Lecture 4: Perception Basics

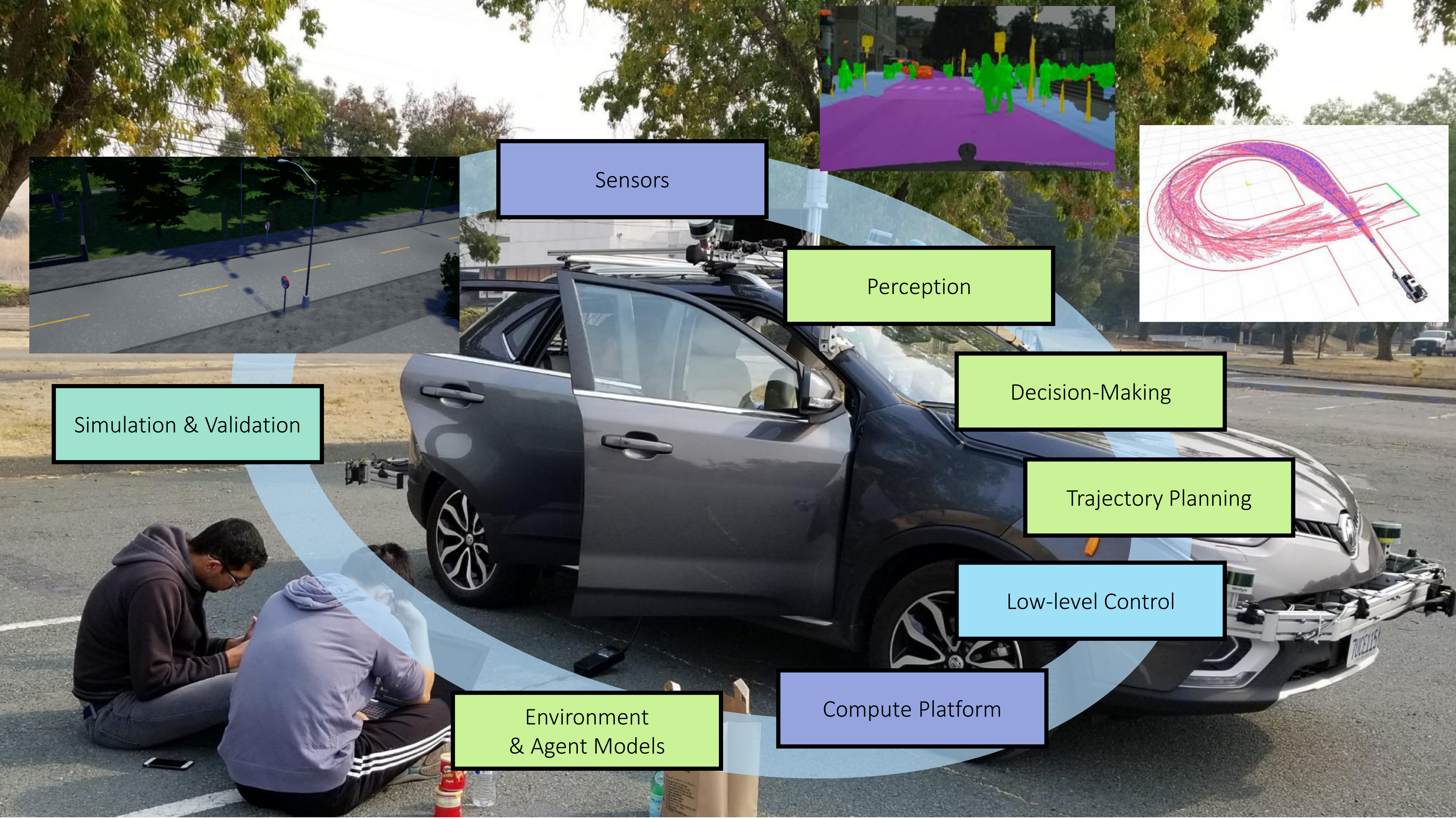
Professor Katie Driggs-Campbell

February 9, 2021

ECE484: Principles of Safe Autonomy

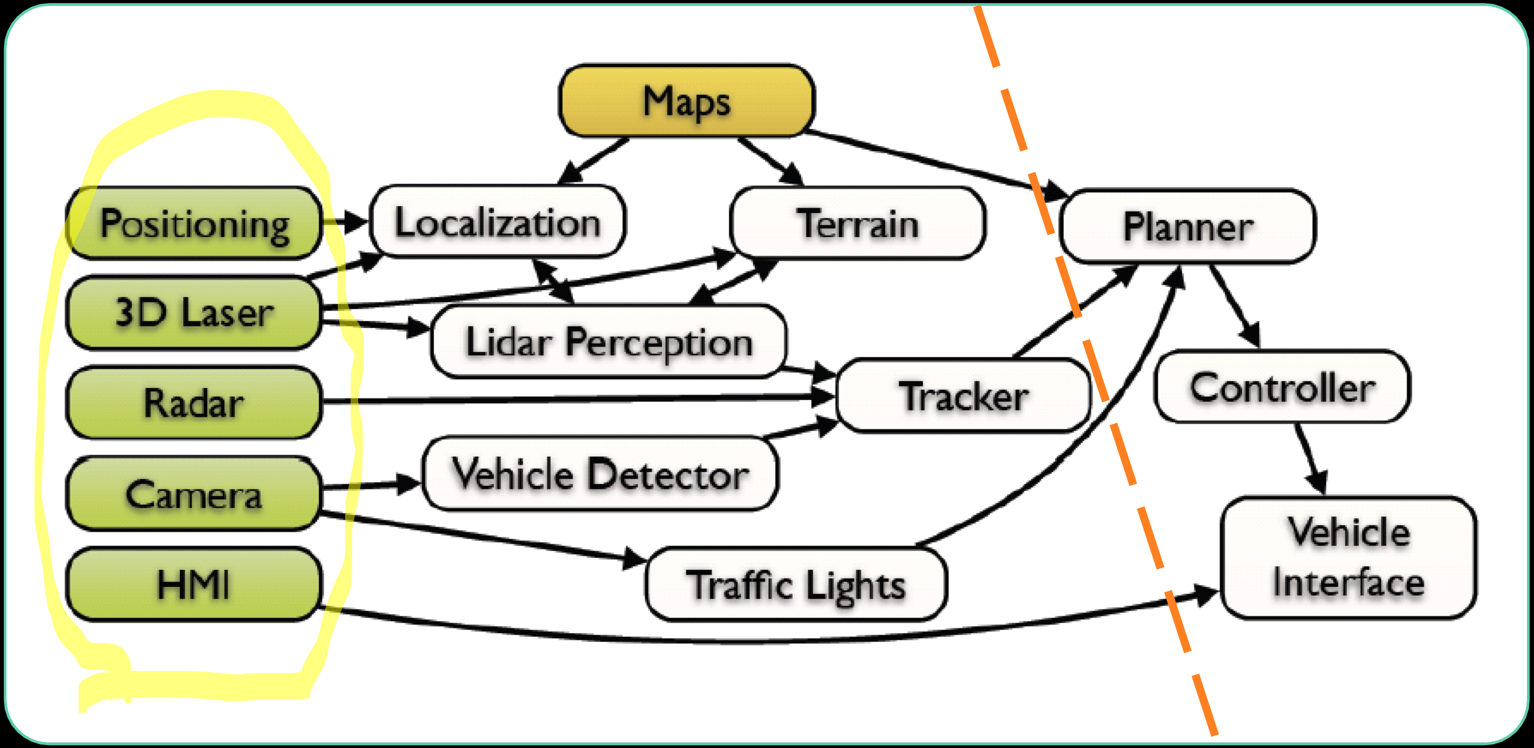
Administrivia

- Project pitch in class on 2/23
  - Five-minute presentation + questions per team
  - Signup will be posted soon (first come, first serve)
  - Asynchronous students must send course staff a video the day before
- Milestone due date 3/19
  - ~1pg outlining a more concrete project idea, including a timeline of what you will accomplish in the next month and a half
  - Outline will be posted soon
- For hardware projects, safety training will likely be in late February
- Oral exams will likely be ~3/29



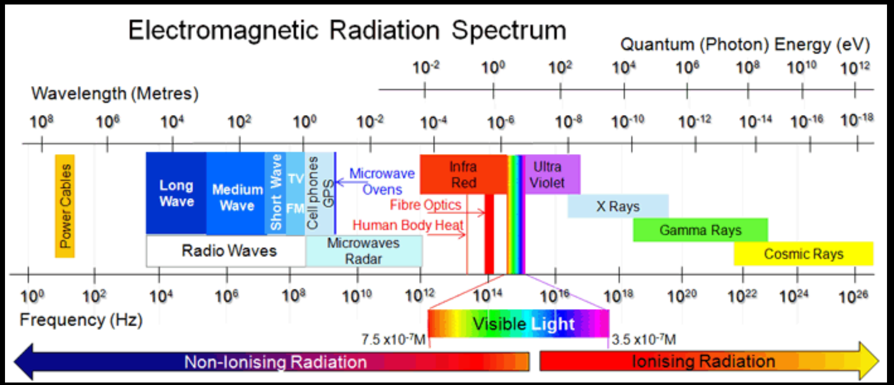
Sensors
Perception
Decision-Making
Trajectory Planning
Low-level Control
Compute Platform
Environment
& Agent Models
Simulation & Validation

AV Perception Pipeline

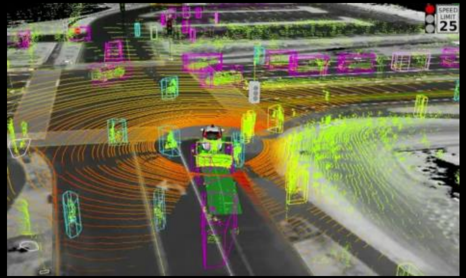


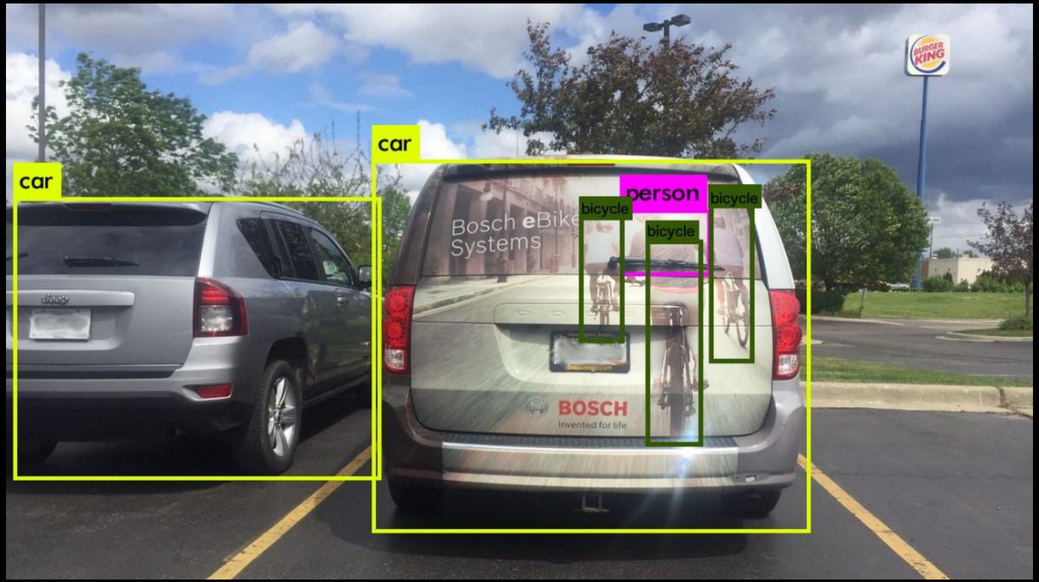
This architecture from a slide from M. James of Toyota Research Institute, North America

The Challenge of Perception



Sensor Goal: Process electromagnetic radiation from the environment to construct a *model* of the world, so that the constructed model is close to the real world and that the output is *actionable*





Today's Plan

- Basic image processing with filtering
- Edge detection

Motivation: Filtering for image de-noising

Modify the pixels in an image based on some function of a local neighborhood of the pixels.

- Scaling: img’ = k*img
- Shifting right by s: img’[k] = img[k-s]
  - img’[0]…img’[s-1] is undefined
- Linear filtering: replace each pixel by a linear combination of neighbors

10	5	3
4	5	1
1	1	7

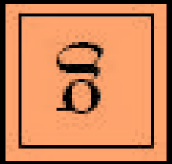
some function →

	7	

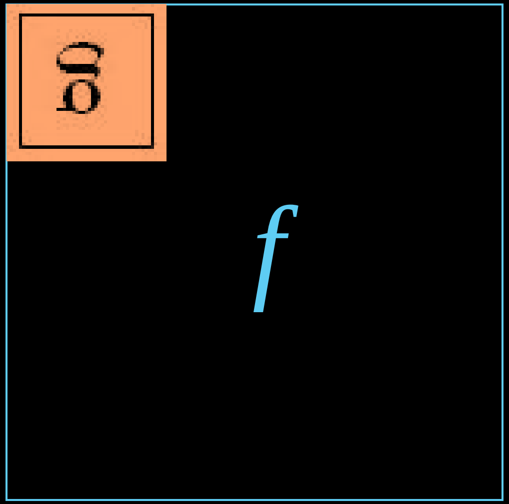
Defining convolution

Let f be the image and g be the kernel. The output of convolving f with g is denoted $f*g$.

$$(f*g)[m,n]=\sum_{k,l} f[m-k,n-l]\,g[k,l]$$



kernel is "flipped" by convention

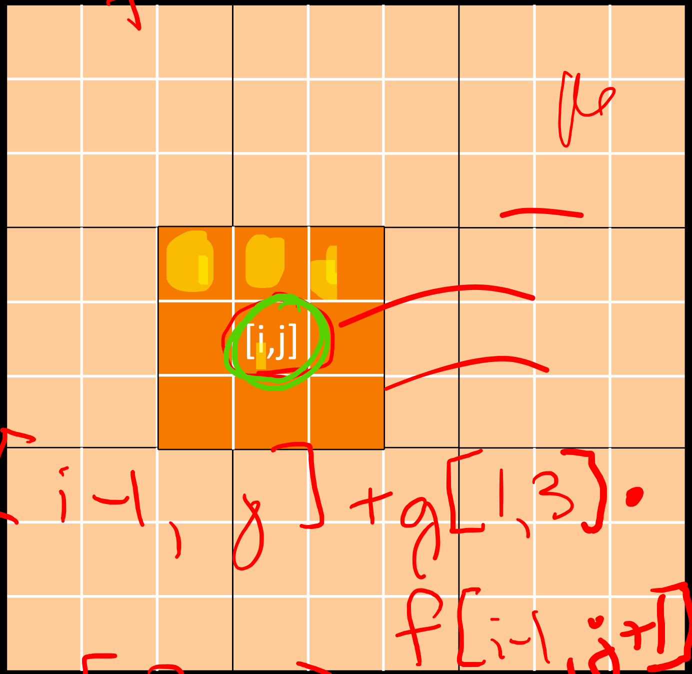


Source: F. Durand

Image Convolution

$$(f * g)[m,n] = \sum_{k,l} f[m-k, n-l]\, g[k,l]$$

1,1	1,2	1,3
2,1	2,2	2,3
3,1	3,2	3,3



$h = g * f$

$h[i,j] = \cancel{g[1,1] f[i-1, j-1]} + g[1,2] f[i-1, j] + g[1,3] \cdot f[i-1, j+1]$

$+ \cancel{g[2,1] f[i, j-1]} + g[2,2] f[i,j] + g[2,3] f[i, j+1]$

$+ \cancel{g[3,1] f[i+1, j-1]} + g[3,2] f[i+1, j] + g[3,3] f[i+1,$

ex] weighted avg or box filter

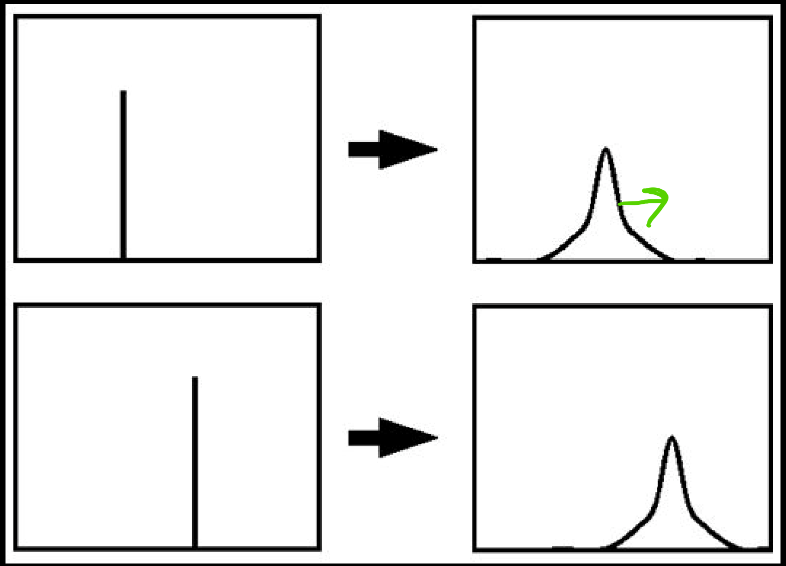
$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Key properties

- **Shift invariance:** same behavior regardless of pixel location:
  filter(shift(f)) = shift(filter(f))

- **Linearity:**
  filter($f_1 + f_2$) = filter(f_1) + filter(f_2)

→ **Theoretical result:** any linear shift-invariant operator can be represented as a convolution



Properties in more detail

- Commutative: $a * b = b * a$
  - Conceptually no difference between filter and signal
- Associative: $a * (b * c) = (a * b) * c$
  - Often apply several filters one after another: $(((a * b_1) * b_2) * b_3)$
  - This is equivalent to applying one filter: $\mathrm{a} * (b_1 * b_2 * b_3)$
- Distributes over addition: $a * (b + c) = (a * b) + (a * c)$
- Scalars factor out: $ka * b = a * kb = k\,(a * b)$
- Identity: unit impulse $e = [\ldots, 0, 0, 1, 0, 0, \ldots]$, $a * e = a$

Practice with linear filters

Original

0	0	0
0	1	0
0	0	0

Filtered
(no change)

Practice with linear filters

Original

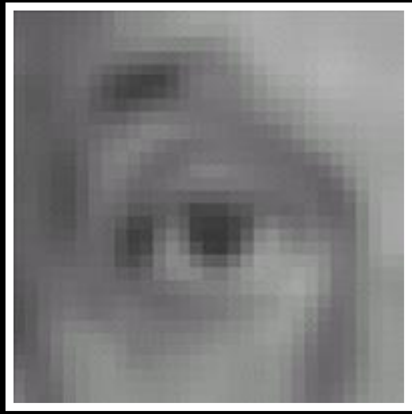
0	0	0
0	0	1
0	0	0

Shifted *left*
By 1 pixel

Practice with linear filters

Original

1	1	1
1	1	1
1	1	1



Blur (with a box filter)

Practice with linear filters

Original

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

(Note that filters should sum to 1)

Practice with linear filters

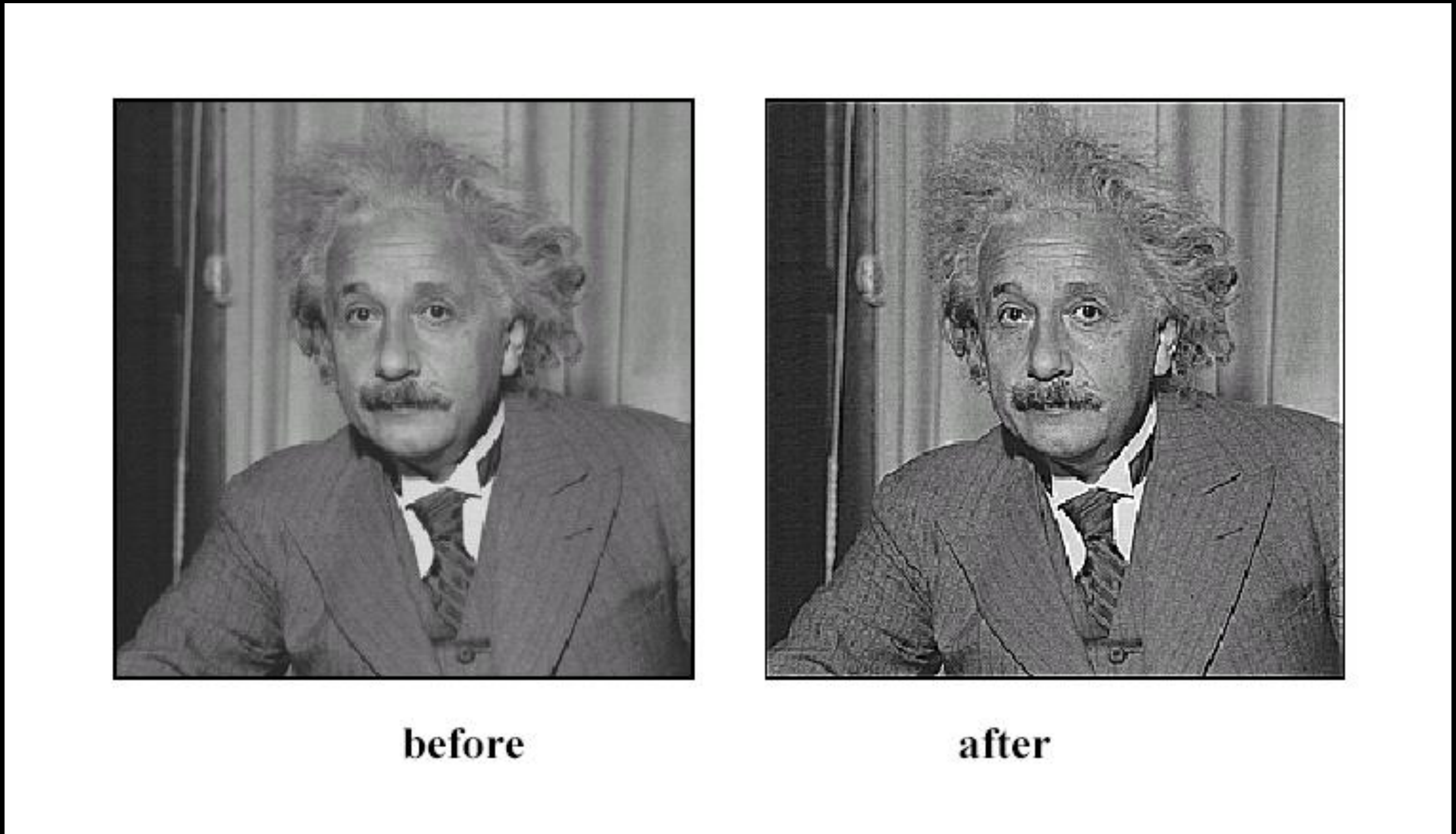
Original

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

Sharpening filter: Accentuates differences with local average

Source: D. Lowe

Sharpening Filter



Sharpening

What does blurring take away?

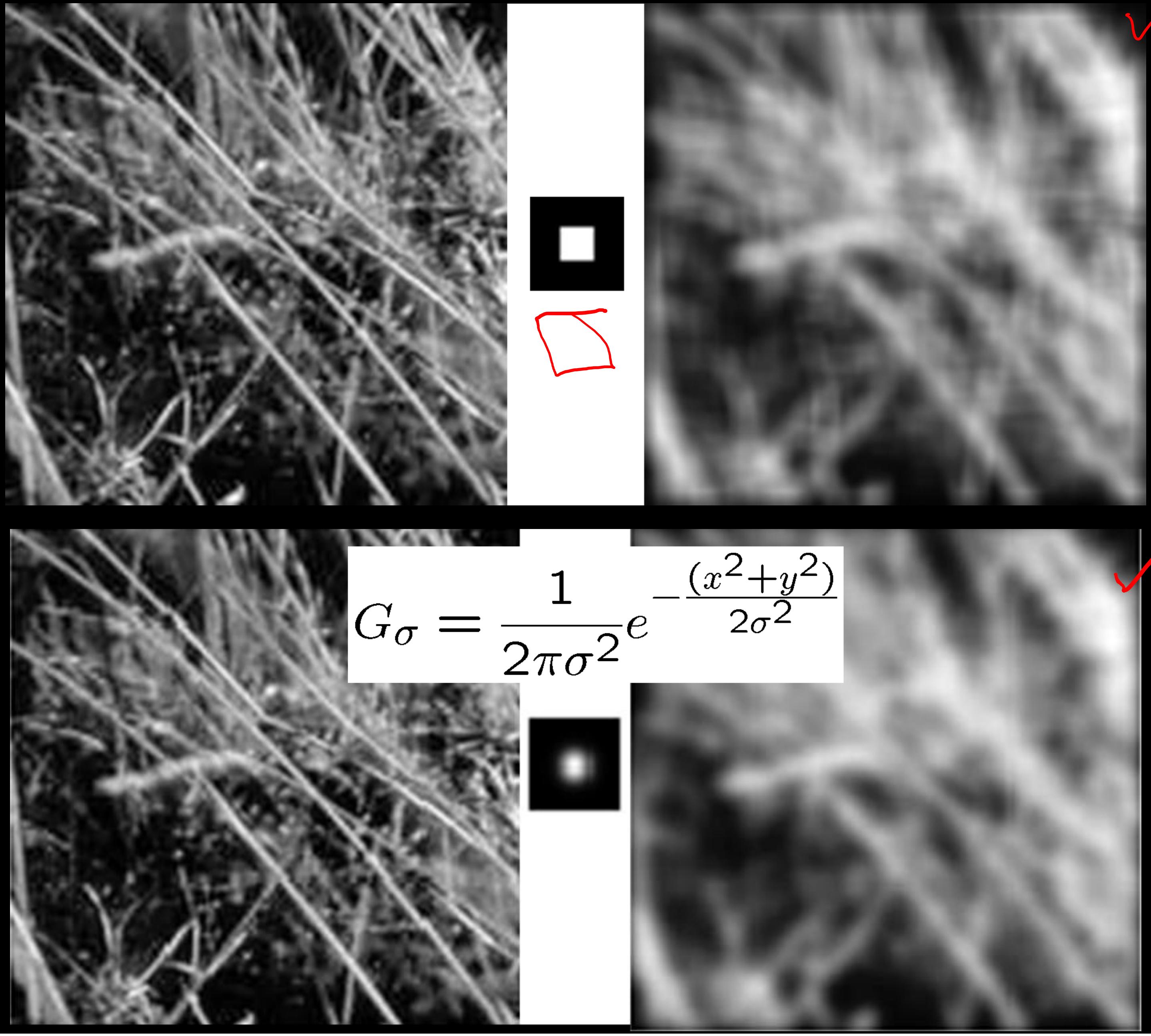
Let's add it back:

Source: D. Lowe

Soft Smoothing

Box Filter

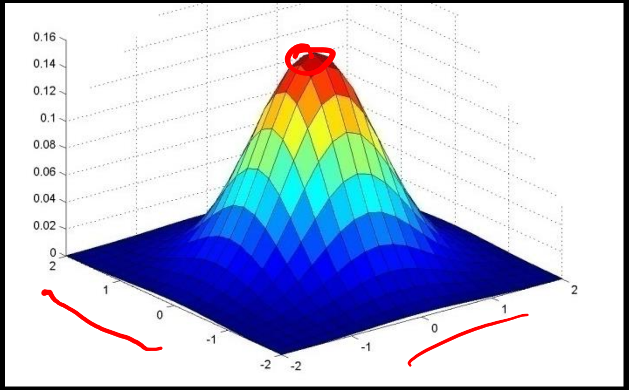
Gaussian Filter

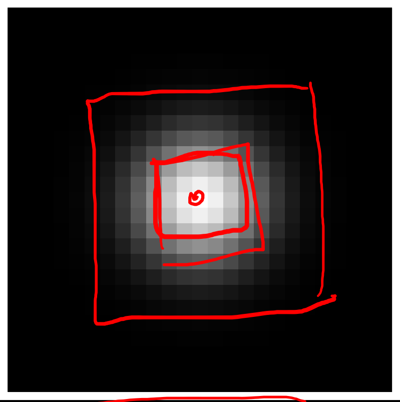


$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

Gaussian Kernel

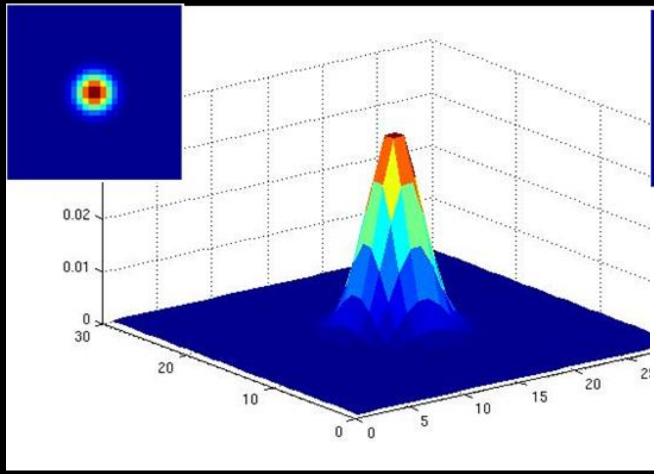
$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$



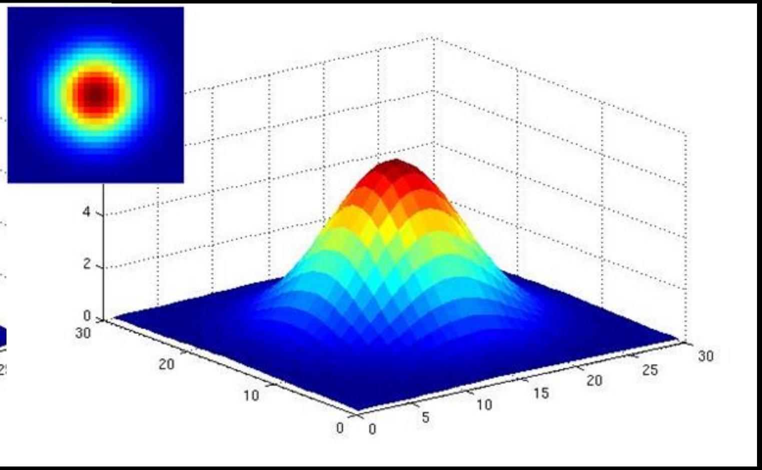


0.003	0.013	0.022	0.013	0.003
0.013	0.059	0.097	0.059	0.013
0.022	0.097	0.159	0.097	0.022
0.013	0.059	0.097	0.059	0.013
0.003	0.013	0.022	0.013	0.003

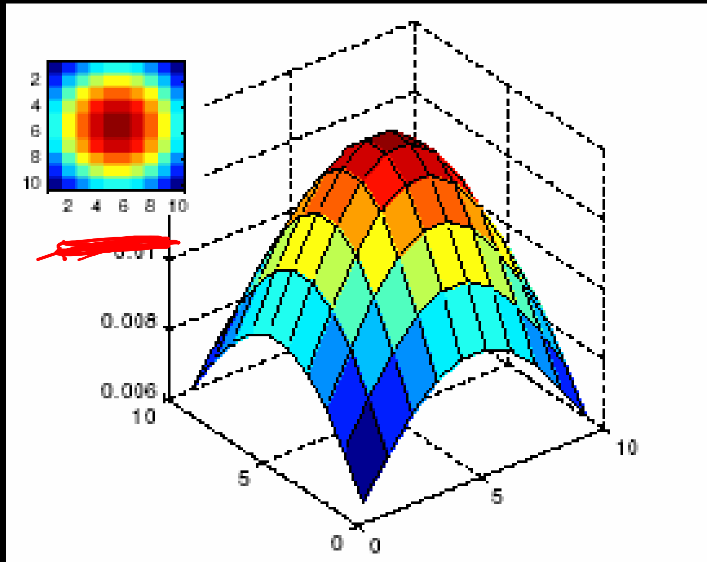
5 x 5, σ = 1



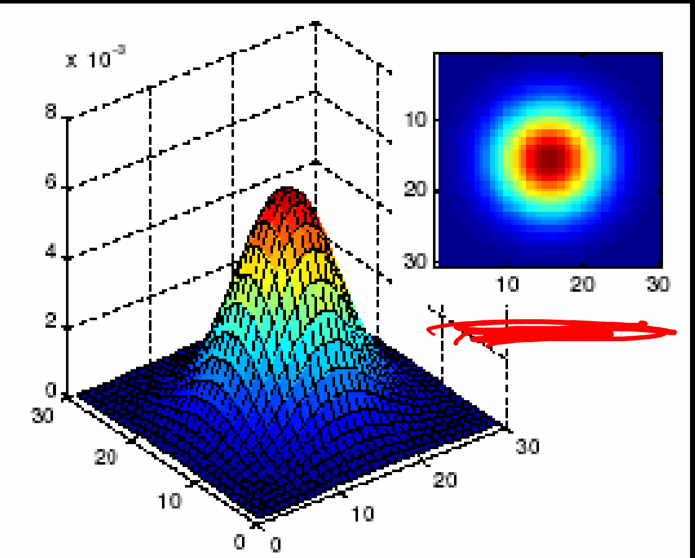
σ = 2 with 30 x 30 kernel



σ = 5 with 30 x 30 kernel



σ = 5 with 10x10 kernel



σ = 5 with 30x30 kernel

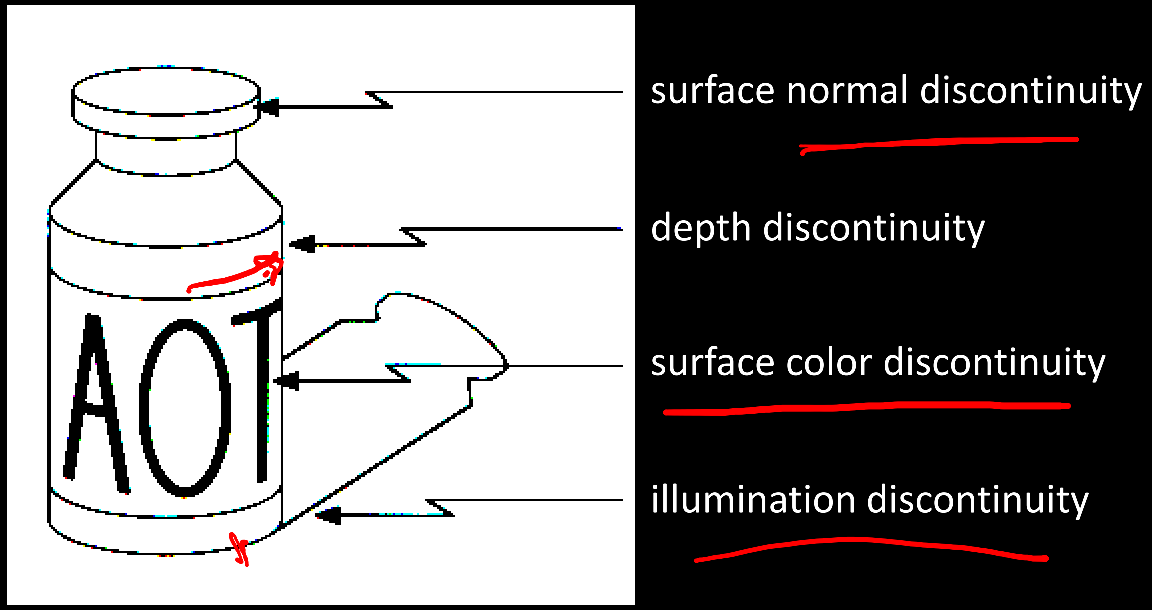
Edge detection

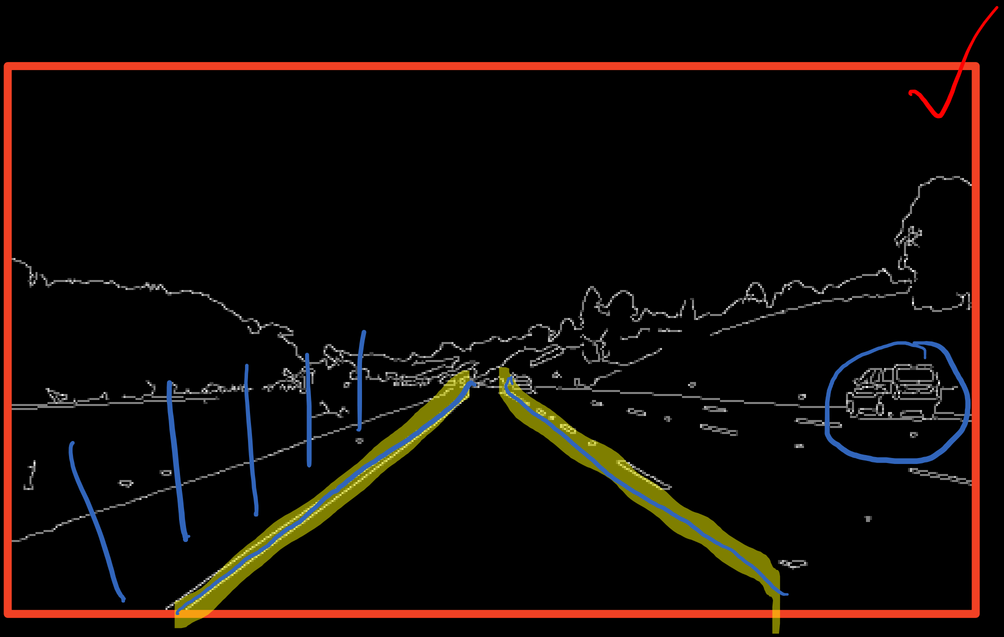
Winter in Kraków photographed by Marcin Ryczek

Edge detection

Goal: Identify sudden changes (discontinuities) in an image

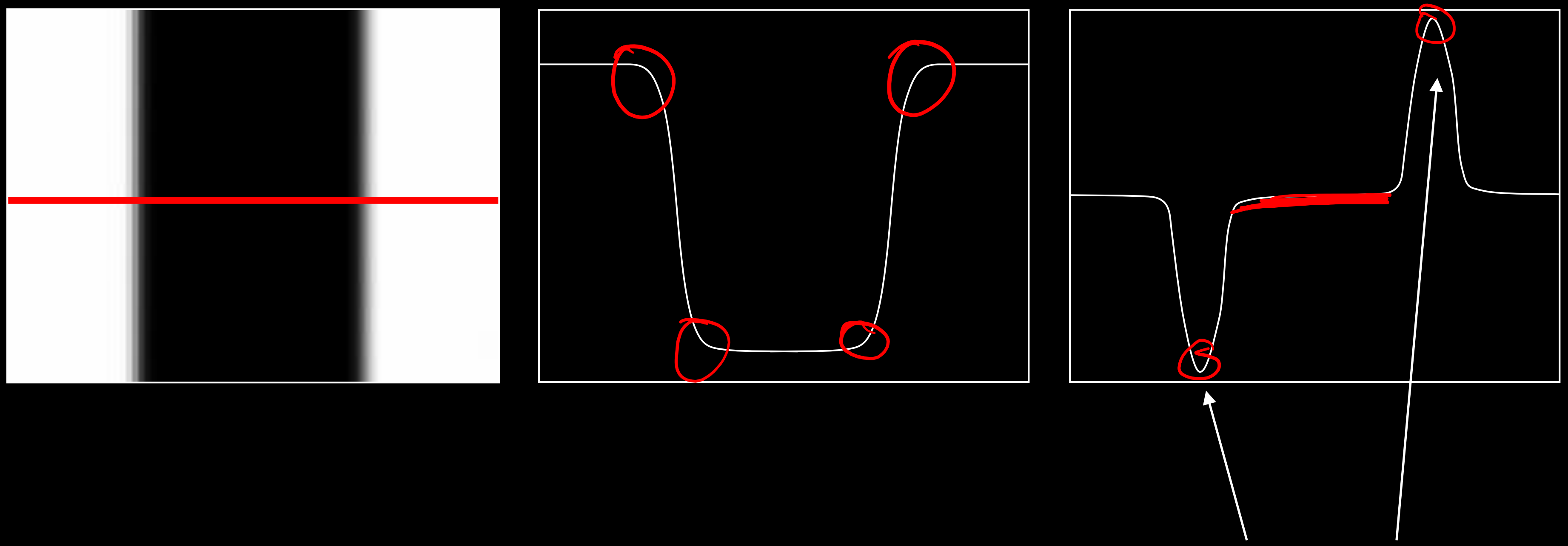
→Intuitively, edges carry most of the semantic and shape information from the image (e.g., lanes, traffic signs, cars)





Edge detection

An edge is a place of rapid change in the image intensity function



Partial Derivatives

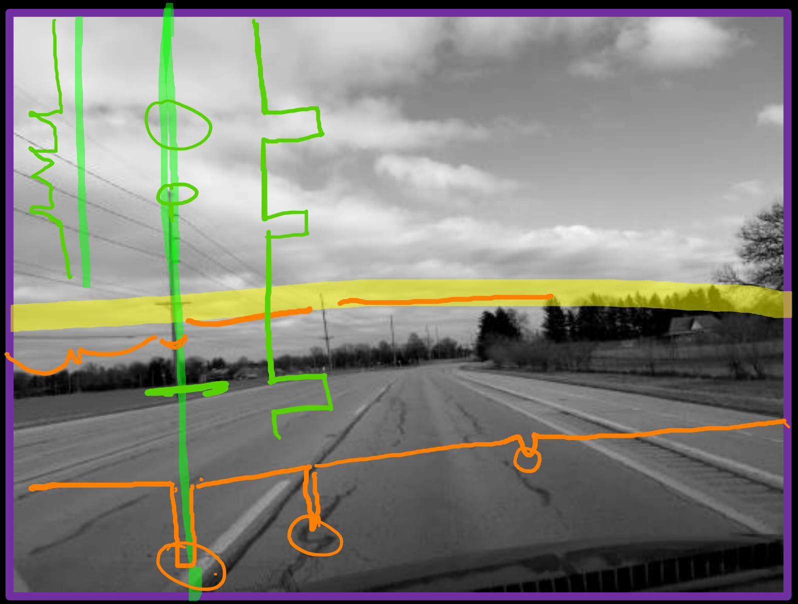
for $f(x,y)$, the pd w.r.t. x is:

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

Finite difference approx:

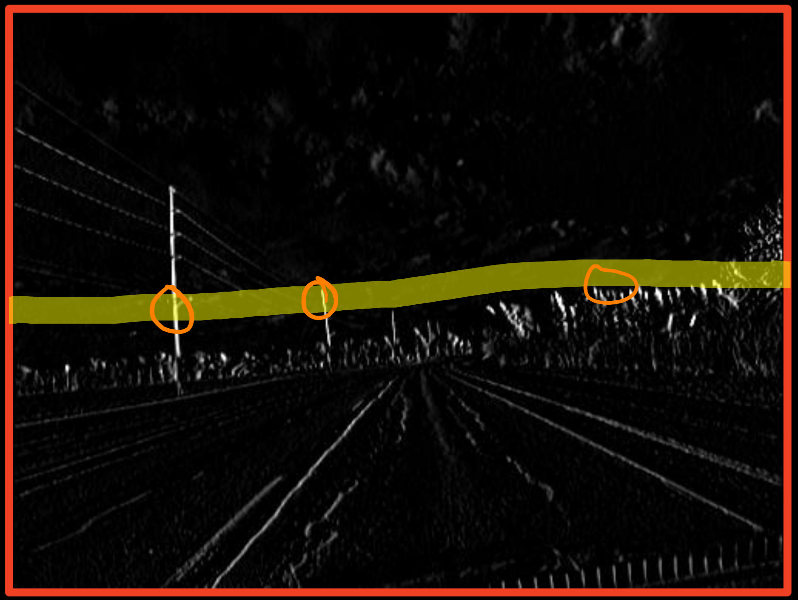
$$\frac{\partial f(x,y)}{\partial x} \approx \frac{f(x+1, y) - f(x,y)}{1} \checkmark$$

Partial derivatives of an image



$x \rightarrow$

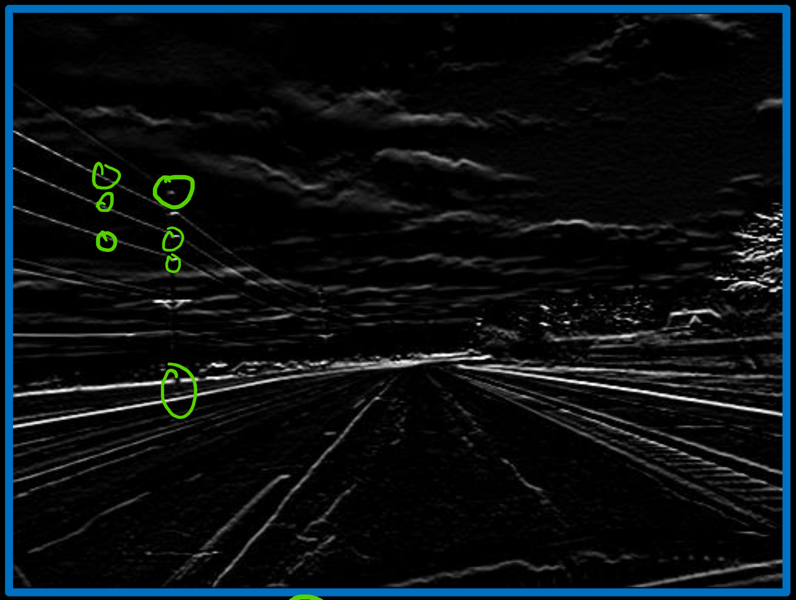
1



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

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

2



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

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

Other approximations for finite difference filters

Prewitt: $M_x = \begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -1 & 0 & 1 \\ \hline -1 & 0 & 1 \\ \hline\end{array}$; $M_y = \begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -1 & -1 \\ \hline\end{array}$

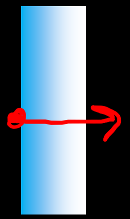
Sobel: $M_x = \begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline\end{array}$; $M_y = \begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -2 & -1 \\ \hline\end{array}$

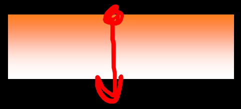
Roberts: $M_x = \begin{array}{|c|c|}\hline 0 & 1 \\ \hline -1 & 0 \\ \hline\end{array}$; $M_y = \begin{array}{|c|c|}\hline 1 & 0 \\ \hline 0 & -1 \\ \hline\end{array}$

Image Gradient

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

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



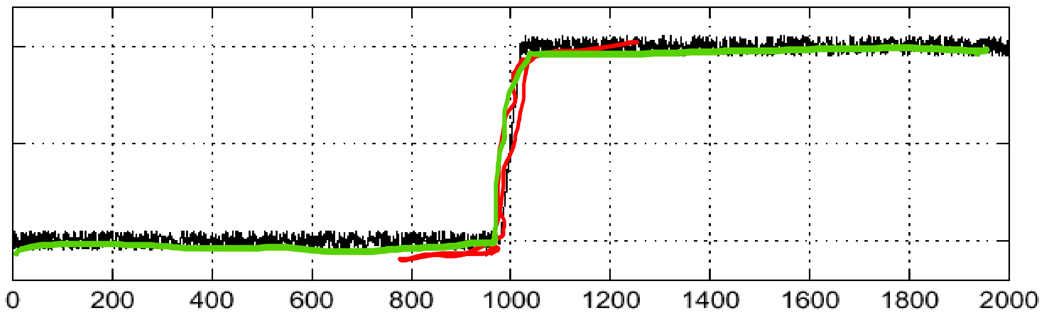


Gradient direction: $\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} \Big/ \frac{\partial f}{\partial x} \right)$

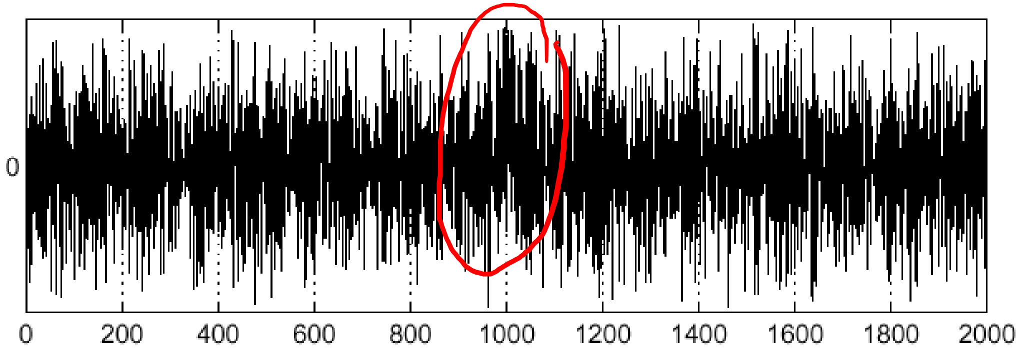
Edge Strength: $\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$

Impact of Noise

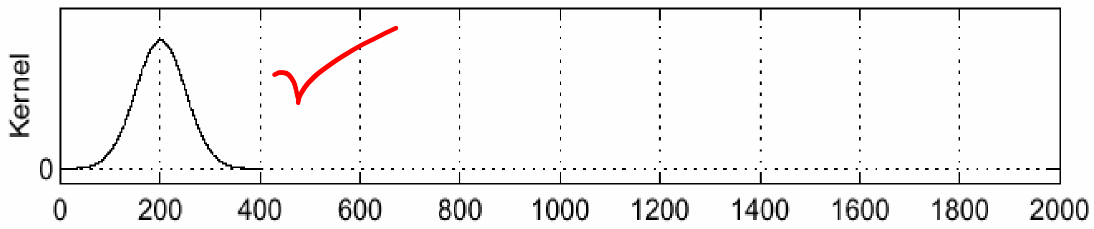
f



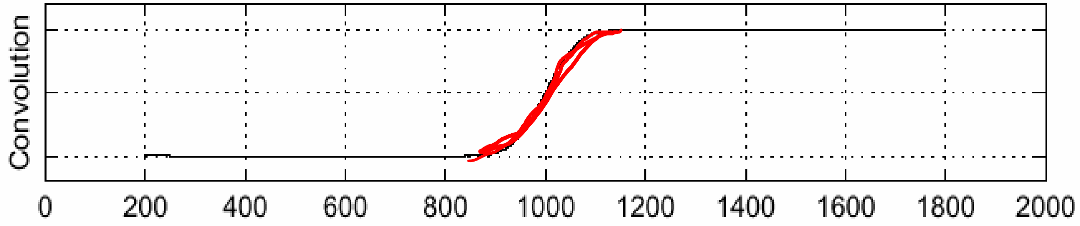
$\frac{d}{dx} f(x)$



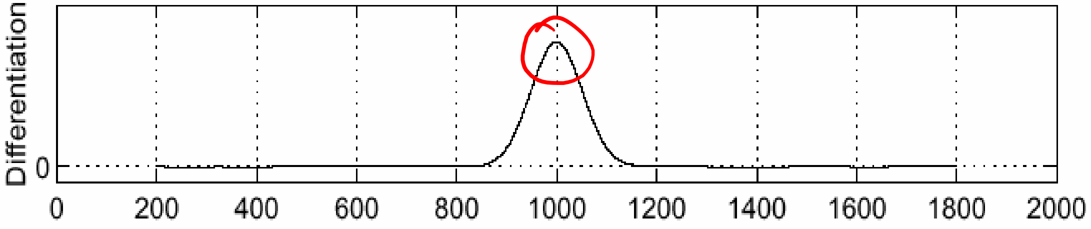
g



$f * g$

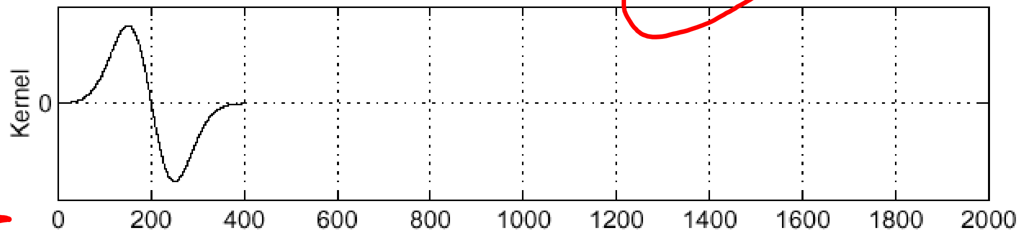


$\frac{d}{dx}(f * g)$

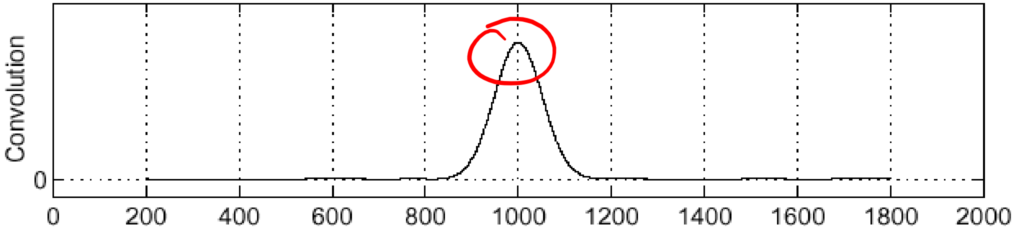


$$\frac{d}{dx}(f * g) = f * \frac{d}{dx} g$$

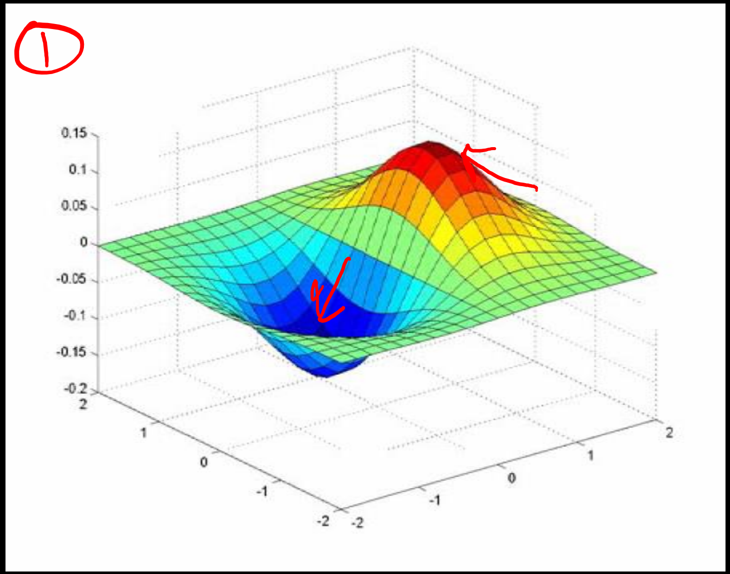
$\frac{d}{dx} g$

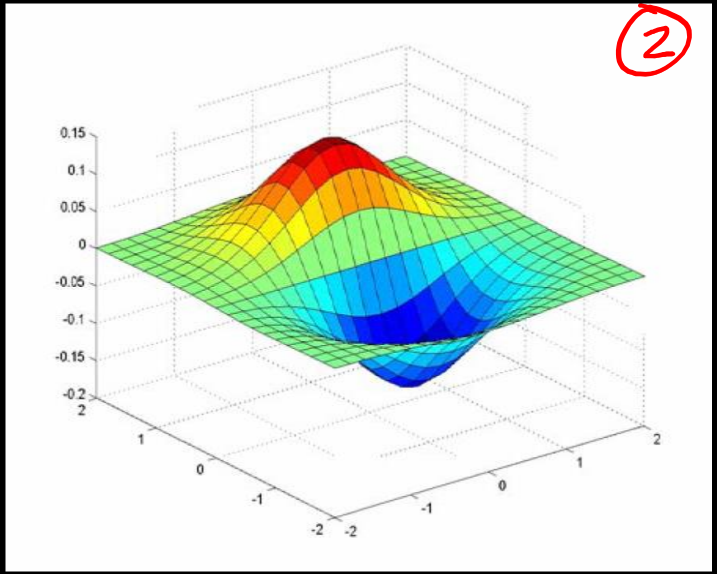


$f * \frac{d}{dx} g$

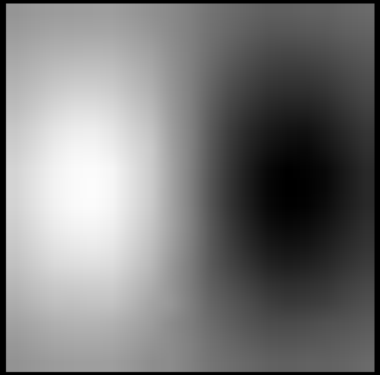


Derivative of Gaussian filters

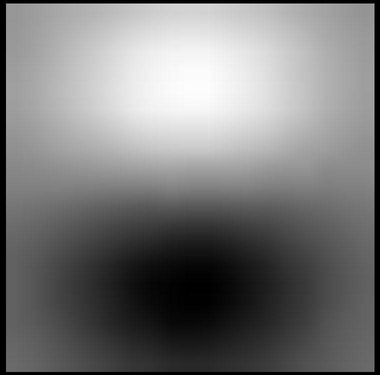




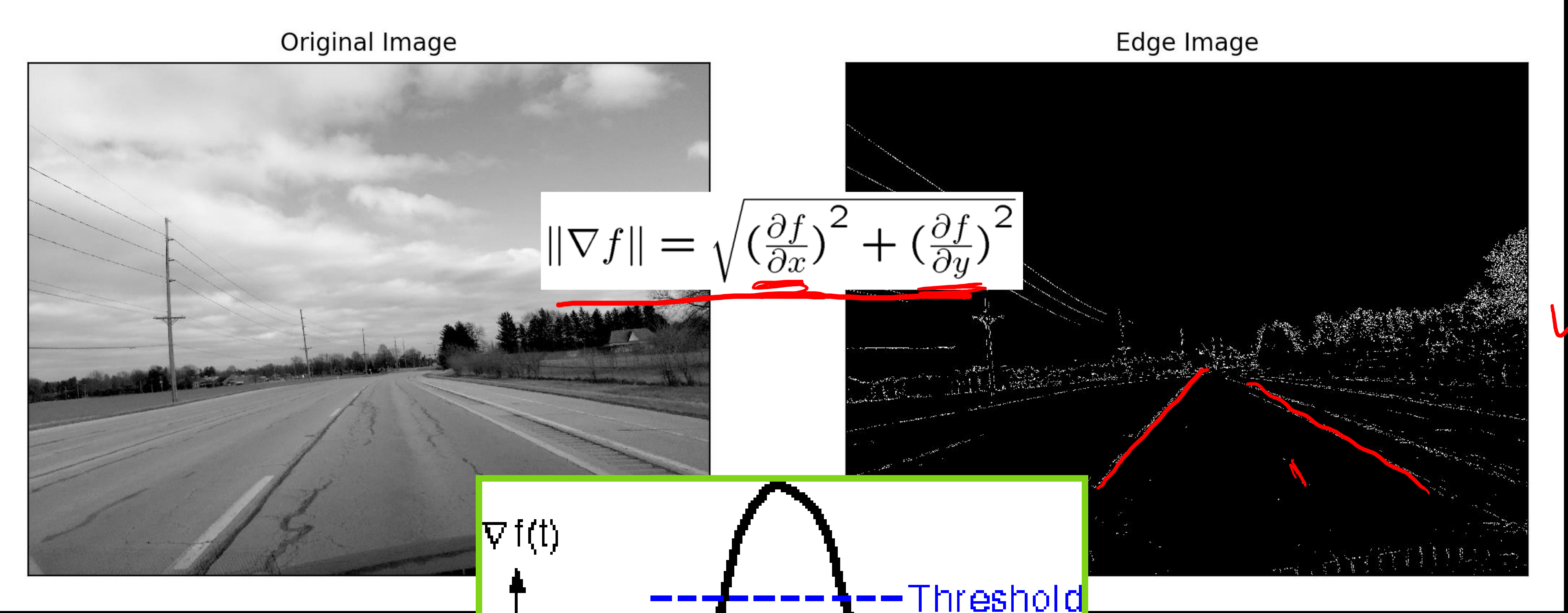
x-direction



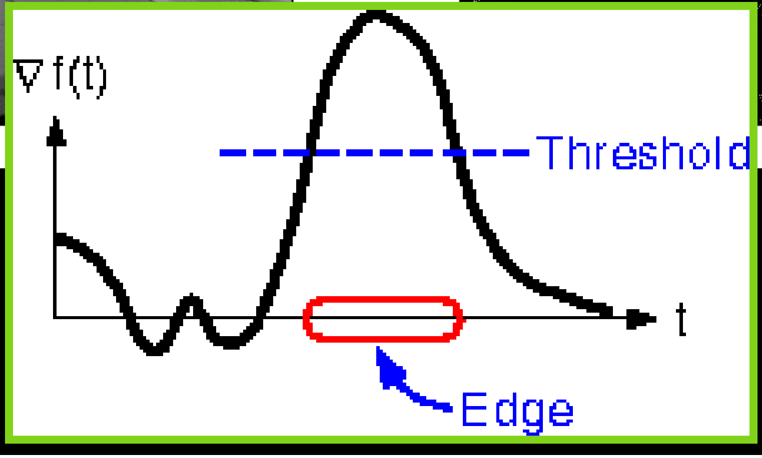
y-direction



Building an edge detector

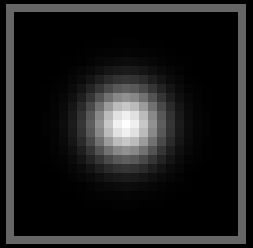


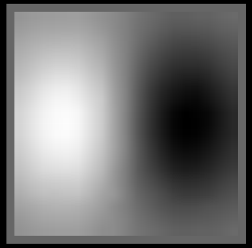
$$||\nabla f|| = \sqrt{(\frac{\partial f}{\partial x})^2 + (\frac{\partial f}{\partial y})^2}$$

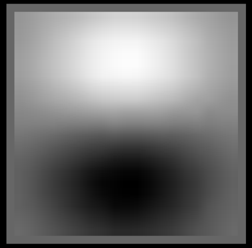


Review: smoothing vs. derivative filters

- Smoothing filters
  - Gaussian: remove "high-frequency" components; "low-pass" filter
  - What should the values sum to?
    - One: constant regions are not affected by the filter
- Derivative filters
  - Derivatives of Gaussian
  - What should the values sum to?
    - Zero: no response in constant regions







Summary

- Convolution as translation invariant linear operations on signals and images
  - Examples of filters for smoothing and sharpening images
  - Did not discuss: how to pick a filter for different types of noise?
  - Did not discuss: nice properties of Gaussian Filters (like sparability)
- Explored edge detection to understand shapes and semantic meaning in an image
  - Use partial derivatives to determine changes in intensity
  - Did not discuss: how does thresholding impact the edges detected?
  - Did not discuss: advanced edge detectors (Canny Edge Detector)
- Next time: the basics of object recognition!

Backup slides

Separability of the Gaussian filter

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

$$= \left(\frac{1}{\sqrt{2\pi}\sigma} \exp^{-\frac{x^2}{2\sigma^2}} \right) \left(\frac{1}{\sqrt{2\pi}\sigma} \exp^{-\frac{y^2}{2\sigma^2}} \right)$$

The 2D Gaussian can be expressed as the product of two functions, one a function of x and the other a function of y

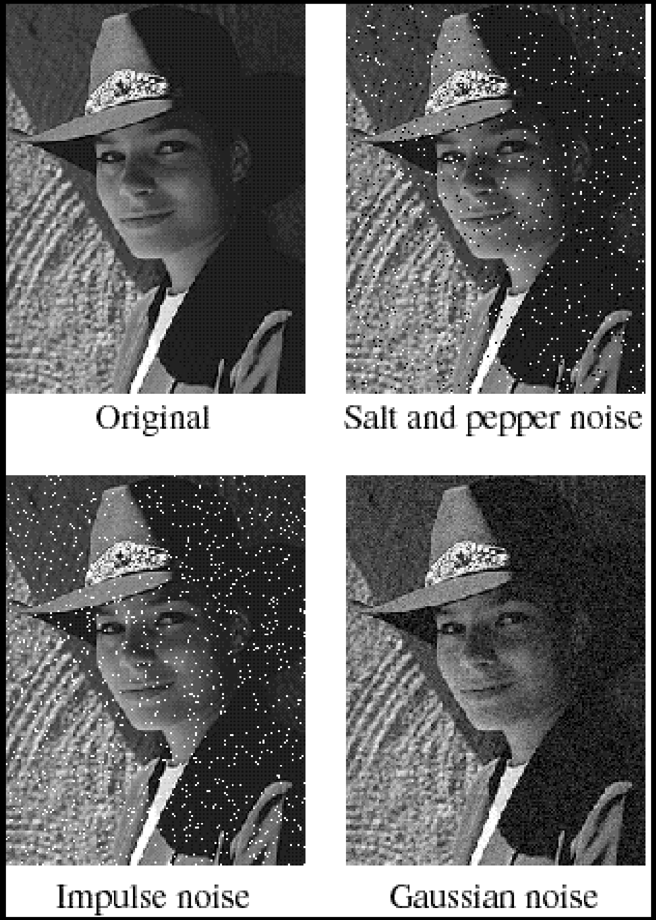
In this case, the two functions are the (identical) 1D Gaussian

Why is separability useful?

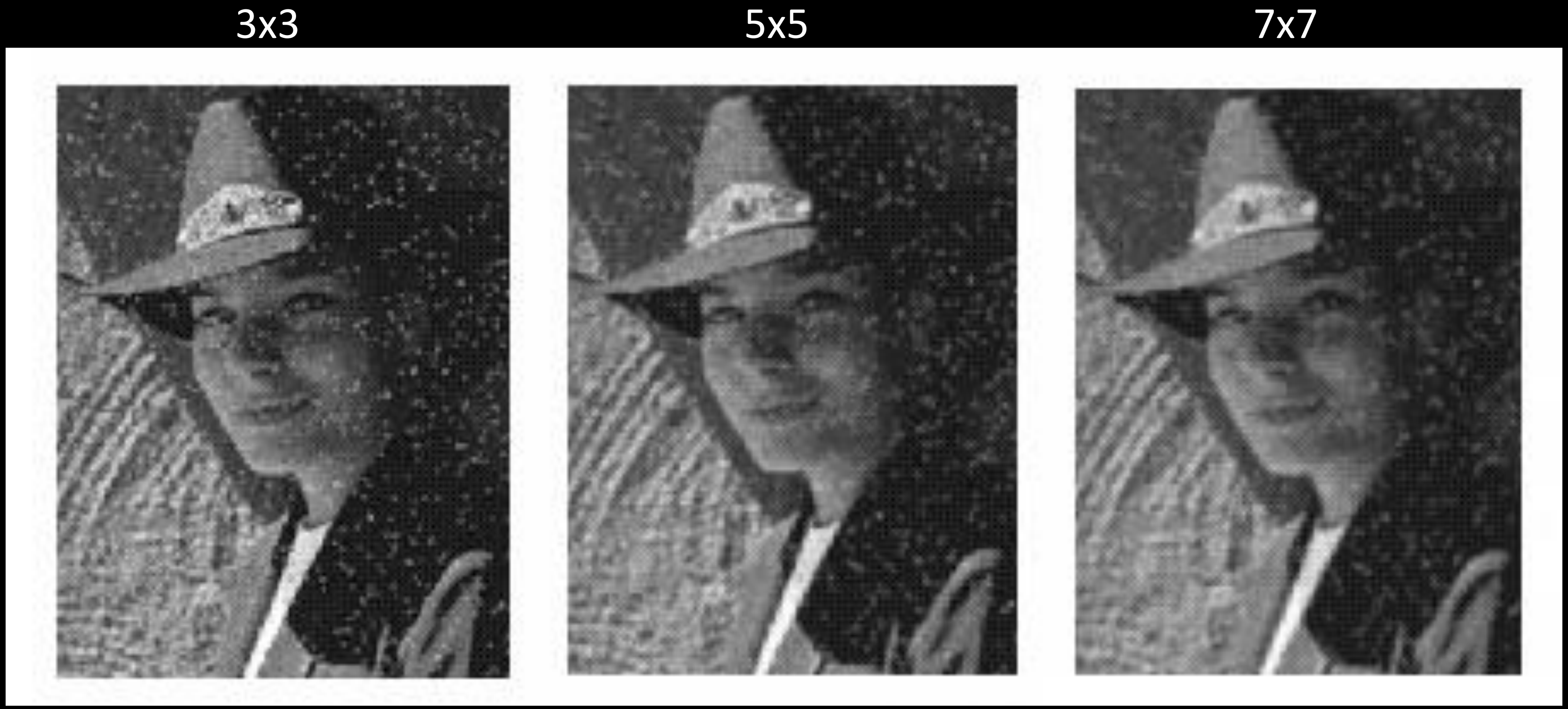
- Separability means that a 2D convolution can be reduced to two 1D convolutions (one along rows and one along columns)
- What is the complexity of filtering an $n \times n$ image with an $m \times m$ kernel?
  - $O(n^2 m^2)$
- What if the kernel is separable?
  - $O(n^2 m)$

Noise in Images

- 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

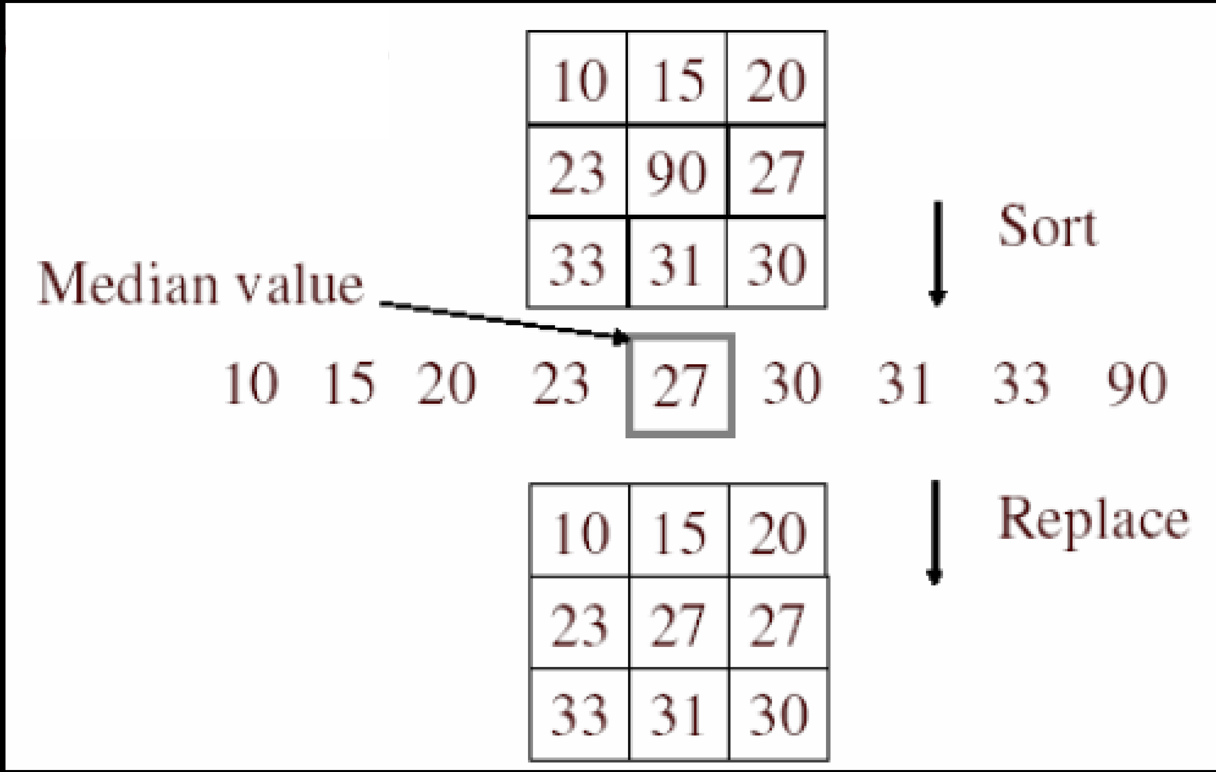


Reducing salt-and-pepper noise



Alternative idea: Median filtering

- A median filter operates over a window by selecting the median intensity in the window



Median filter

- Is median filtering linear?
- Let's try filtering

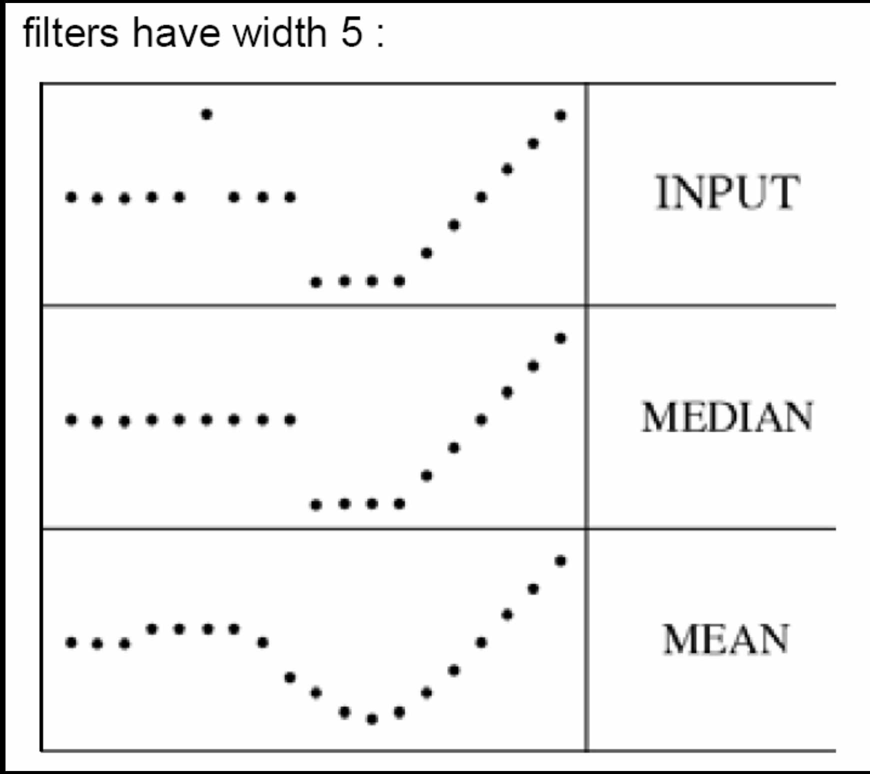
$$\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 2\\ 2 & 2 & 2\end{bmatrix} + \begin{bmatrix}0 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 0\end{bmatrix} \to 1$$

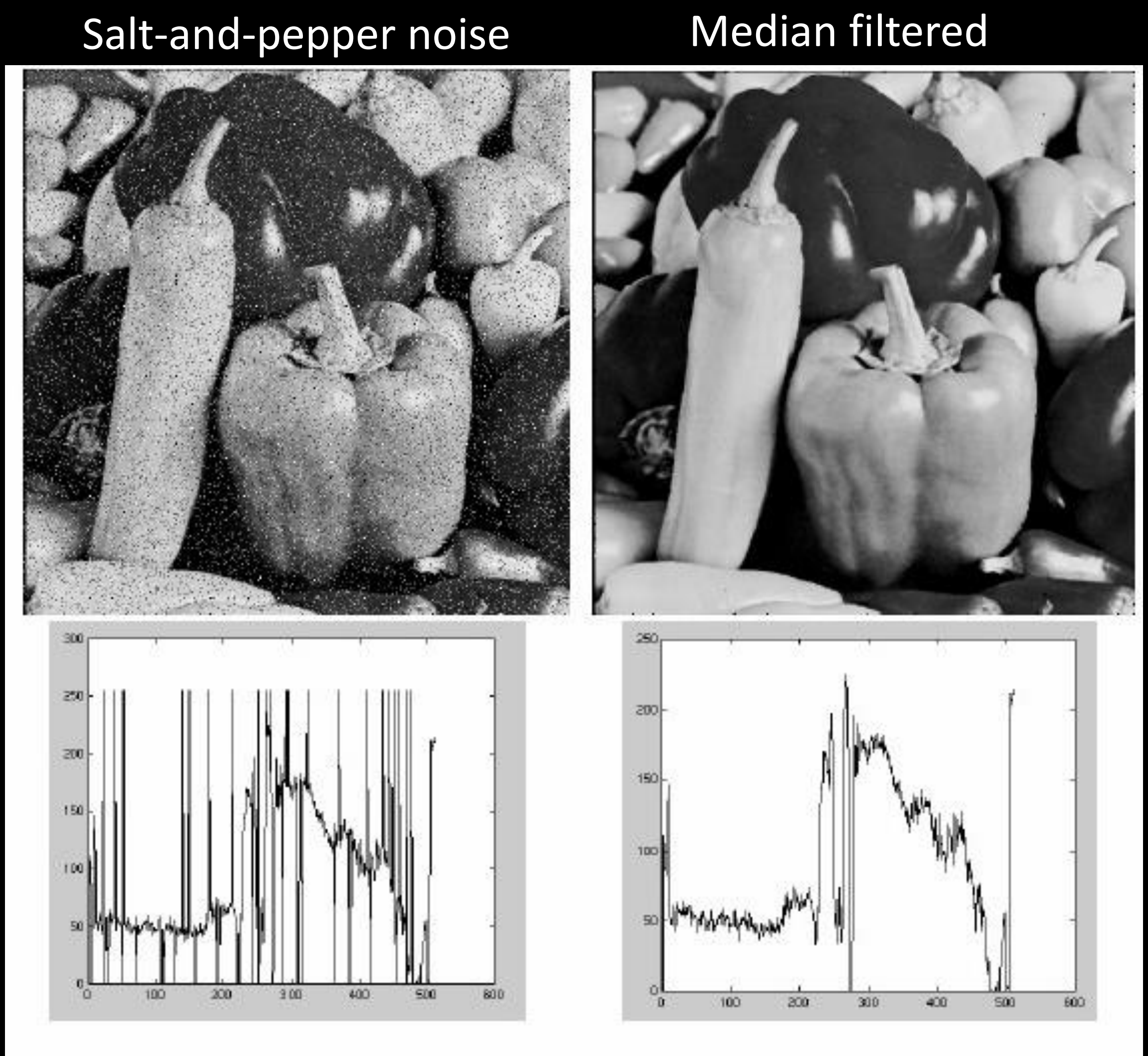
1 0

$$\begin{matrix}1 & 1 & 1\\ 1 & 2 & 2\\ 2 & 2 & 2\end{matrix} \xrightarrow{f} 2$$

Median filter

- What advantage does median filtering have over Gaussian filtering?
  - Robustness to outliers





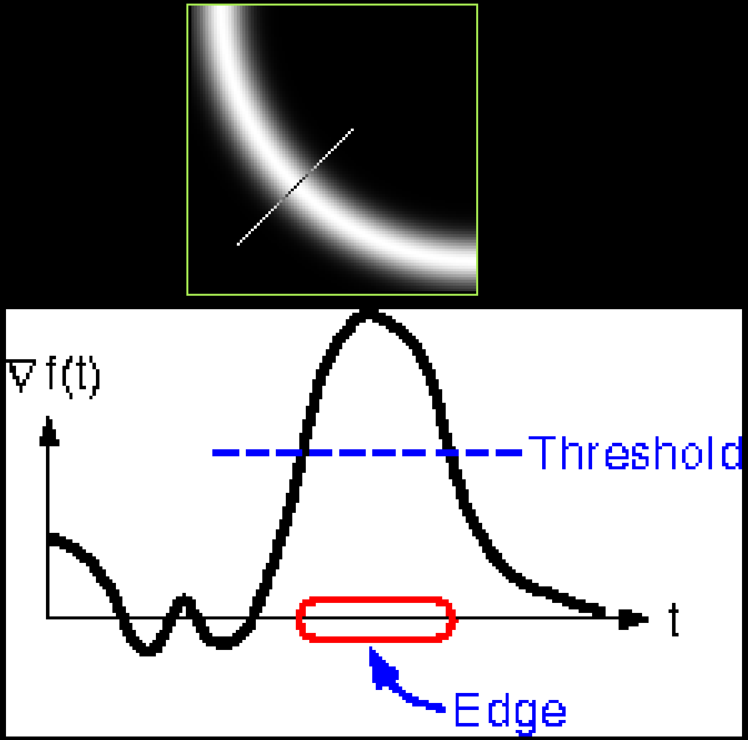
Gaussian vs. median filtering

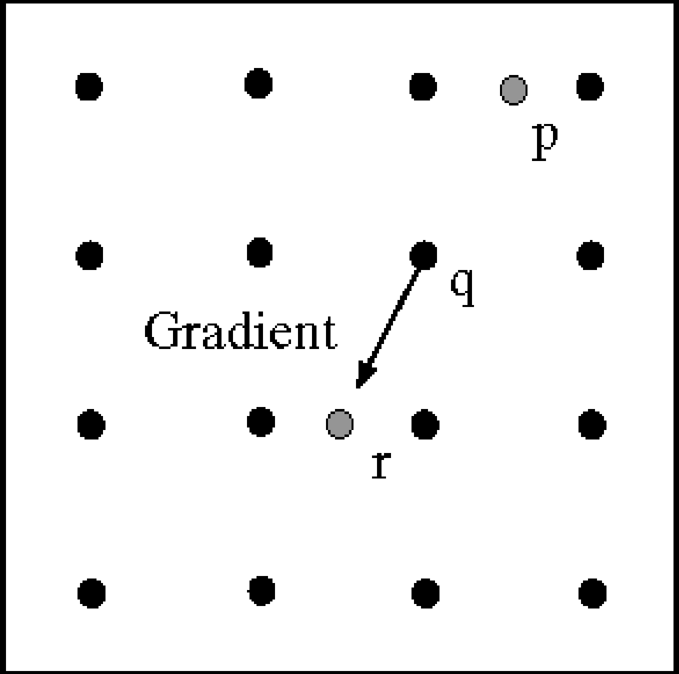
3x3 5x5 7x7

Gaussian

Median

Non-maximum suppression





- For each location q above threshold, check that the gradient magnitude is higher than at neighbors p and r along the direction of the gradient
  - May need to interpolate to get the magnitudes at p and r