin{bmatrix} 1 & \underset{\substack{\uparrow \\ 0}}{-1} \end{bmatrix} * \begin{bmatrix} \underset{\substack{\uparrow \\ 0}}{1} & -1 \end{bmatrix} = \begin{bmatrix} 1 & \underset{\substack{\uparrow \\ 0}}{-2} & 1 \end{bmatrix}$

Reminder - 1D Fourier Transform

- Moving from the ***time*** domain to the ***frequency*** domain
- Discrete Fourier transform (DFT):

$$F(\omega) = \sum_{x=0}^{N-1} f(x)\, e^{-\frac{2\pi i x \omega}{N}}$$

- Inverse Discrete Fourier transform (IDFT):

$$f(x) = \frac{1}{N} \sum_{\omega=0}^{N-1} F(\omega)\, e^{\frac{2\pi i x \omega}{N}}$$

Reminder - Image derivatives using FT

Image derivative is the inverse FT of the **weighted** frequency domain.

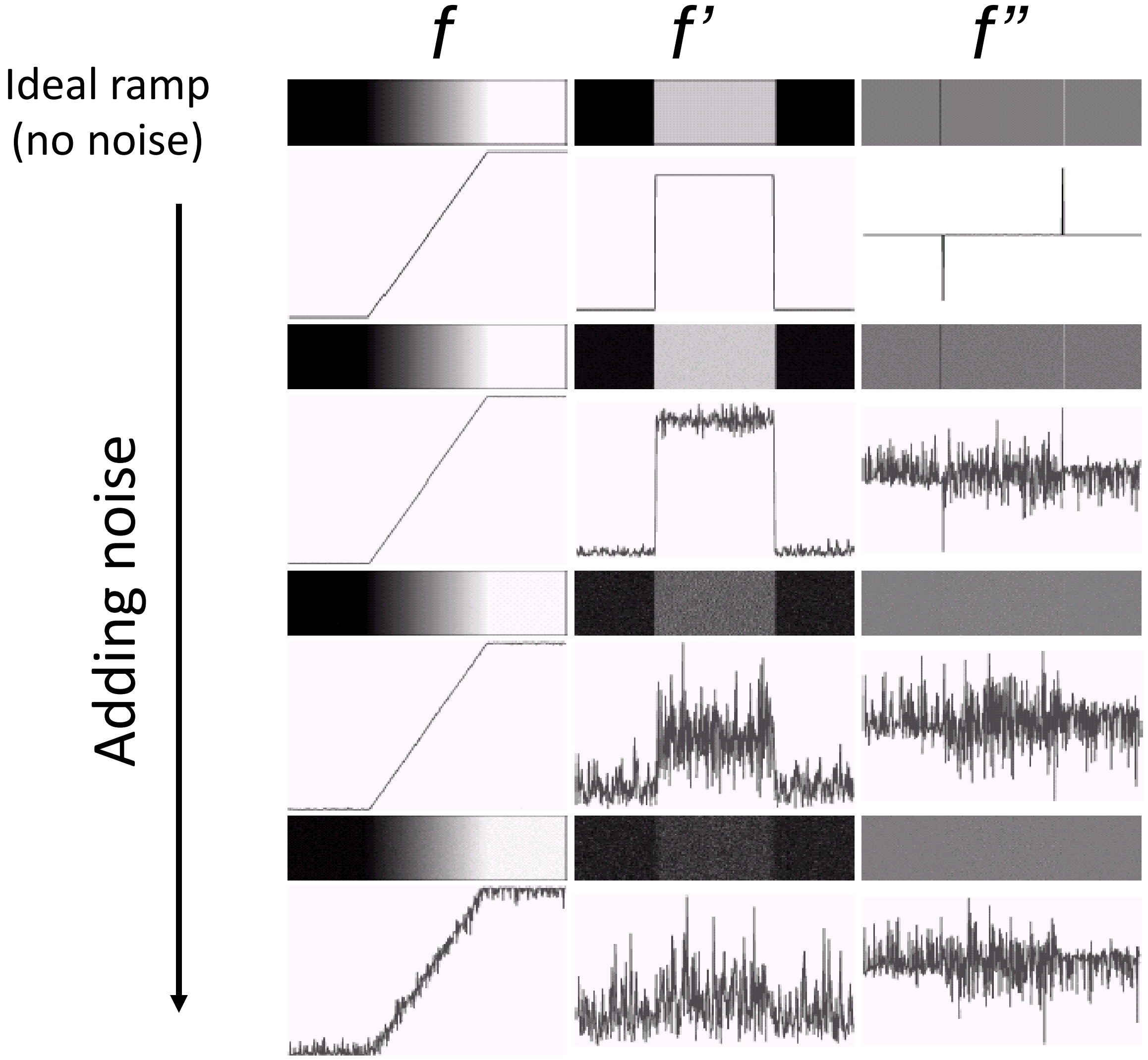
High frequencies affect the image derivative more than low frequencies.

Noise has more high frequency than normal image.

$$\frac{\partial f(x,y)}{\partial x} = \frac{2\pi i}{N} \cdot \Phi^{-1}\Big(u \cdot \Phi\big(f(x,y)\big)\Big)$$

Influence of Noise on Derivatives

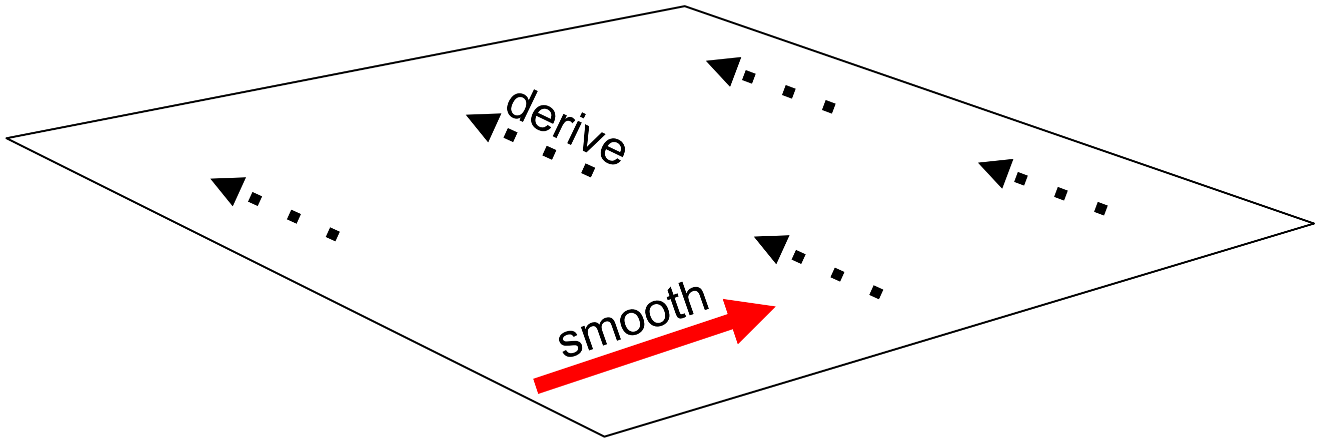
Noise strongly affects the derivatives, in greater proportion than it does the original image



Improving the Derivative Approximation

Problem: We would like to reduce the effects of noise on the derivative, but symmetric smoothing may eliminate the derivative response altogether.

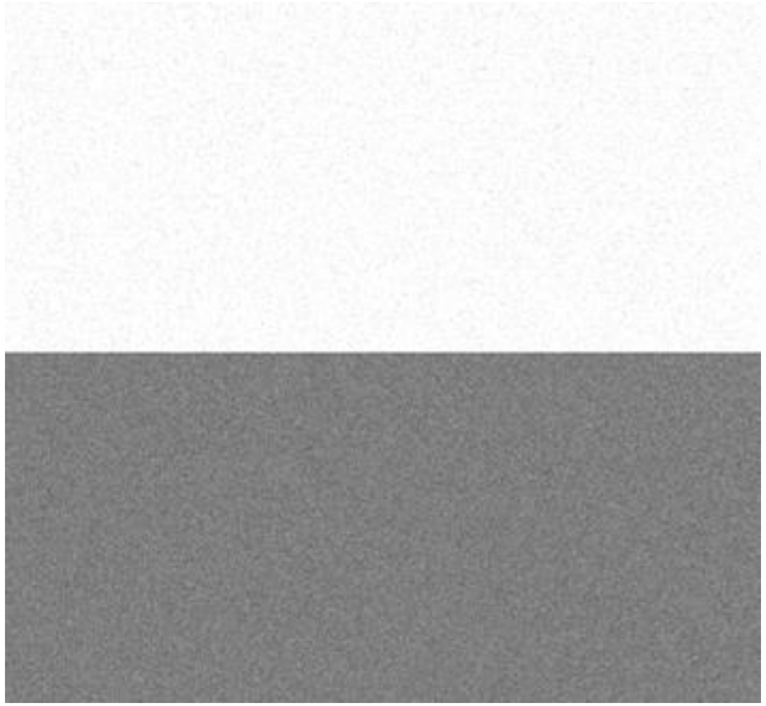
Idea (Sobel): Smooth the image in one direction, and compute the derivative in the orthogonal direction.



Improving the Derivative Approximation

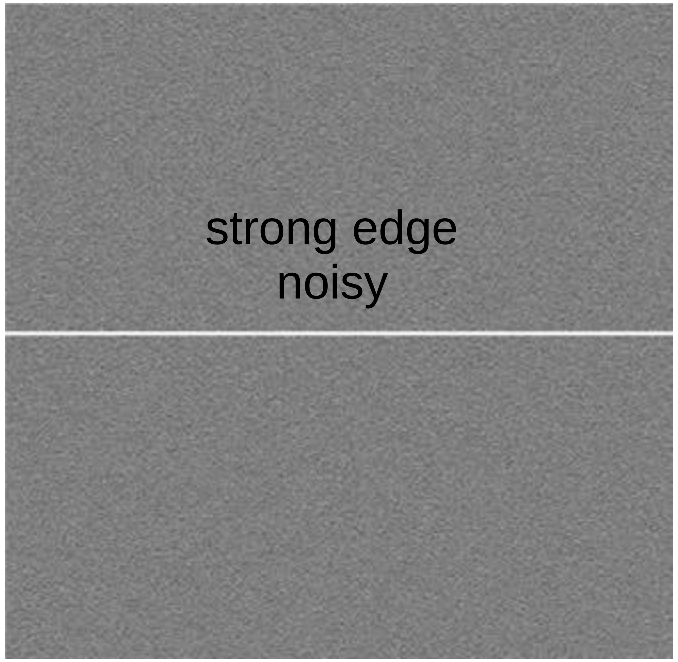
input

noisy input

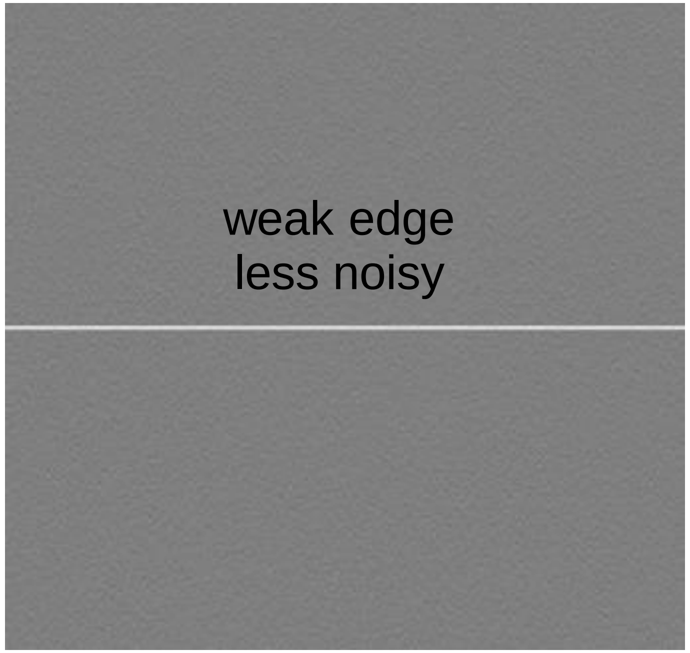


Improving the Derivative Approximation

derivative only

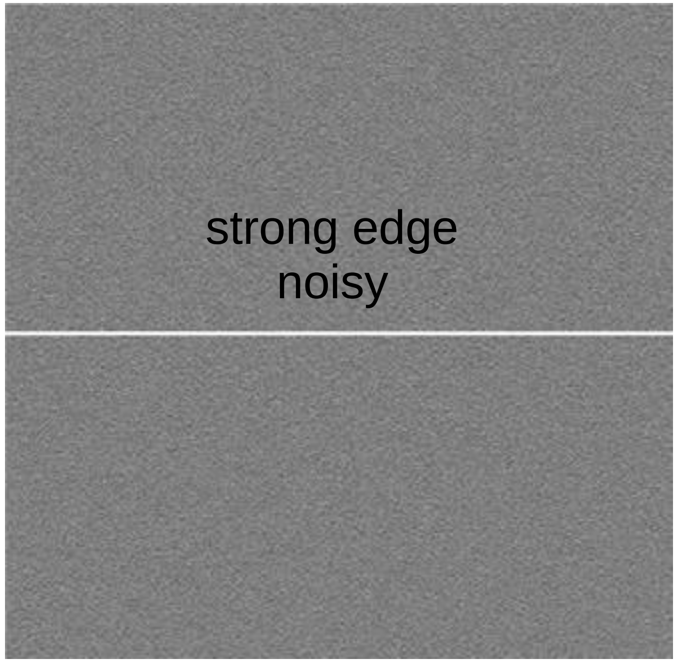


derivative+smooth in the same direction

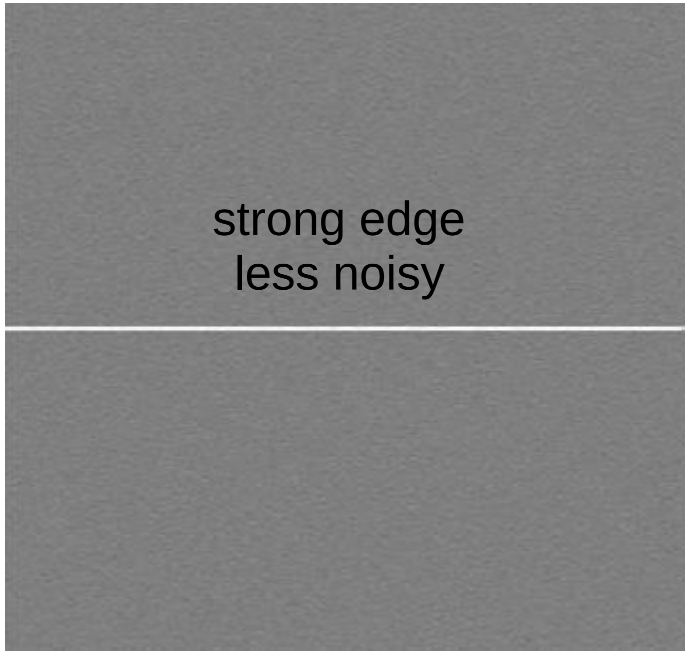


Improving the Derivative Approximation

derivative only



derivative+smooth in the orthogonal direction



Improving the Derivative Approximation

Sobel kernels:

$$\frac{\partial f}{\partial x} = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \qquad \frac{\partial f}{\partial y} = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

- Note:

$$f * \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & -1 \end{bmatrix} \quad = \quad f * \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$$

Convolution – 1D and 2D

Image Derivatives

Fourier Transform

Sharpening

Edge Detection

The Convolution Theorem

The Convolution Theorem:

$$\Phi(f * g) = F \cdot G$$

$$\Phi(f \cdot g) = F * G$$

Convolution Vs. Fourier

Convolution by Fourier:

$$\boxed{f * g = \Phi^{-1}(F \cdot G)}$$

Complexity (using the **FFT algorithm**): $O(NlogN)$, where *N* is the number of pixels in the image.

- Different Fourier transform phenomena can be explained by convolution, and vice versa.
- The Fourier interpretation is used for designing convolution filters.

Resemblance to Convolution

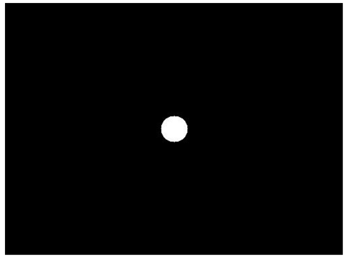
Fourier Filter

$$F(u,v) \cdot H(u,v)$$

Convolution Filter

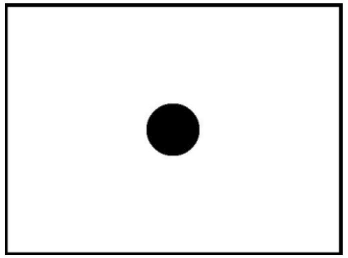
$$f(x,y) * g(x,y)$$

Example: Low-pass filter



$$\frac{1}{16}\begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}$$

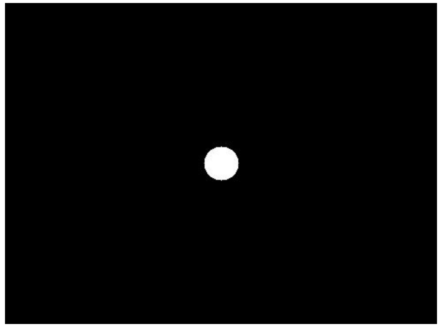
Example: High-pass filter



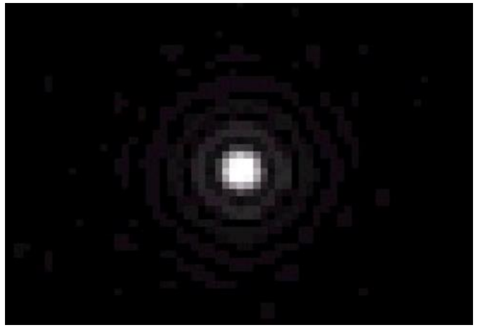
$$\begin{pmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{pmatrix}$$

Why do we get the rings?

$$\Phi^{-1}(F \cdot G) = f * g$$



Frequency domain
(dot product)



Spatial domain
(convolution)

1D Simplification:

$$rect(ax) \xrightarrow{\text{Fourier}} \frac{1}{|a|}\operatorname{sinc}\left(\frac{u}{a}\right)$$

Cross-Correlation

Definition: $$(f \otimes g)\,(x, y) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} f^*(i, j) g(x + i, x + j)$$

- Cross Correlation measures the **similarity of two signals** over all possible translations between them.
- Commonly used for **matching:**
  - Motion between two similar images
  - Small template in a large image

$$\left(f \otimes g\right)(x, y) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} f^*(i, j)\; g(x + i, y + j)$$

The Cross-Correlation Theorem

Definition: $$(f \otimes g)\,(x, y) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} f^*(i, j) g(x + i, x + j)$$

- The Cross-Correlation Theorem:

$$\Phi(f \otimes g) = F^* \cdot G$$

(follows from the Convolution Theorem)

Convolution – 1D and 2D

Image Derivatives

Fourier Transform

Sharpening

Edge Detection

Image Enhancement

Histogram enhancement	Histogram equalization
Noise reduction	Smoothing Median filtering

Sharpening

Sharpening via Laplacian Subtraction

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

The Laplacian in matrix form: $$\begin{bmatrix} 0 & 0 & 0 \\ 1 & -2 & 1 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 \\ 0 & -2 & 0 \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

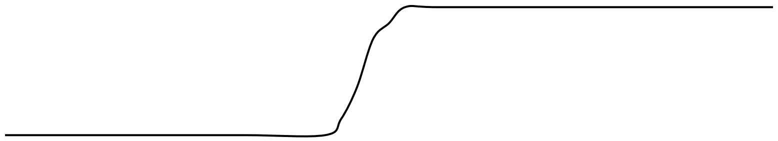
Subtracting from the image: $$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Sharpening via Laplacian Subtraction

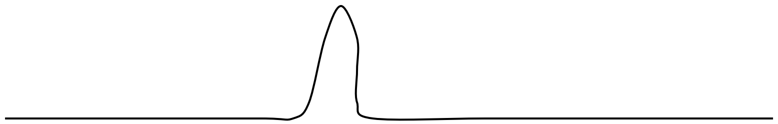
Sharpening via Laplacian Subtraction

Sharpening via Laplacian Subtraction

$f(x)$

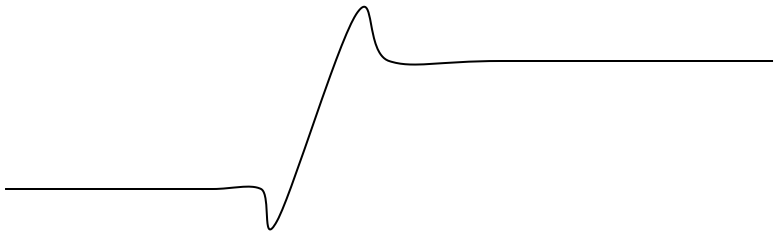


$f'(x)$



$f''(x)$

$f(x) - f''(x)$



Sharpening Example

Details are enhanced but so is the noise

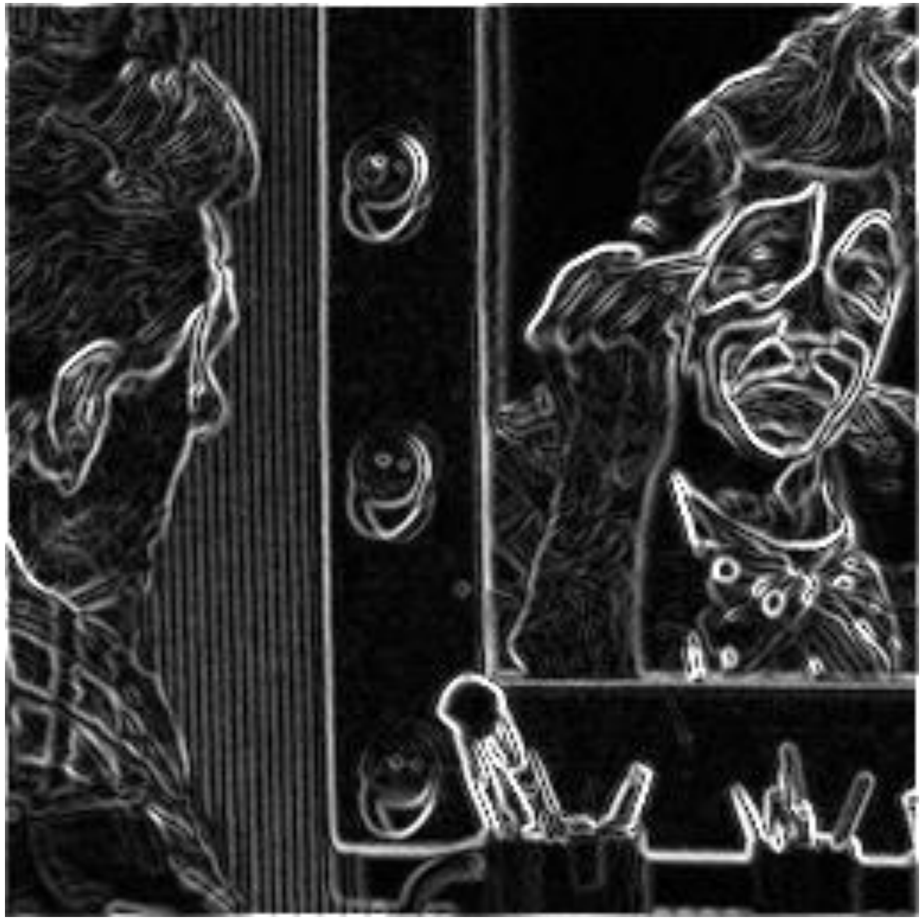
Convolution – 1D and 2D
Image Derivatives
Fourier Transform
Sharpening

Edge Detection

How Can the Gradient be Used For Edge Detection?

Original

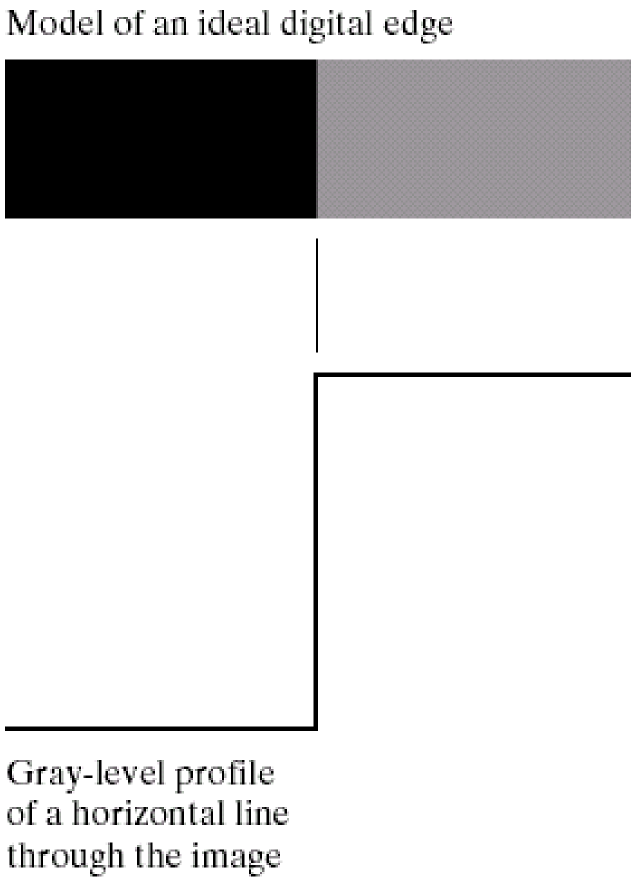
Gradient



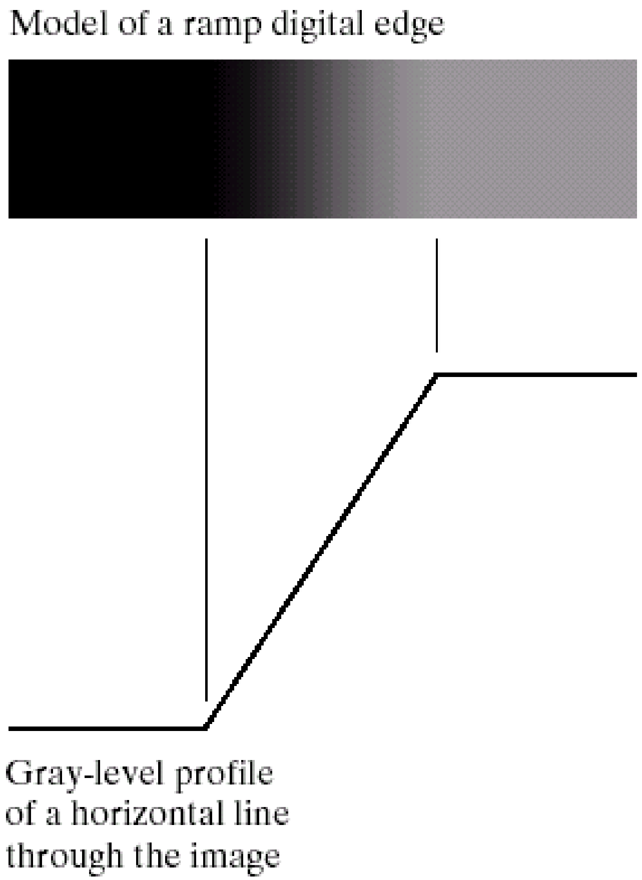
Edge Detection

Binary image where “1” resembles an edge

Ideal Edges

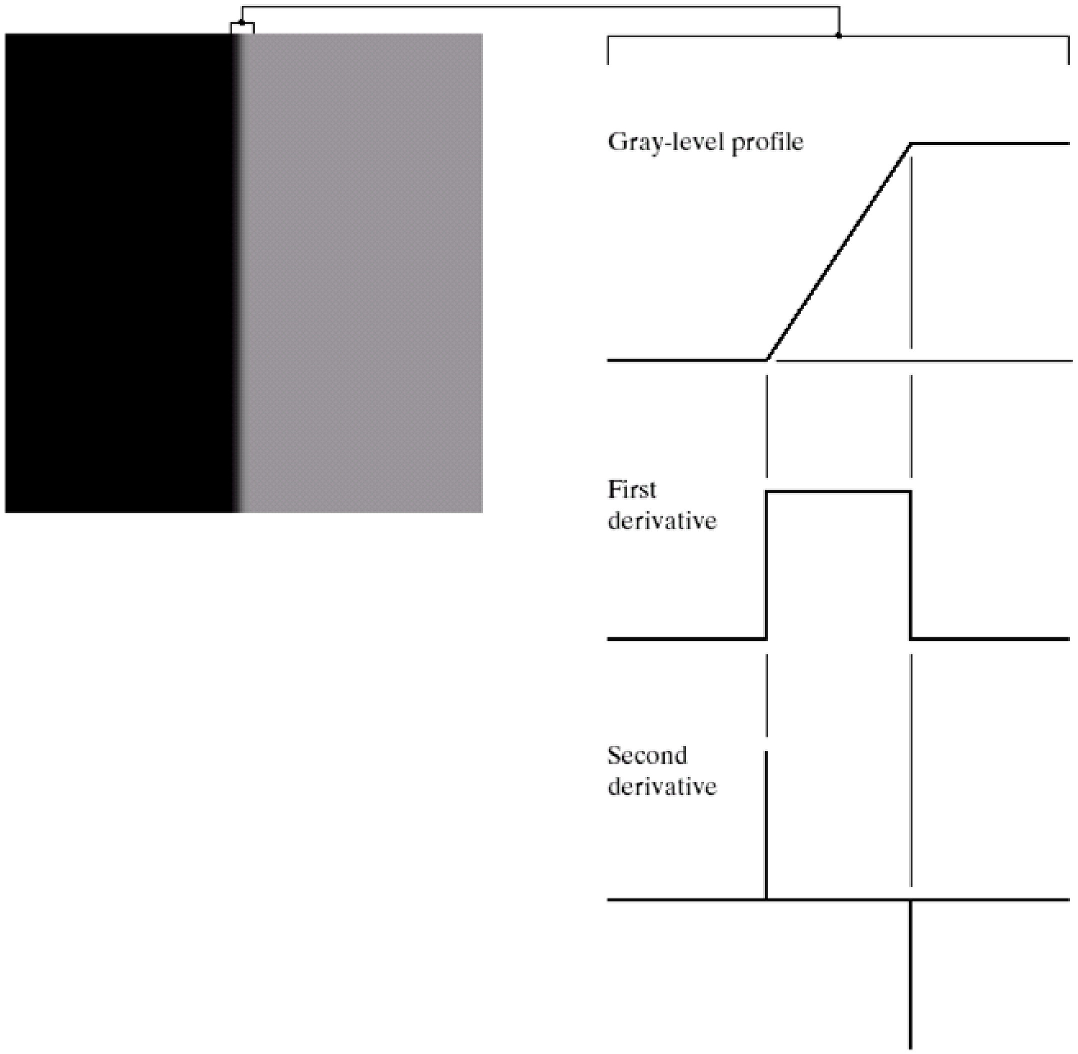


Model of an ideal digital edge
Gray-level profile
of a horizontal line
through the image

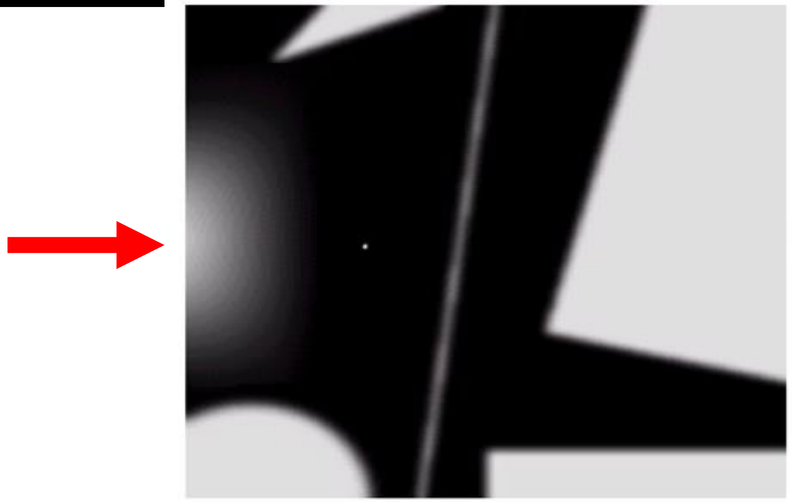


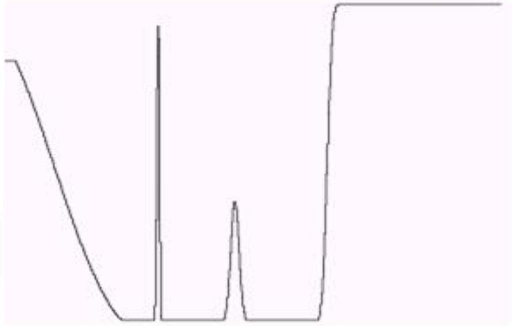
Model of a ramp digital edge
Gray-level profile
of a horizontal line
through the image

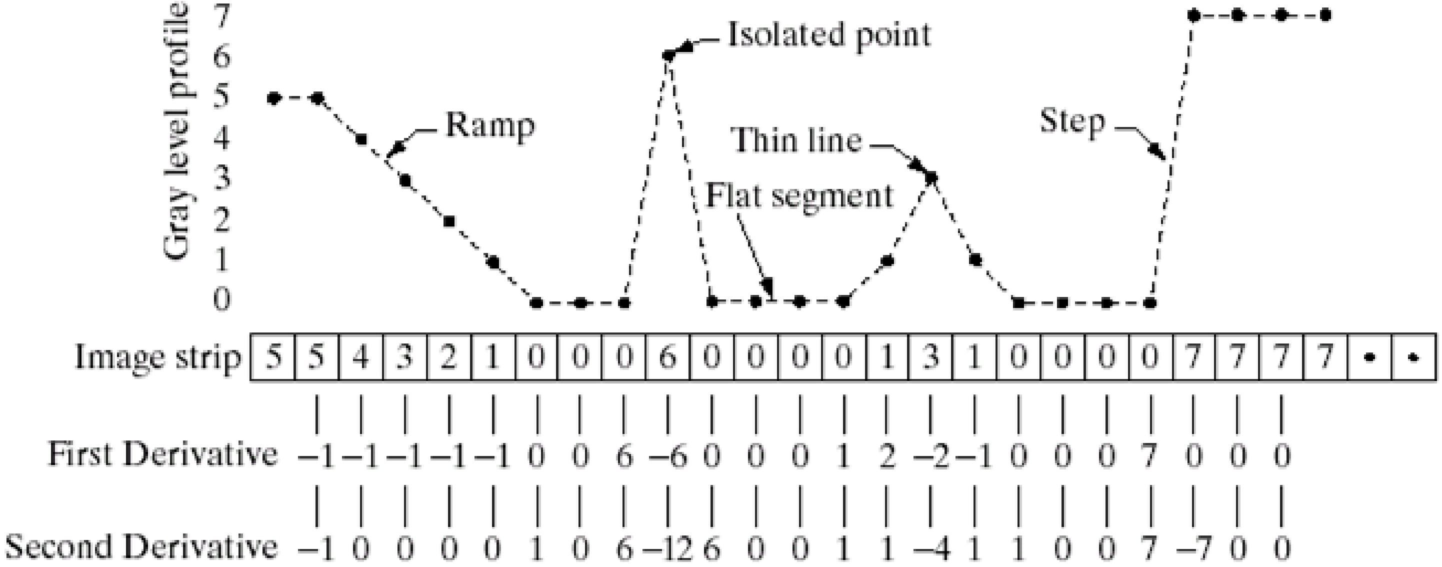
Derivative Response at Edges



Derivative - Numeric Example

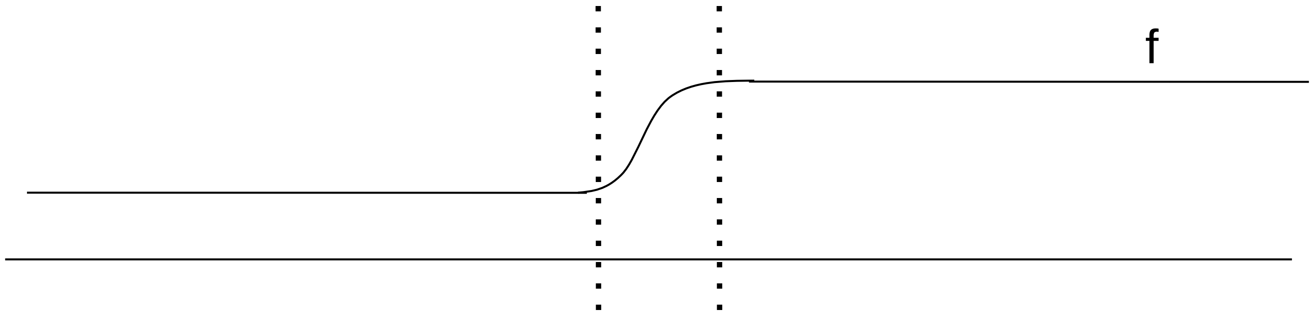




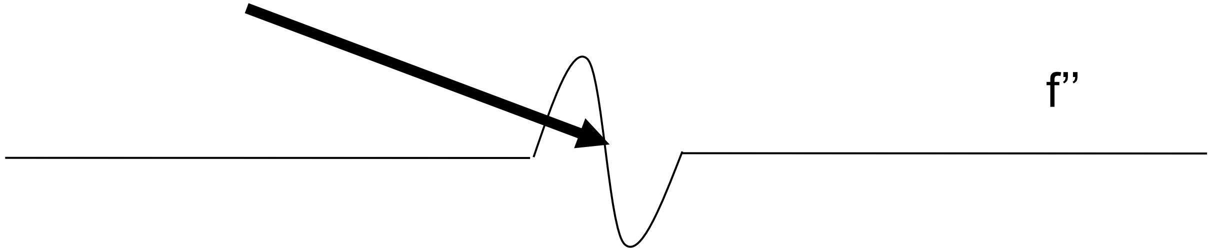


Edge Localization – Laplacian Zero Crossing

Where <u>exactly</u> is the edge ?



At zero crossing of *f*''



Exam

Next week:

2D Transformations