
Image Processing

CSE 457, Autumn 2003

Graphics

<http://www.cs.washington.edu/education/courses/457/03au/>

Reading

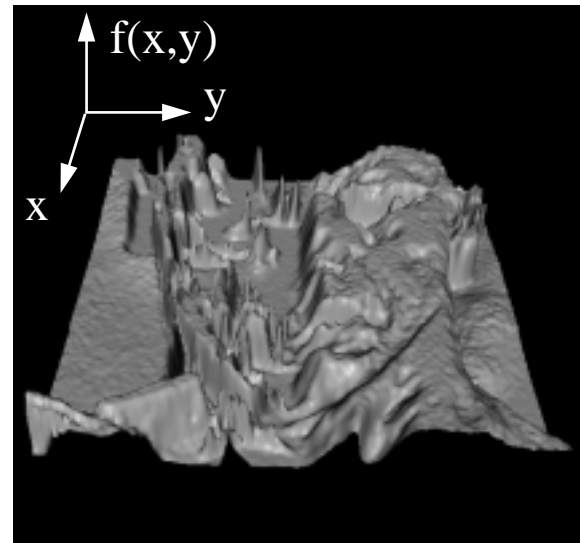
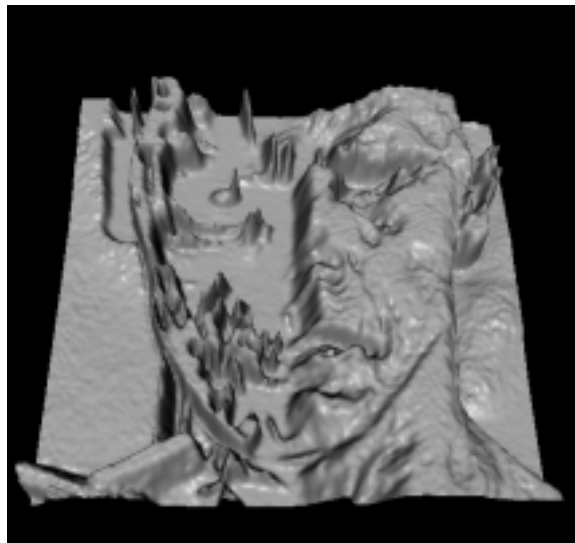
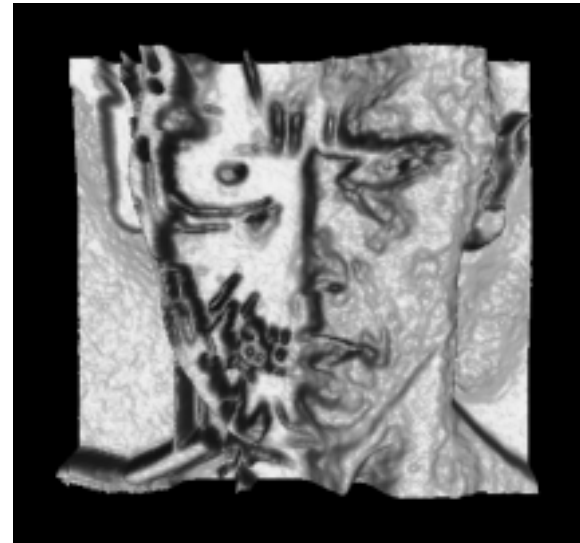
- Sections 4.2-4.4, 4.5(intro), 4.5.5, 4.5.6, 5.1-5.4. in *Machine Vision*, Jain, Kasturi, Schunck.
 - » on reserve in Engineering Library

What is an image?

- We can think of an **image** as a function, f , from \mathbb{R}^2 to \mathbb{R} :
 - » $f(x, y)$ gives the intensity of a channel at position (x, y)
 - » Realistically, we expect the image only to be defined over a rectangle, with a finite range:
 - $f: [a,b] \times [c,d] \rightarrow [0,1.0]$
- A color image is just three functions pasted together. We can write this as a “vector-valued” function:

$$\vec{f}(x, y) = \begin{bmatrix} r(x, y) \\ g(x, y) \\ b(x, y) \end{bmatrix}$$

Images as functions



What is a digital image?

- In computer graphics, we usually operate on **digital (discrete)** images:
 - » **Sample** the space on a regular grid
 - » **Quantize** each sample (round to nearest integer)

If our samples are Δ apart, we can write this as:

$$f[i, j] = \text{Quantize}\{ f(i \Delta, j \Delta) \}$$

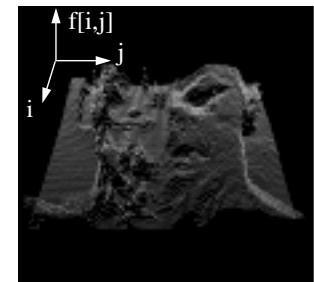
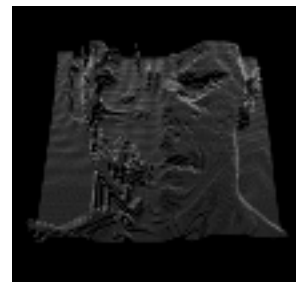
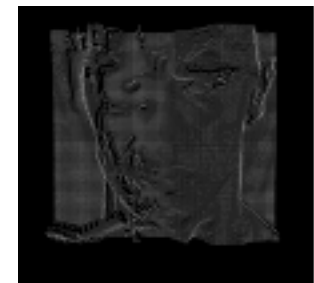


Image processing

- An **image processing** operation typically defines a new image g in terms of an existing image f .
- The simplest operations are those that transform each pixel in isolation. These pixel-to-pixel operations can be written as $g(x,y)=t(f(x,y))$
- Examples:
 - » threshold: emphasize a particular transition level
 - » RGB \rightarrow grayscale: extract the luminance for the pixel

A typical choice for mapping to grayscale is to apply the YIQ television matrix and keep the Y.

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.523 & 0.311 \end{bmatrix} * \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

single pixel changes



to grayscale



threshold at 55



threshold at 128

Pixel movement

- Some operations preserve intensities, but move pixels around in the image $g(x, y) = f(\tilde{x}(x, y), \tilde{y}(x, y))$



example: image registration

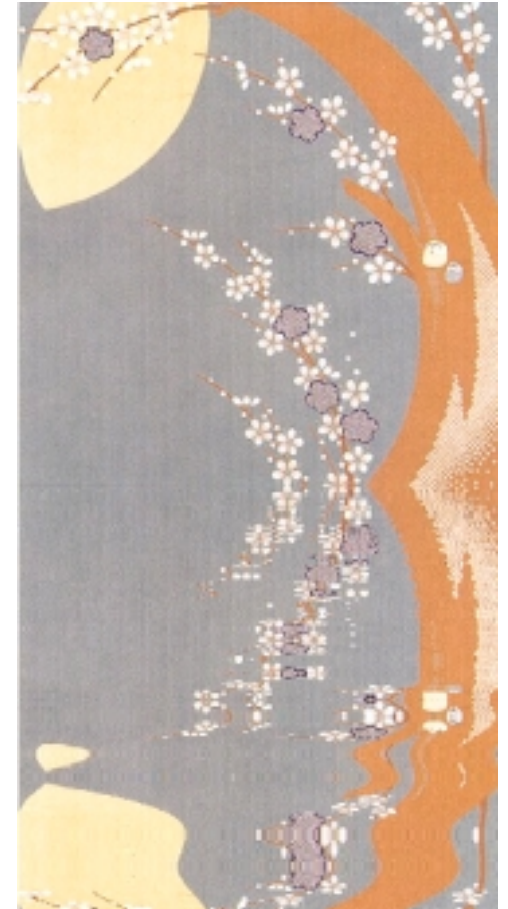
more pixel movement effects



ripple



image transitions



reflection in ripples

Noise

- Image processing is also useful for noise reduction
- Common types of noise:
 - » **Salt and pepper noise:** contains random occurrences of black and white pixels
 - » **Impulse noise:** contains random occurrences of white pixels
 - » **Gaussian noise:** variations in intensity drawn from a Gaussian normal distribution



Original



Salt and pepper noise



Impulse noise



Gaussian noise

Ideal noise reduction



image 1



image 2



average

image 1



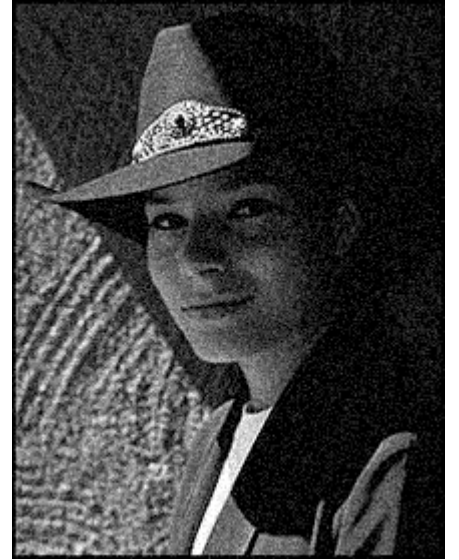
image 2



image 3



image 4



average

Convolution

- One of the most common methods for filtering an image is called **convolution**.
- In 1D, convolution is defined as:

$$\begin{aligned}g(x) &= f(x) * h(x) \\ &= \int_{-\infty}^{\infty} f(x')h(x - x')dx' \\ &= \int_{-\infty}^{\infty} f(x')\tilde{h}(x' - x)dx' \quad \text{where } \tilde{h}(x) = h(-x)\end{aligned}$$

$g(x)$ is “f convolved with h” :

evaluate the integral of $f \cdot h$ with h flipped around the y -axis and centered at x

Discrete convolution

- For a digital signal, we define **discrete convolution** as:

$$\begin{aligned}g[i] &= f[i] * h[i] \\ &= \sum_j f[j] h[i - j] \\ &= \sum_j f[j] \tilde{h}[j - i] \quad \text{where } \tilde{h}[i] = h[-i]\end{aligned}$$

$g[i]$ is “ f convolved with h ” :

evaluate the sum of $f \cdot h$ with h flipped around the y -axis and centered at i

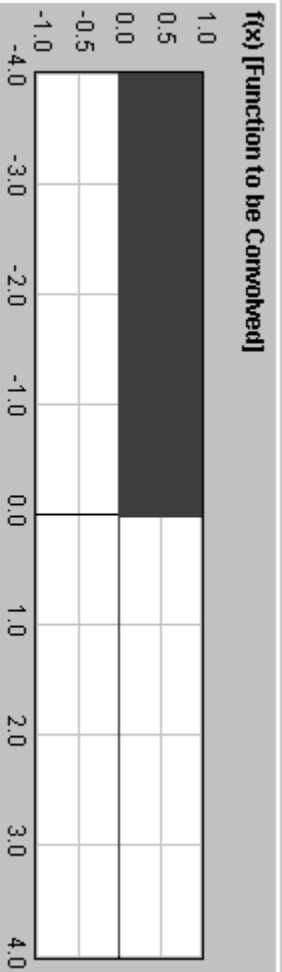
Convolution in 2D

In two dimensions, convolution becomes:

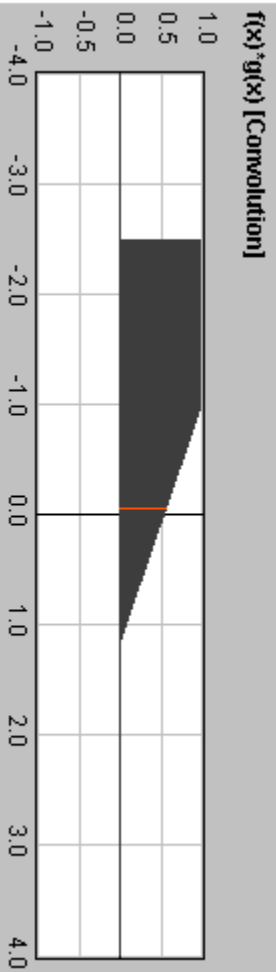
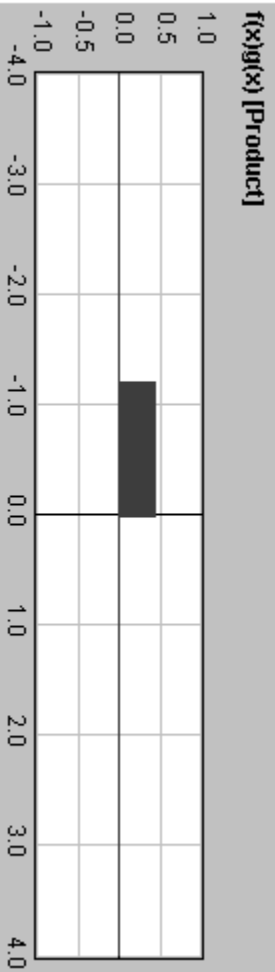
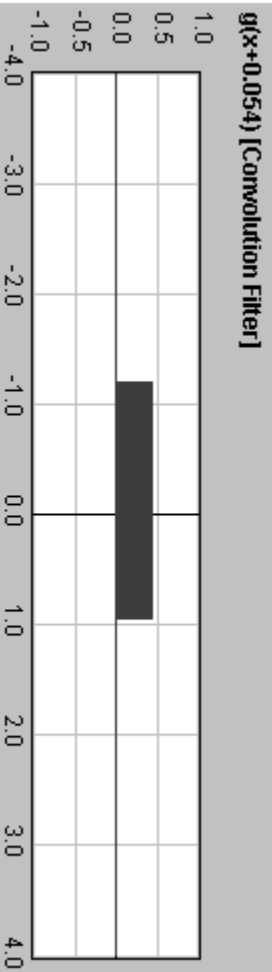
$$\begin{aligned}g(x, y) &= f(x, y) * h(x, y) \\ &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x', y') h(x - x', y - y') dx' dy' \\ &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x', y') \tilde{h}(x' - x, y' - y) dx' dy' \quad \text{where } \tilde{h}(x, y) = h(-x, -y)\end{aligned}$$

Similarly, discrete convolution in 2D becomes:

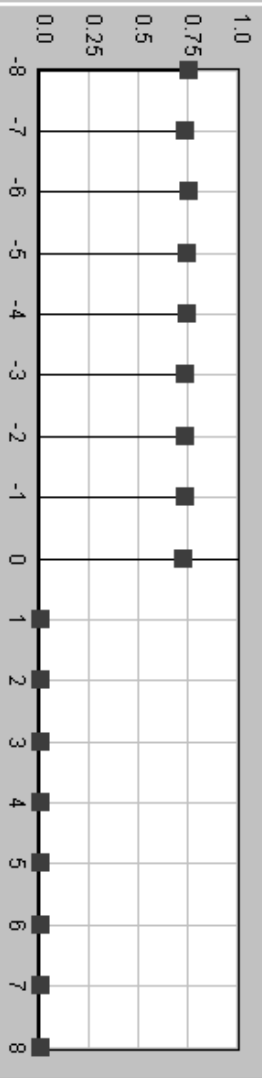
$$\begin{aligned}g[i, j] &= f[i, j] * h[i, j] \\ &= \sum_k \sum_l f[k, l] h[i - k, j - l] \\ &= \sum_k \sum_l f[k, l] \tilde{h}[k - i, l - j] \quad \text{where } \tilde{h}[i, j] = h[-i, -j]\end{aligned}$$



Normalize Clear

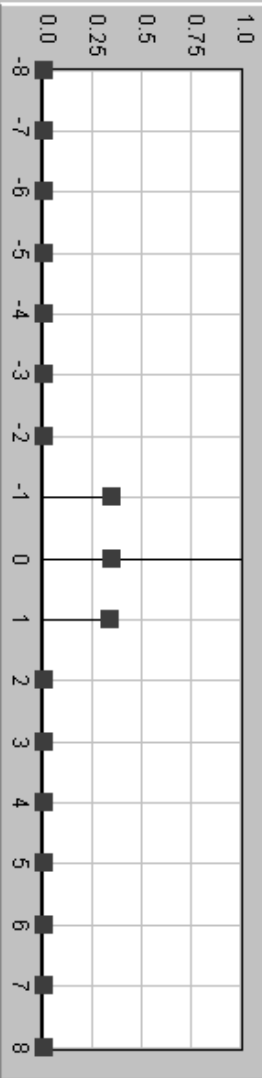


f(x) [Function for the Convolver]

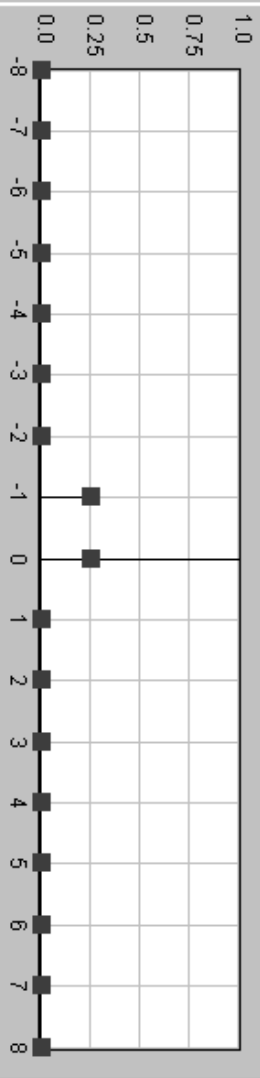


Normalize Clear

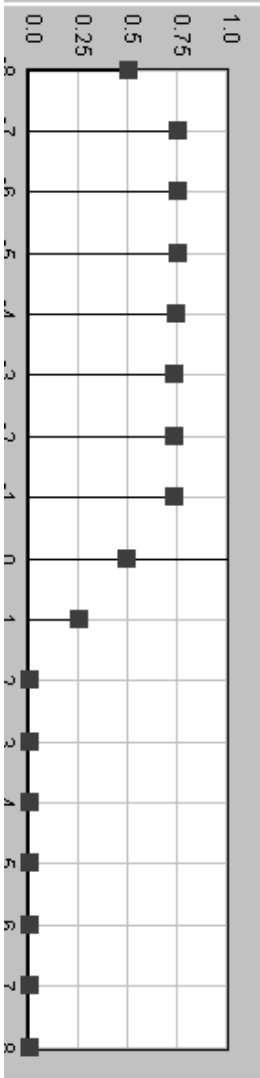
g(x) [Convolution Filter]



f(x)g(x) [Product]

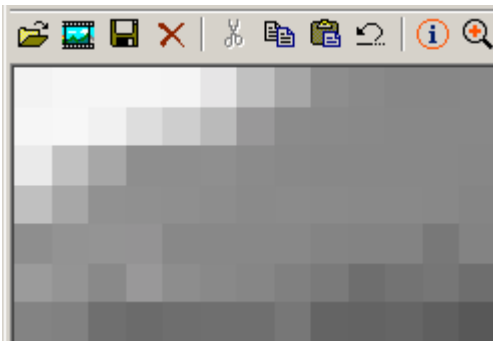


f(x)*g(x) [Convolution]



Convolution representation

- Since f , g , and \tilde{h} are defined over finite regions, we can write them out in two-dimensional arrays:



f

| | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 242 | 245 | 245 | 245 | 244 | 228 | 191 | 165 |
| 245 | 246 | 240 | 220 | 204 | 184 | 151 | 138 |
| 232 | 192 | 165 | 141 | 131 | 142 | 138 | 136 |
| 190 | 165 | 144 | 143 | 142 | 140 | 137 | 138 |
| 141 | 145 | 148 | 146 | 135 | 135 | 135 | 135 |

| | | |
|-----|-----|-----|
| 0.2 | 0 | 0.2 |
| 0 | 0.2 | 0 |
| 0.2 | 0 | 0.2 |

\tilde{h}

g

| | | | | | | | |
|--|--|--|--|--|--|--|--|
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |

Note: *This is not matrix multiplication!*

Mean filters

- How can we represent our noise-reducing averaging filter as a convolution diagram?

Effect of mean filters

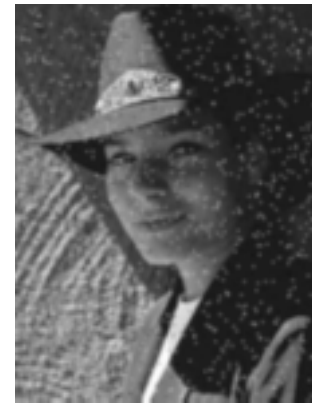
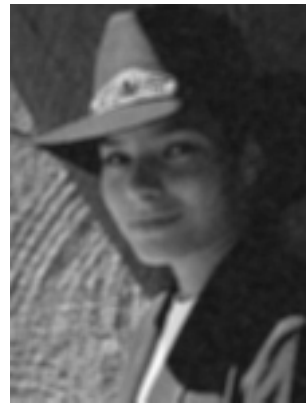
Gaussian noise

Salt and pepper noise

3x3



5x5



7x7



Gaussian filters

- Gaussian filters weigh pixels based on their distance from the center of the convolution filter. In particular:

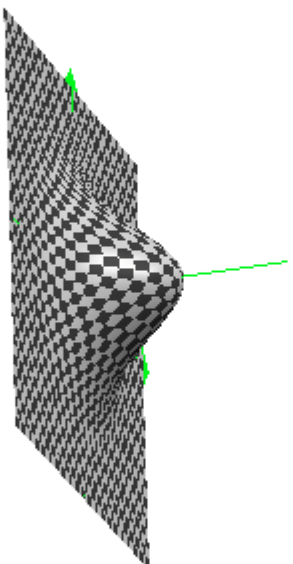
$$h [i , j] = \frac{e^{- (i^2 + j^2) / (2 \sigma^2)}}{C}$$

- This does a decent job of blurring noise while preserving features of the image.
- What parameter controls the width of the Gaussian?
- What happens to the image as the Gaussian filter kernel gets wider?
- What is the constant C ? What should we set it to?

$$z = e^{-\frac{(x^2 + y^2)}{2\pi^2}}$$

Illustration of the formula that gives the values for a Gaussian filter:

x : -2 ... 2
y : -2 ... 2
z : -2 ... 2



0.5

Effect of Gaussian filters

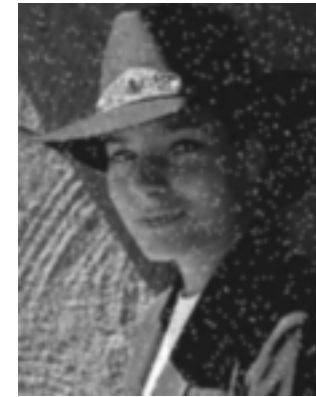
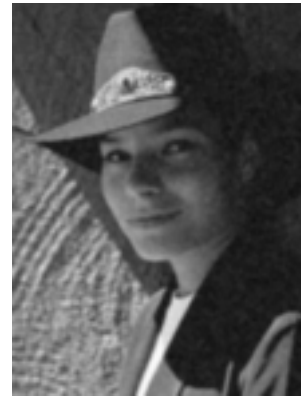
Gaussian noise

Salt and pepper noise

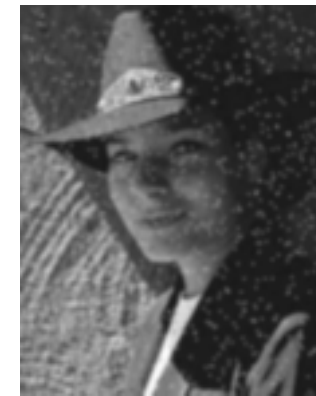
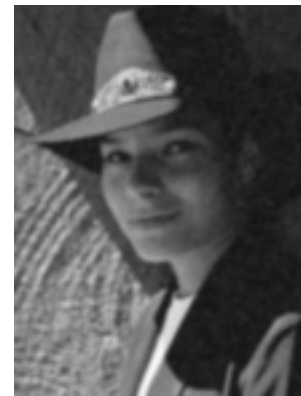
3x3



5x5



7x7



Median filters

- A **Median Filter** operates over an $m \times m$ region by selecting the median intensity in the region.
- What advantage does a median filter have over a mean filter?
- Is a median filter a kind of convolution?

Effect of median filters

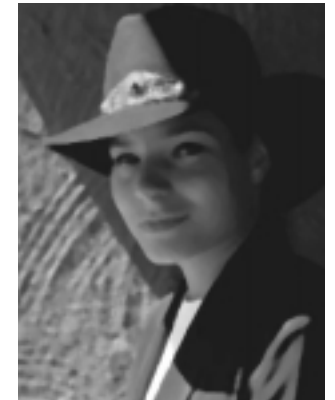
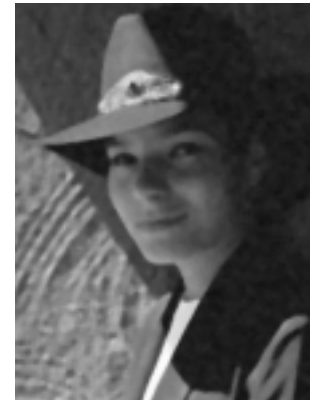
Gaussian
noise

Salt and pepper
noise

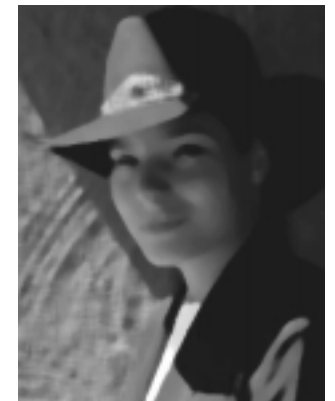
3x3



5x5



7x7



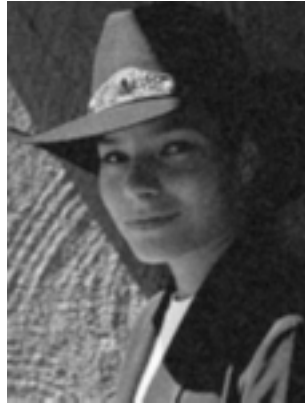
Comparison: Gaussian noise

Mean

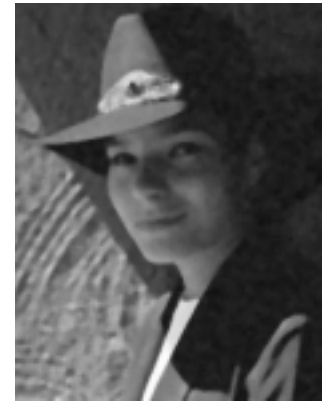
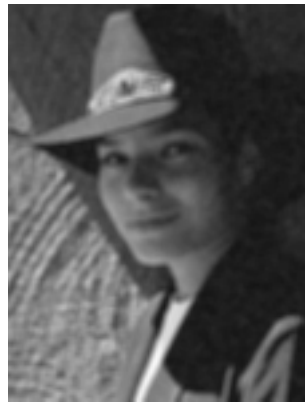
Gaussian

Median

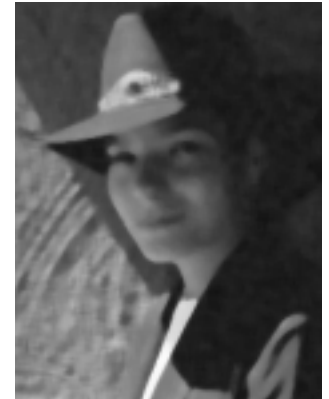
3x3



5x5



7x7



Comparison: salt and pepper noise

3x3

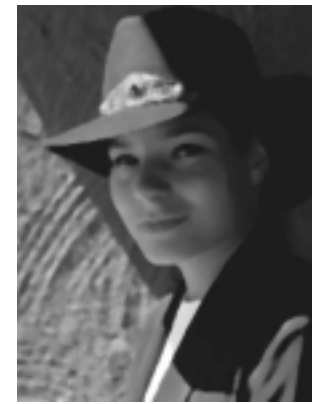
Mean

Gaussian

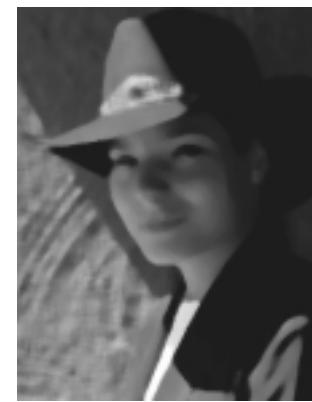
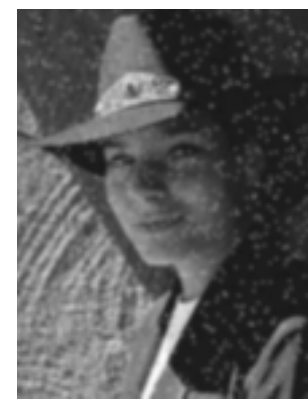
Median



5x5



7x7

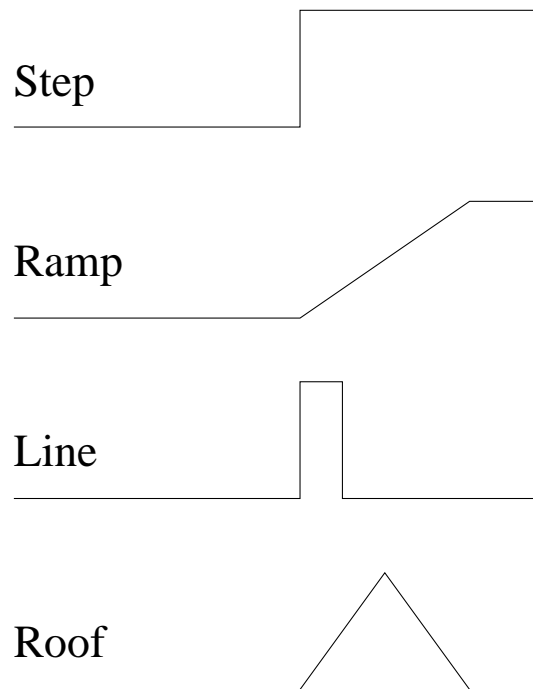


Edge detection

- One of the most important uses of image processing is **edge detection**:
 - » Really easy for humans
 - » Really difficult for computers

 - » Fundamental in computer vision
 - » Important in many graphics applications

What is an edge?



- **Q:** How might you detect an edge in 1D?

Gradients

- The **gradient** is the 2D equivalent of the derivative:

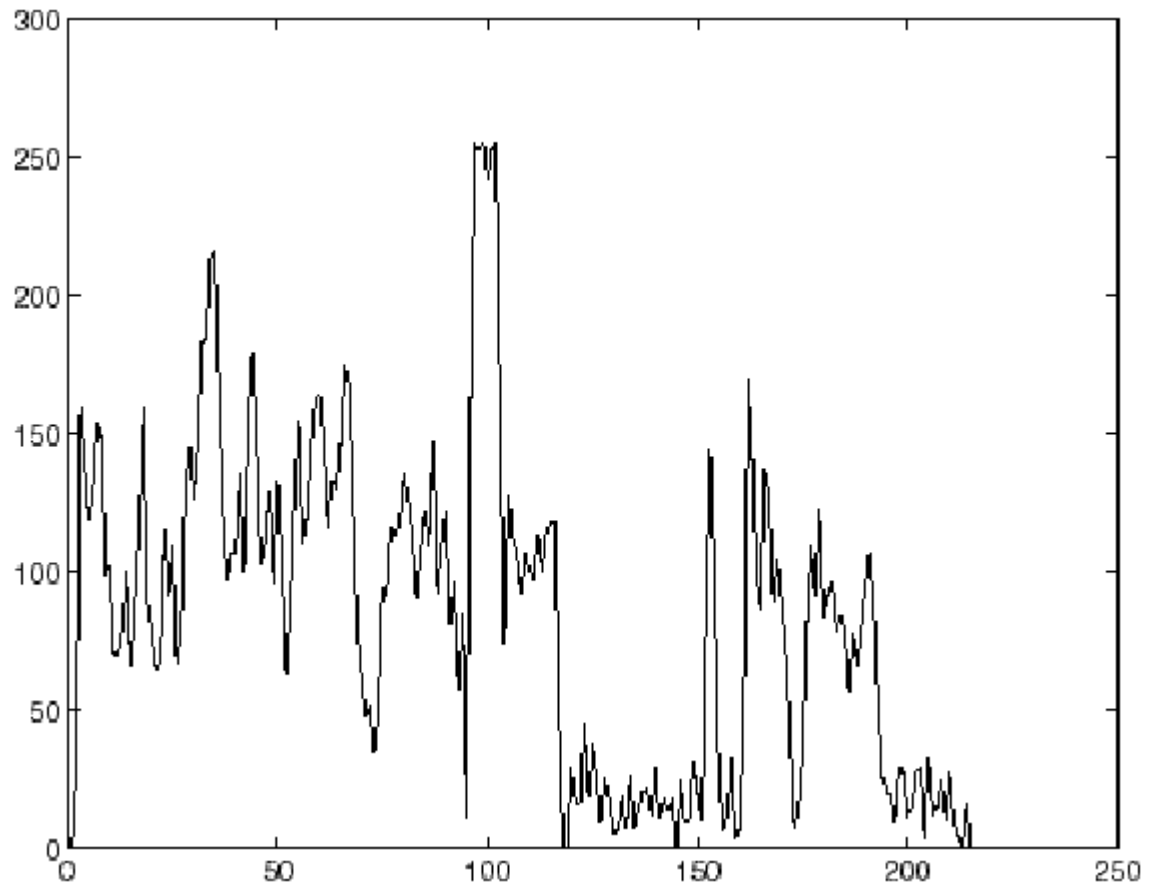
$$\nabla f(x, y) = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right)$$

- Properties of the gradient
 - » It's a vector
 - » Points in the direction of maximum increase of f
 - » Magnitude is rate of increase
- How can we approximate the gradient in a discrete image?

Less than ideal edges



pixel row plotted



Steps in edge detection

- Edge detection algorithms typically proceed in three or four steps:
 - » **Filtering**: cut down on noise
 - » **Enhancement**: amplify the difference between edges and non-edges
 - » **Detection**: use a threshold operation
 - » **Localization** (optional): estimate geometry of edges beyond pixels

Edge enhancement

- A popular gradient magnitude computation is the **Sobel operator**:

$$S_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad S_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

- We can then compute the magnitude of the vector (s_x, s_y) .

Results of Sobel edge detection



Original



Smoothed



Magnitude



Threshold = 64



Threshold = 128



$S_x + 128$

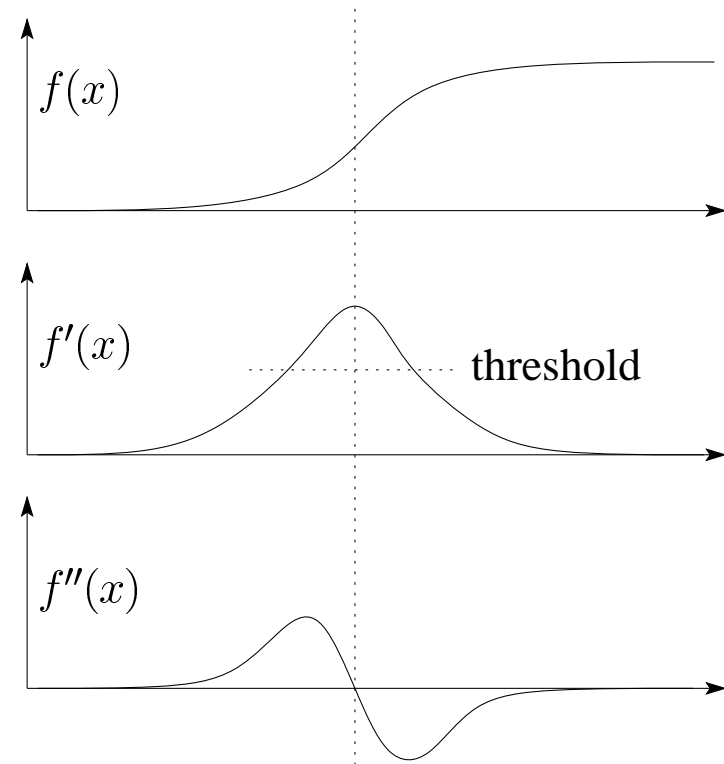


$S_y + 128$

Second derivative operators

- The Sobel operator can produce thick edges. Ideally, we're looking for infinitely thin boundaries.

An alternative approach is to look for local extrema in the first derivative: places where the change in the gradient is highest.



Q: A peak in the first derivative corresponds to what in the second derivative?

Localization with the Laplacian

- An equivalent measure of the second derivative in 2D is the **Laplacian**:

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

- Using discrete difference equations, the Laplacian filter can be shown to be:

$$\Delta^2 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Zero crossings of this filter correspond to positions of maximum gradient. These zero crossings can be used to localize edges.

Localization with the Laplacian



Original



Smoothed



Laplacian (+128)

Sharpening with the Laplacian



Original



Laplacian (+128)



Original + Laplacian



Original - Laplacian

Why does the sign make a difference?

How can you write a filter that makes the bottom image?



100%



Filter matrix

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

Matrix modifiers

Divisor:

1

Compute

Bias:

0

Apply to

Color Channels:

Grey values

Red

Green

Blue

Summary

- What you should take away from this lecture:
 - » The meanings of all the boldfaced terms.
 - » A richer understanding of the terms “image” and “image processing”
 - » How noise reduction is done
 - » How convolution filtering works
 - » The effect of mean, Gaussian, and median filters
 - » What an image gradient is and how it can be computed
 - » How edge detection is done
 - » What the Laplacian image is and how it is used in either edge detection or image sharpening