ACV – Applied Computer Vision

Bachelor Medientechnik & Creative Computing

Matthias Zeppelzauer
matthias.zeppelzauer@fhstp.ac.at

Djordje Slijepcevic
djordje.slijepcevic@fhstp.ac.at

Slide Credits

- Kristen Grauman
- S. Seitz,
- Marschner,
- S. Lazebnik
- L. Fei Fei

Outlook

- So far:
  - Pixel-based operations on images
  - Filter (smooting, sharpening)
- Today:
  - Finding structures in images!
  - (Template matching)
  - Edges

Roadmap

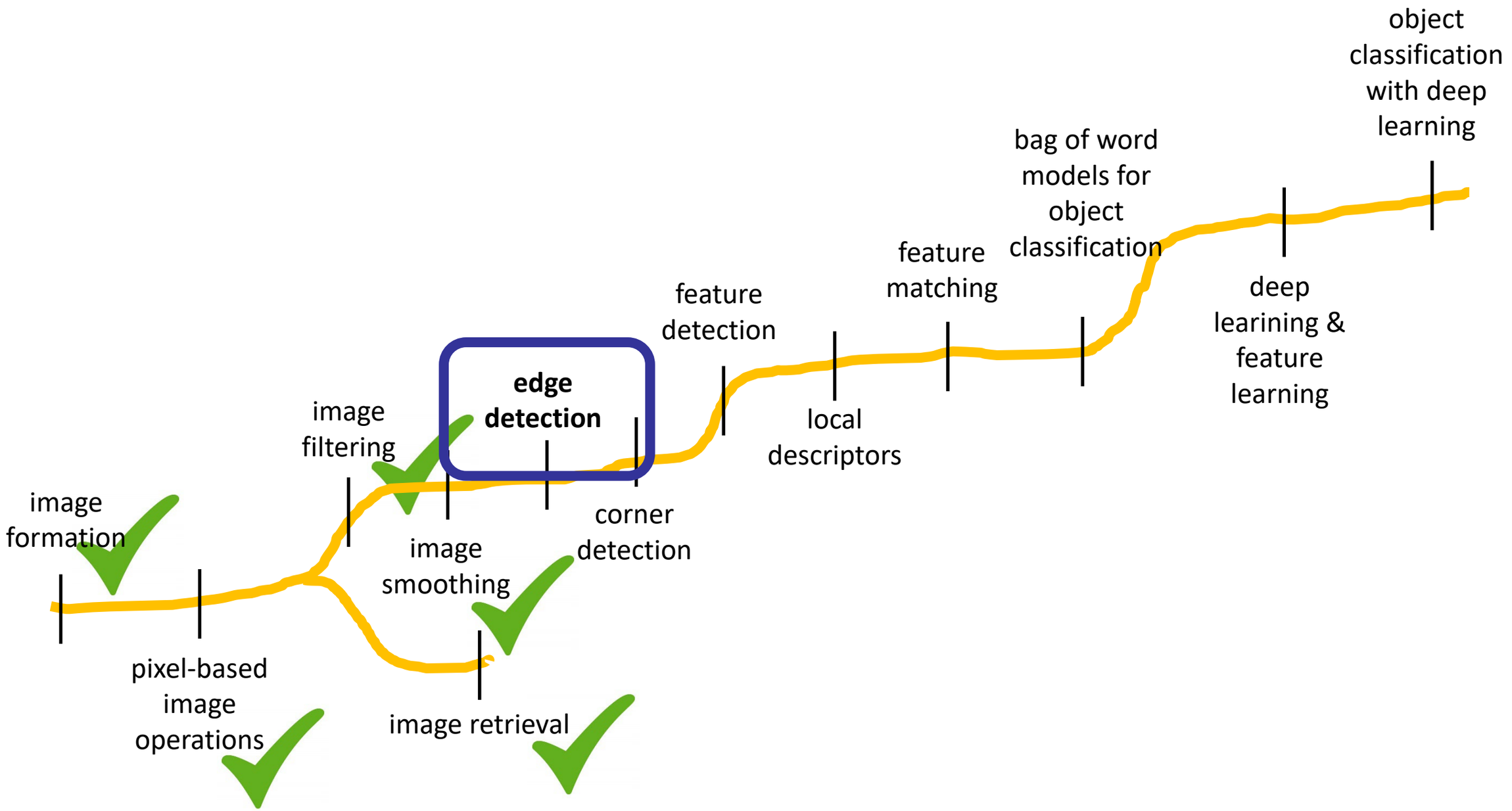
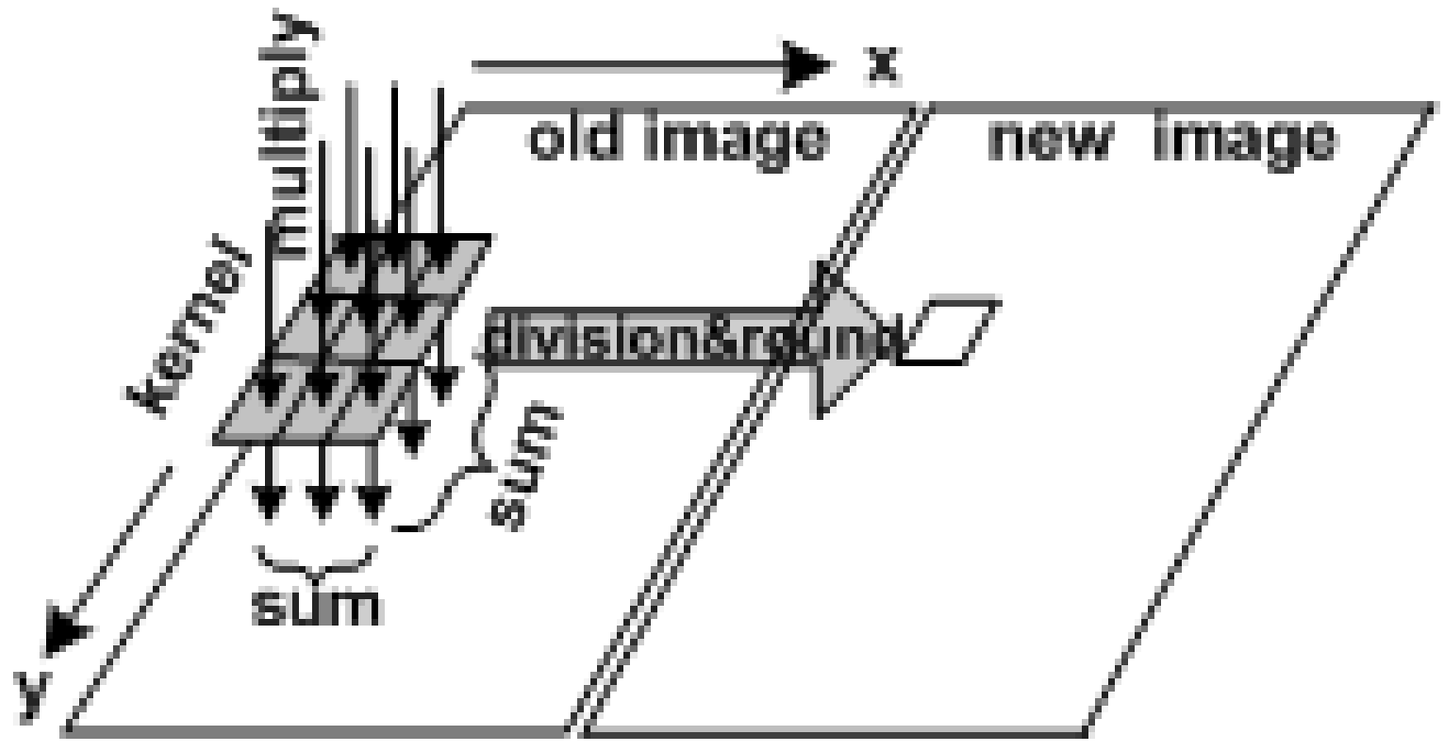


image formation
pixel-based image operations
image filtering
image smoothing
edge detection
corner detection
image retrieval
feature detection
local descriptors
feature matching
bag of word models for object classification
object classification with deep learning
deep learining & feature learning

Remember Filtering

- Moving a filter across an image
- Computing the sum of products between filter and image

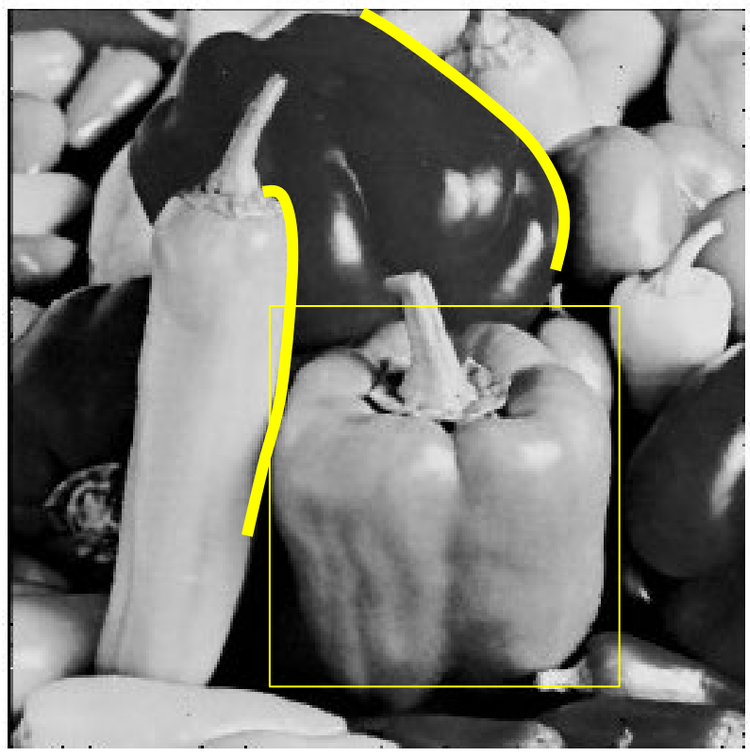


- Can we use this to find structures in the image?

Filters for features

- Previously, thinking of filtering as a way to remove or reduce **noise**

- Now, consider how filters will allow us to abstract higher-level "**features**".



  - Map raw pixels to an intermediate representation that will be used for subsequent processing
  - Goal: reduce amount of data, discard redundancy, preserve what's useful

Edge detection

- **Goal**: map image from 2d array of pixels to a set of curves or line segments or contours.
- **Why?**

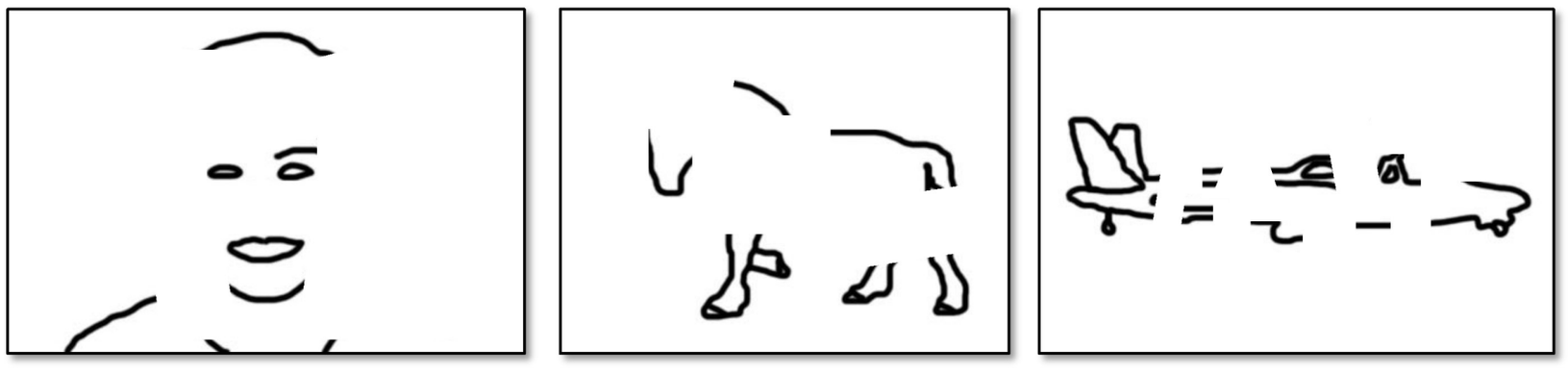


Figure from J. Shotton et al., PAMI 2007

- **Main idea**: look for strong gradients, post-process

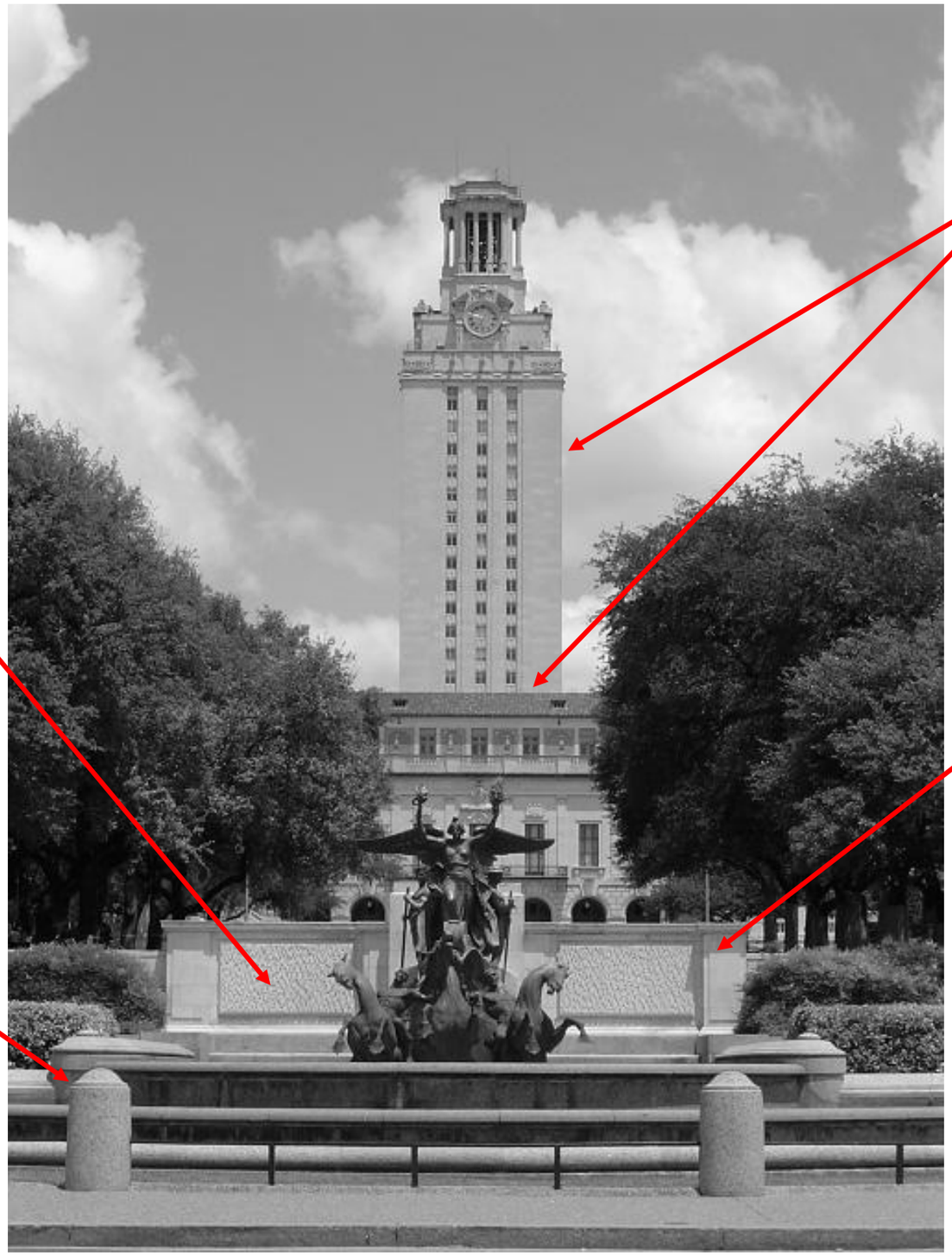
Edge Detection

- **Goal:** Identify sudden changes (discontinuities) in an image
  - Intuitively, most semantic and shape information from the image can be encoded in the edges
  - More compact than pixels

- **Ideal:** artist's line drawing (but artist is also using object-level knowledge and world knowledge)

What can cause an edge?

Reflectance change: appearance information, texture

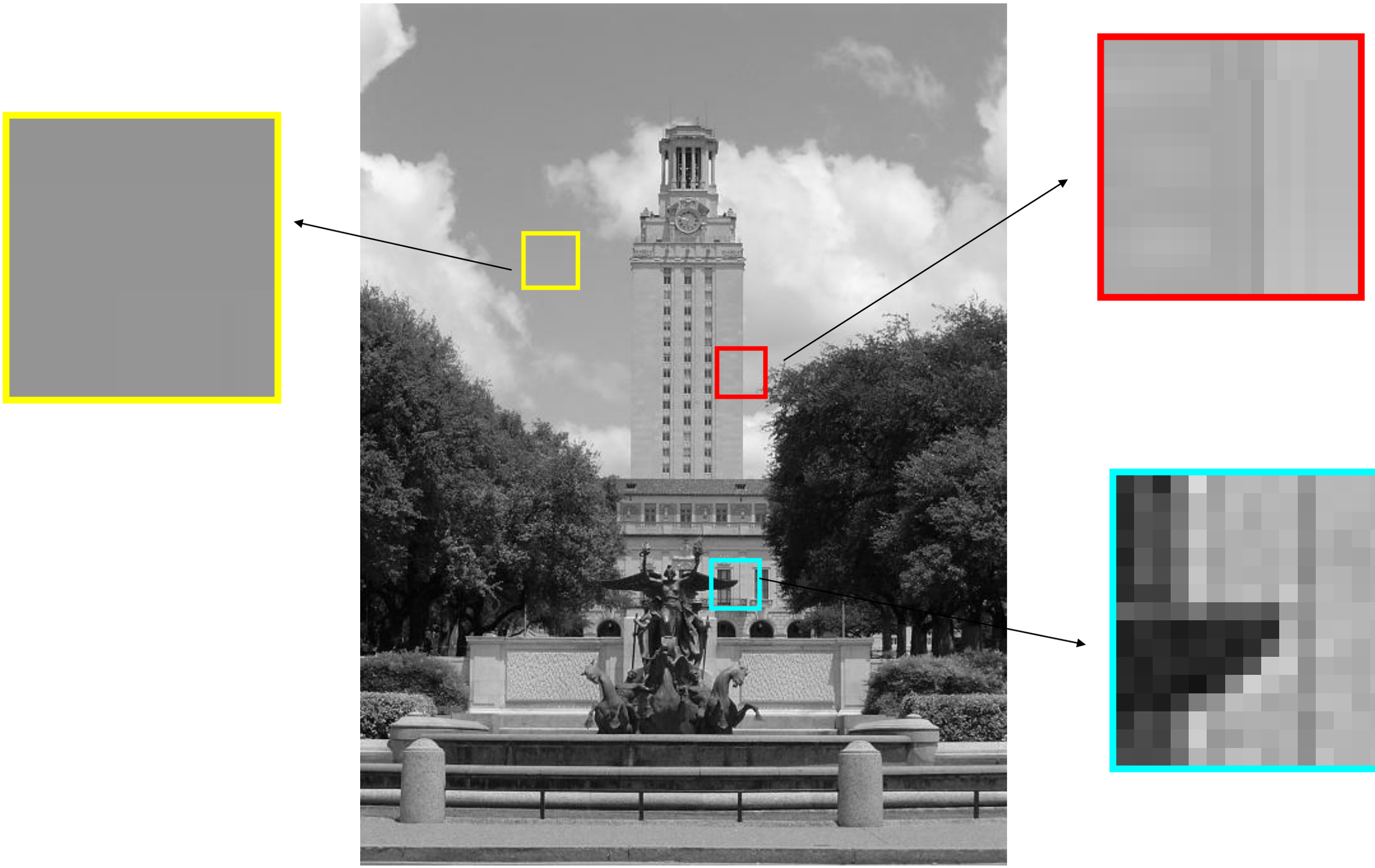


Depth discontinuity: object boundary

Change in surface orientation: shape

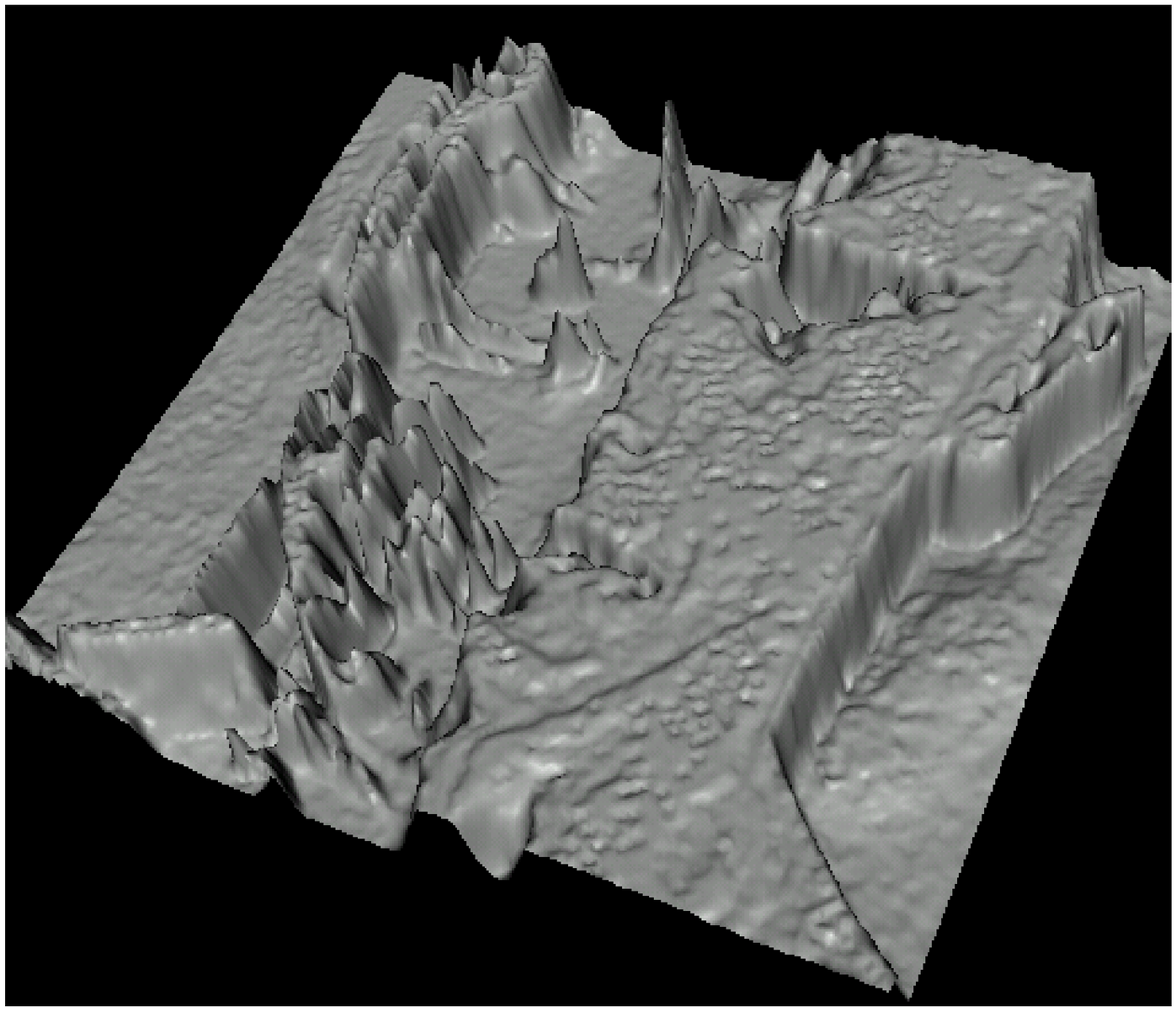
Cast shadows

Contrast and invariance



Recall : Images are functions

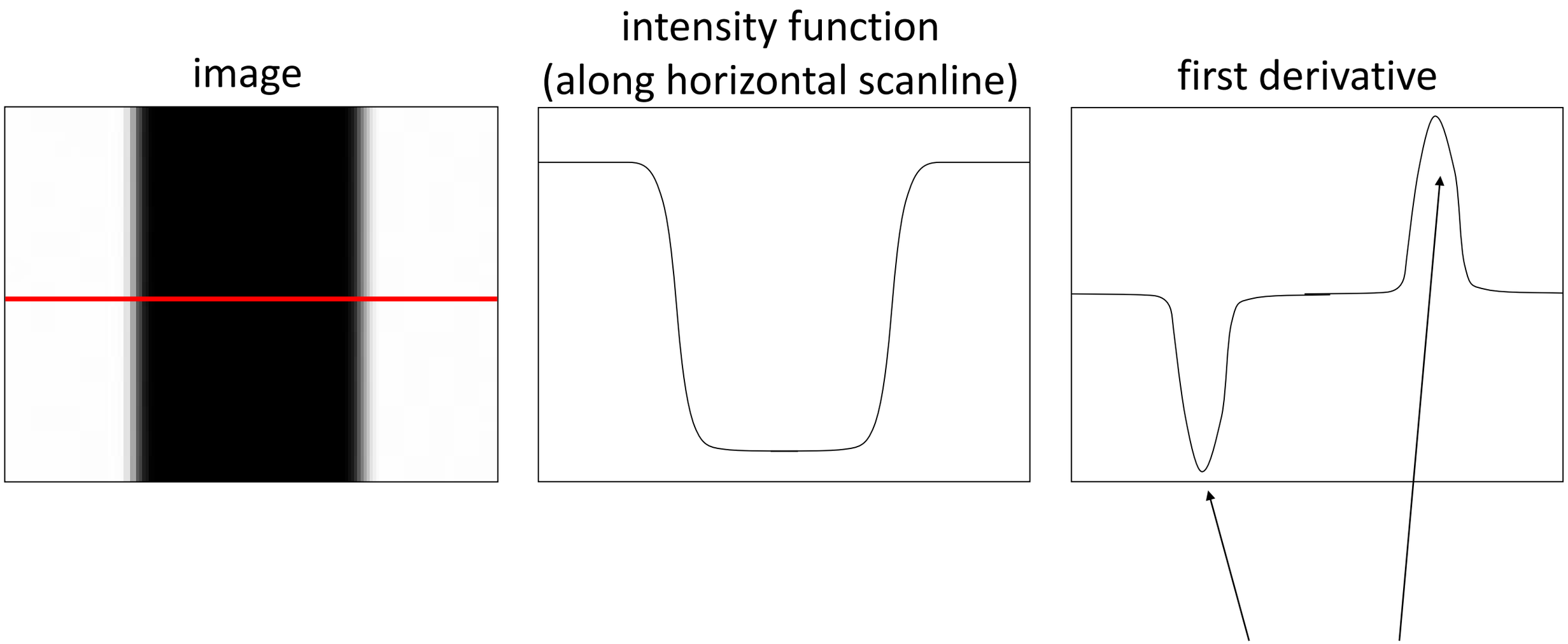
- Edges look like steep cliffs



Source: S. Seitz

Derivatives and edges

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



Source: L. Lazebnik

Differentiation

- For discrete 2D function, f(x,y), the partial derivative is:

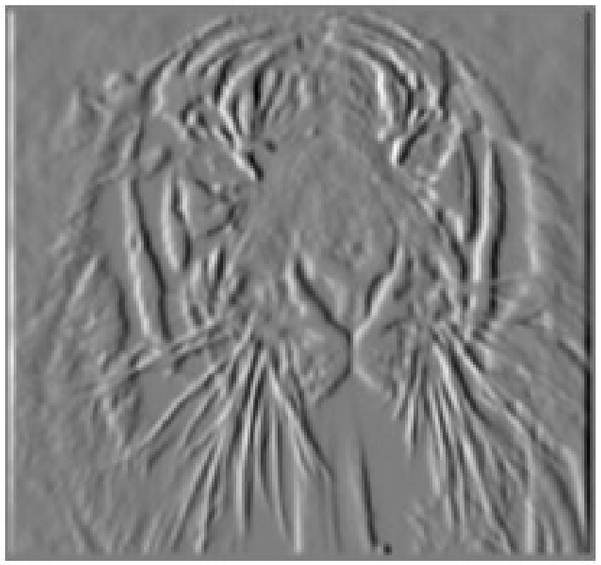
$$\frac{\partial f(x,y)}{\partial x} \approx \frac{f(x+1,y)-f(x,y)}{1}$$

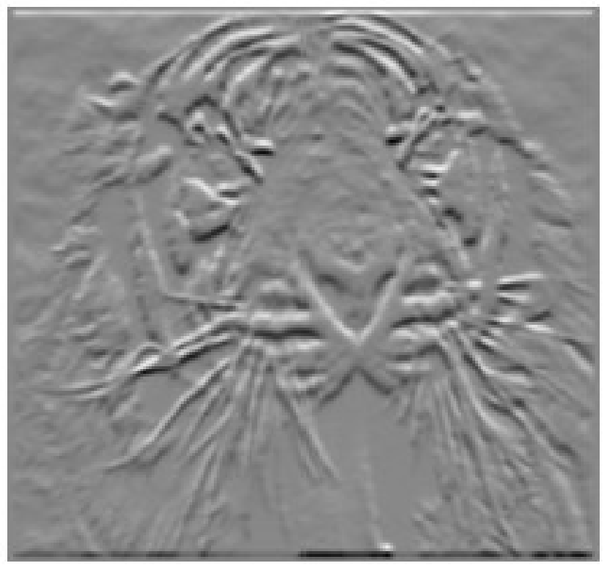
- To implement above as convolution, what would be the associated filter?

Partial derivatives of an image

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

-1	1





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

-1
1

Which shows changes with respect to x?

Typical Edge filters

Prewitt: $M_x =$ | -1 | 0 | 1 |
|---|---|---|
| -1 | 0 | 1 |
| -1 | 0 | 1 |

; $M_y =$ | 1 | 1 | 1 |
|---|---|---|
| 0 | 0 | 0 |
| -1 | -1 | -1 |

Sobel: $M_x =$ | -1 | 0 | 1 |
|---|---|---|
| -2 | 0 | 2 |
| -1 | 0 | 1 |

; $M_y =$ | 1 | 2 | 1 |
|---|---|---|
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Roberts: $M_x =$ | 0 | 1 |
|---|---|
| -1 | 0 |

; $M_y =$ | 1 | 0 |
|---|---|
| 0 | -1 |

Typical Edge filters

Prewitt: $M_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$; $M_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$

Sobel: $M_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$; $M_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$

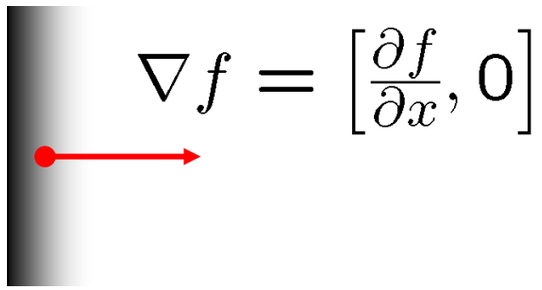
Roberts: $M_x = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$; $M_y = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$

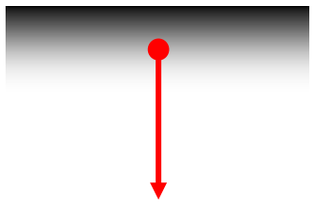
In general: the filters yield strong response on image locations which look similar – what does that mean?

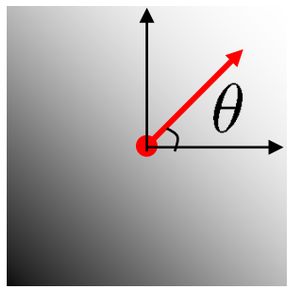
Image gradient

- The gradient of an image: $\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$

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







- The gradient direction (orientation of edge normal) is given by:

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

- The *edge strength* is given by the gradient magnitude

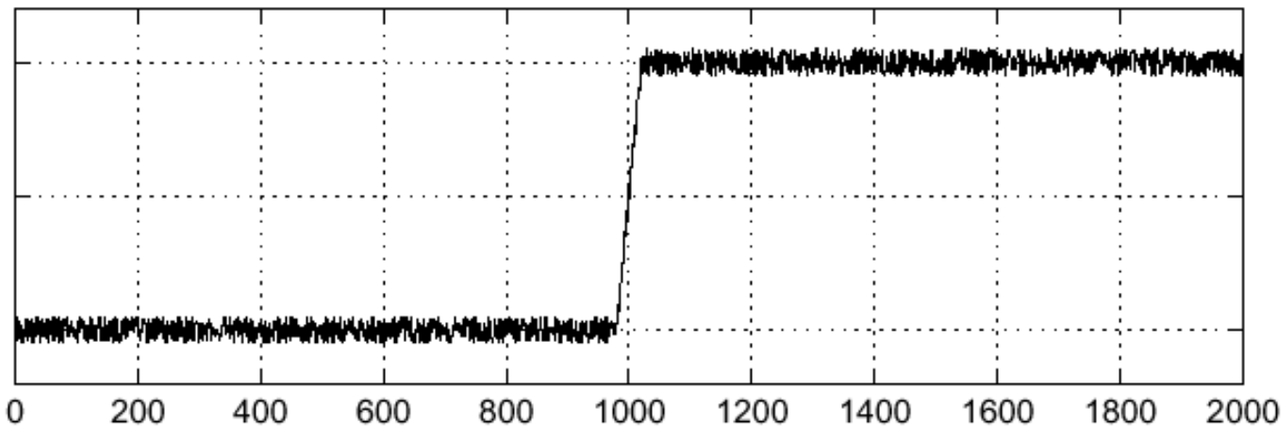
$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

Slide credit S. Seitz

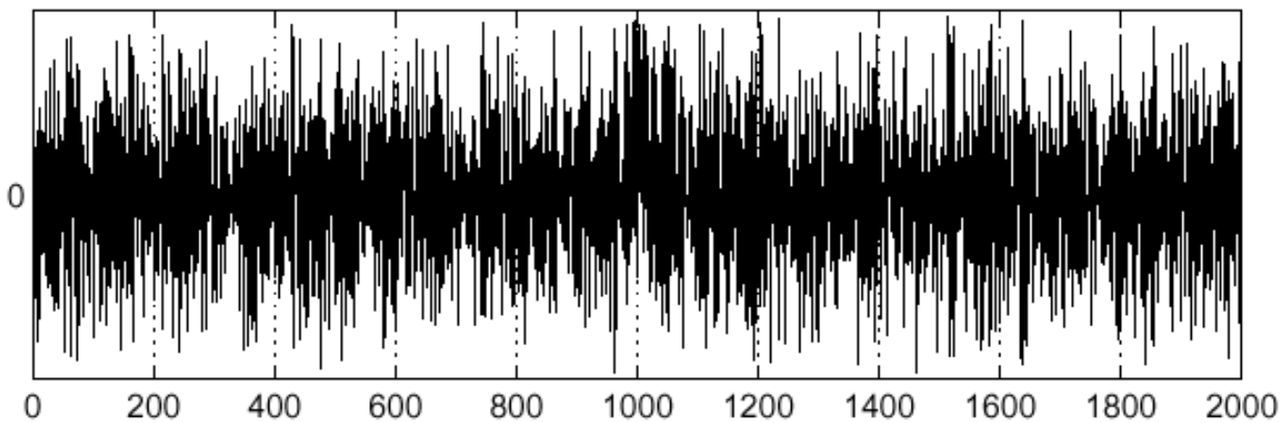
Effects of noise on first derivation (gradient)

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

$f(x)$



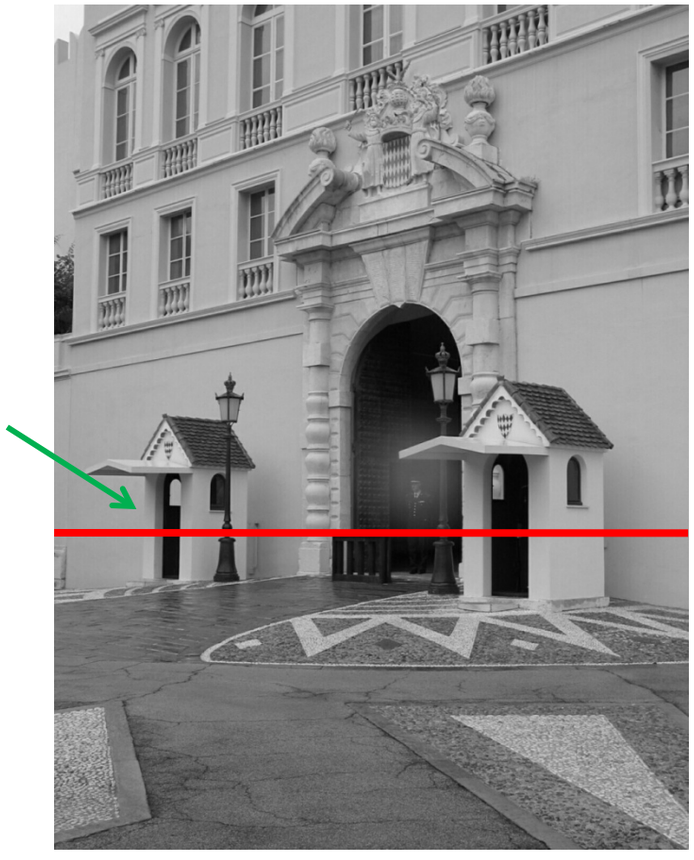
$\frac{d}{dx} f(x)$

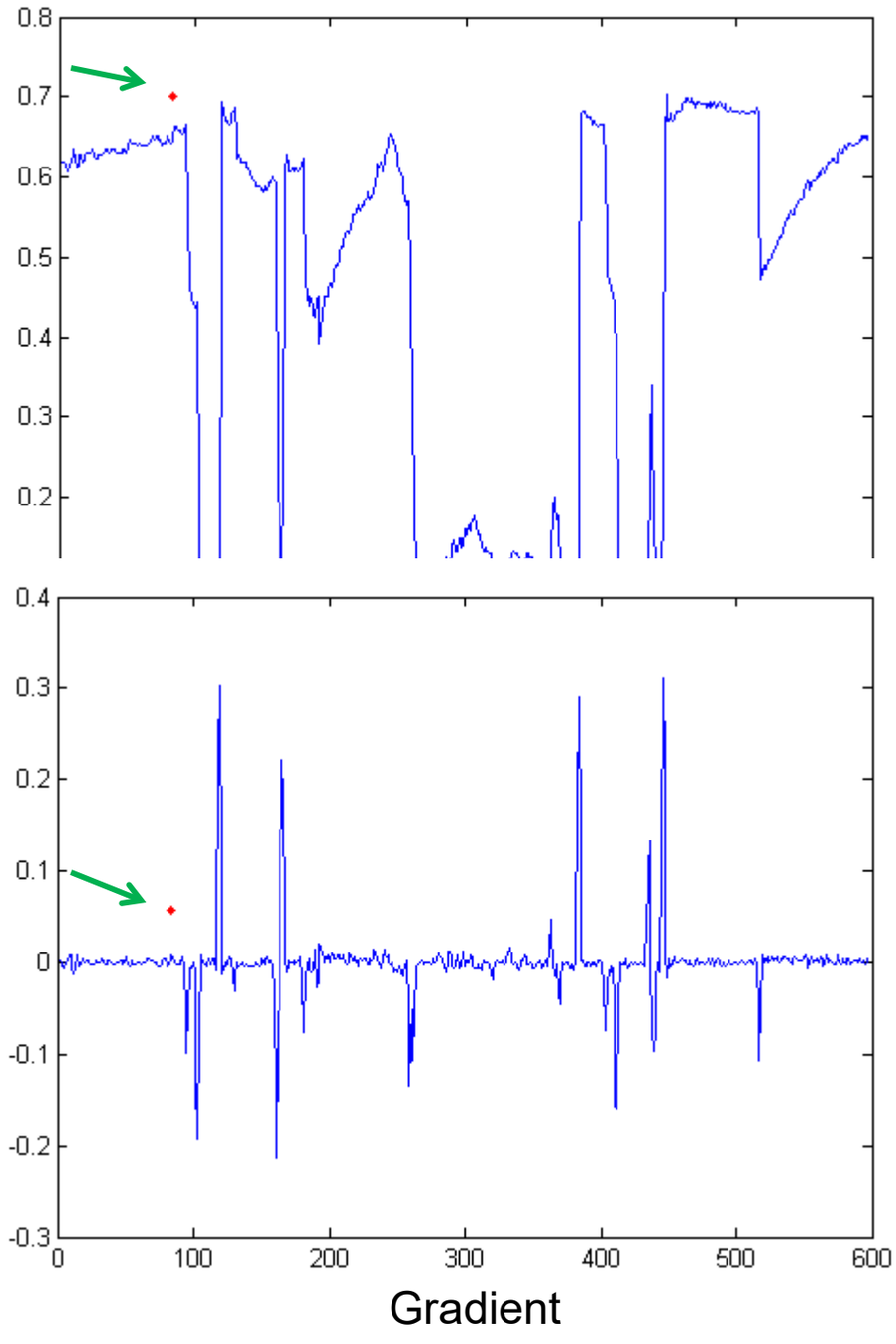


- Where is the edge?

Example

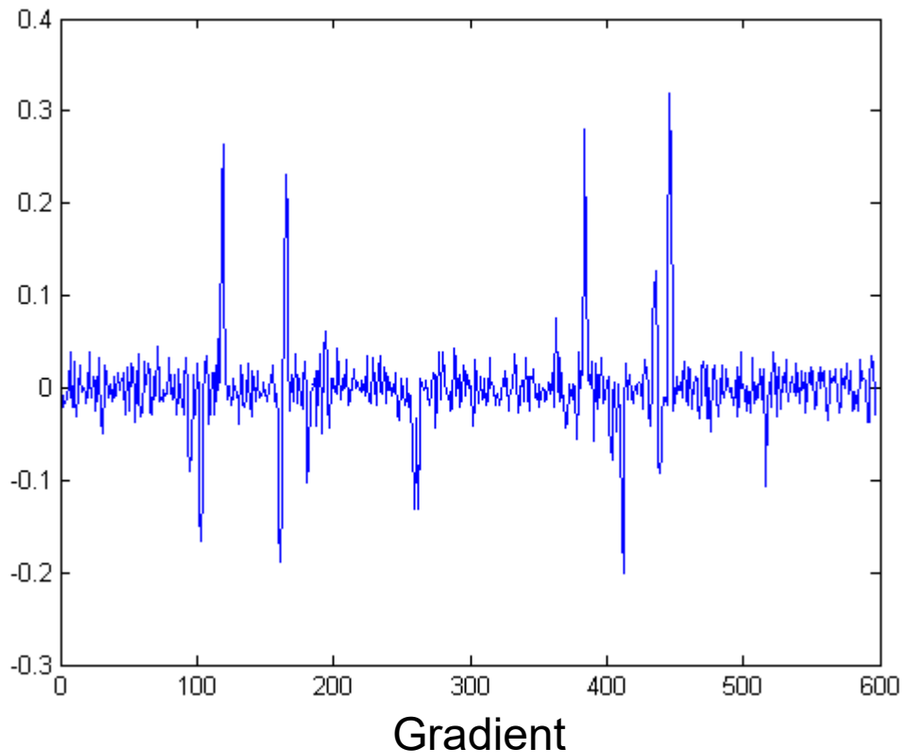
- Intensity profile:





Example with added Noise

- Intensity profile:



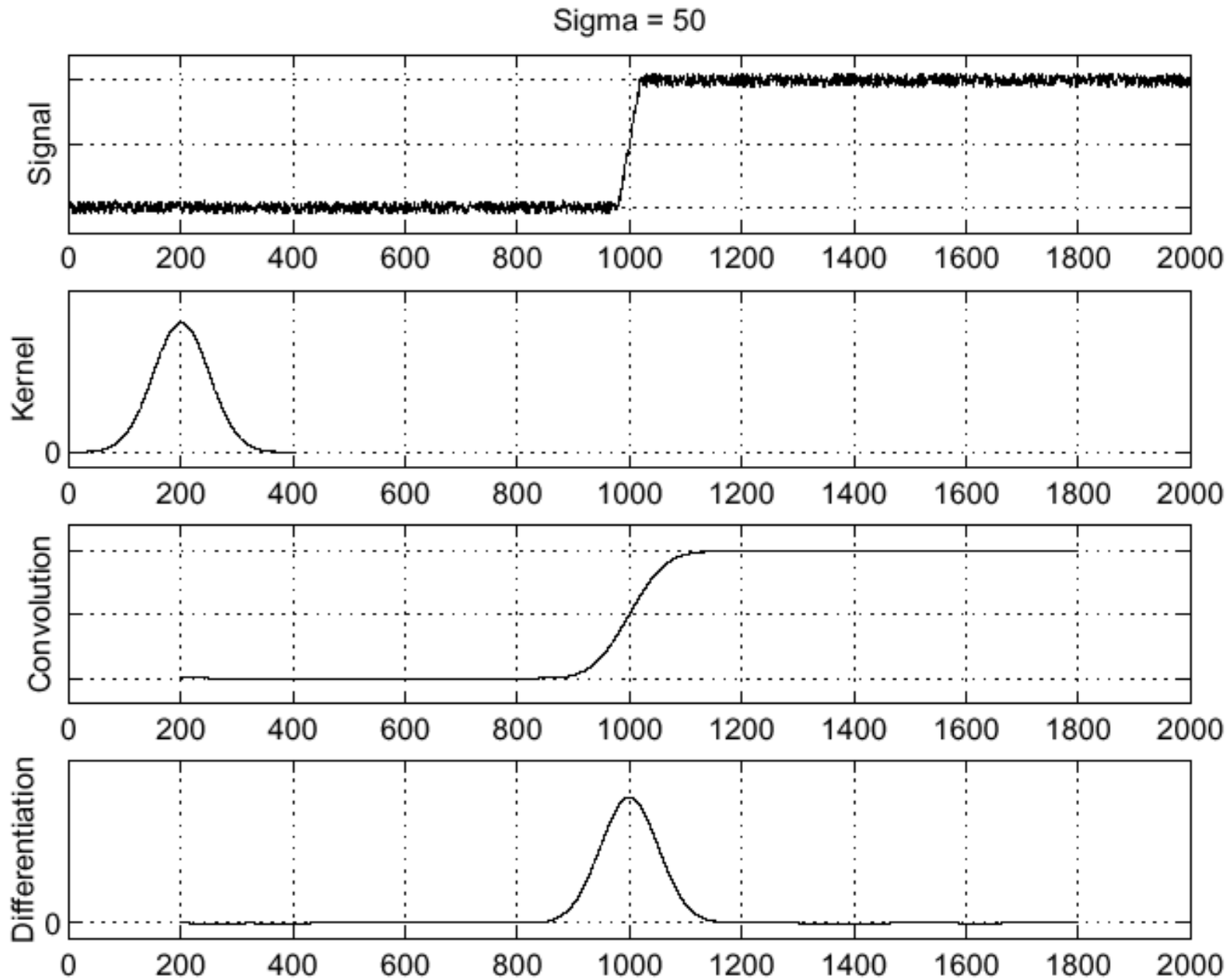
Solution: smooth first to remove noise

f

h

$h \star f$

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



Where is the edge? Look for peaks in $\frac{\partial}{\partial x}(h \star f)$

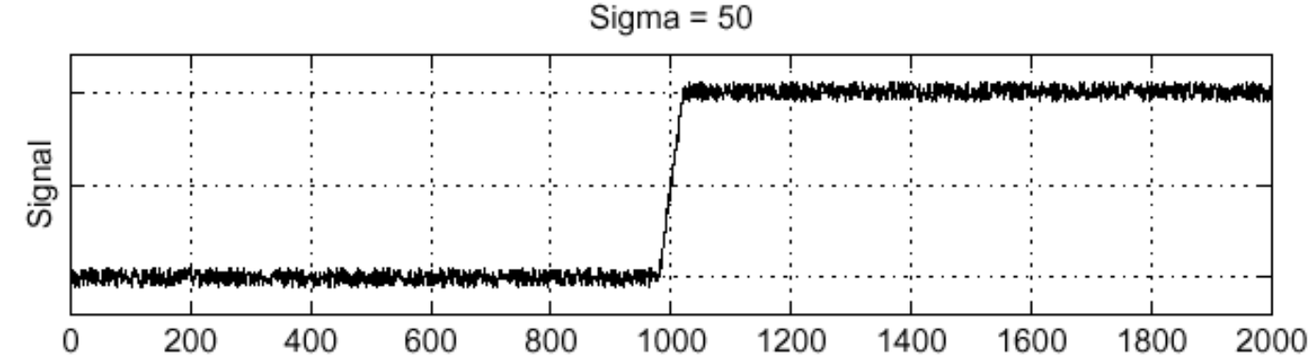
Derivative theorem of convolution

- Combine the filters first, i.e. take Gaussian and compute 1st derivation

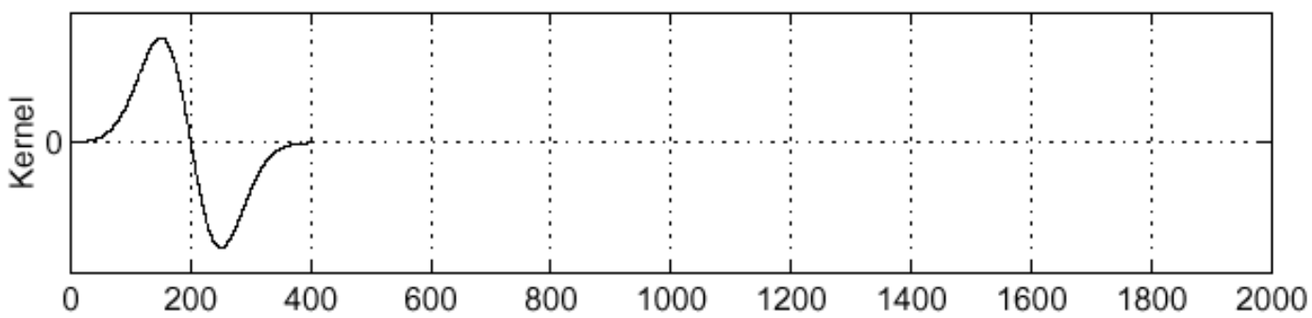
$$\frac{\partial}{\partial x}(h \star f) = (\frac{\partial}{\partial x}h) \star f$$

- And filter then → same result!

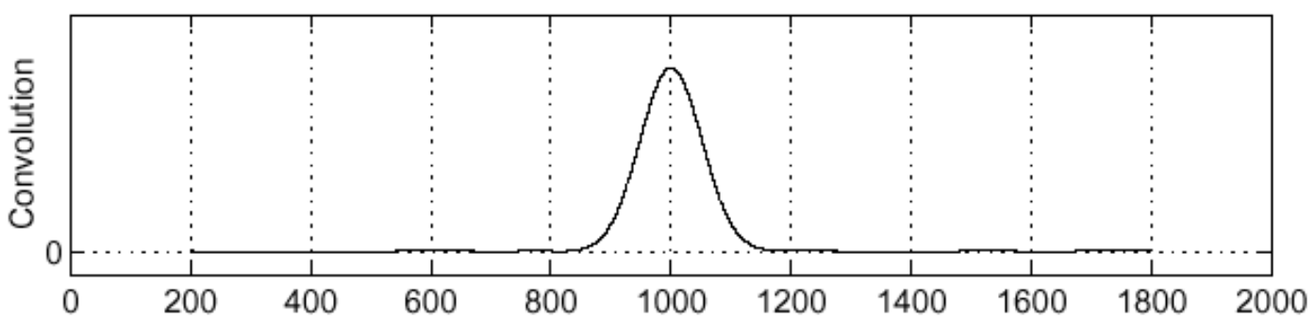
f



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



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

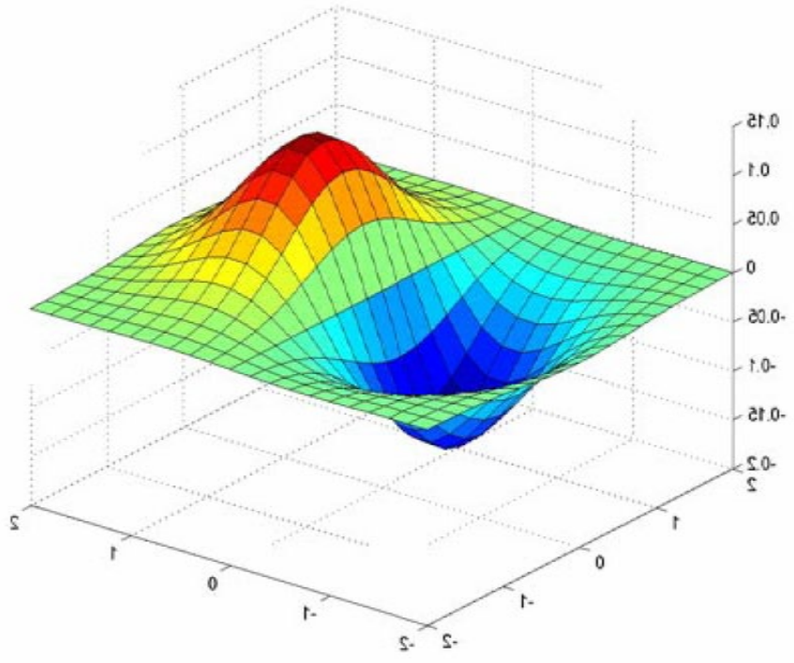


Derivative of Gaussian filter

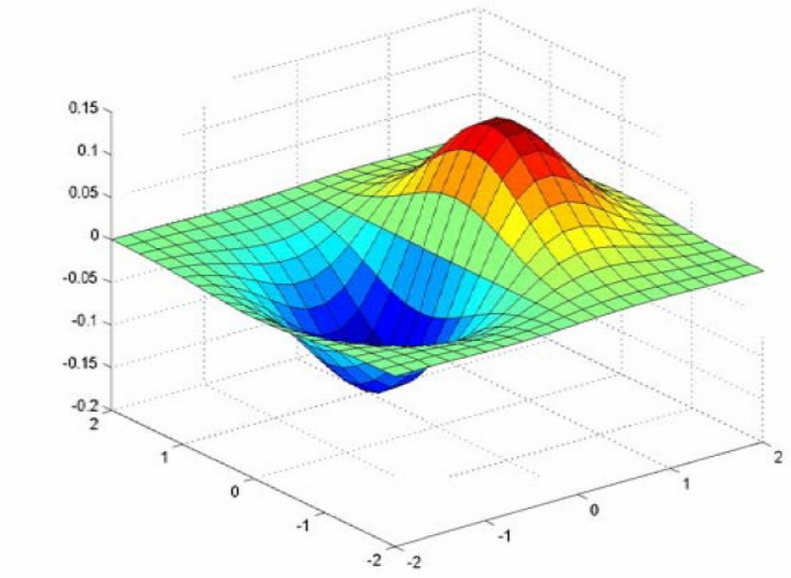
$$(I \otimes g) \otimes h = I \otimes (g \otimes h)$$

$$\begin{bmatrix} 0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \\ 0.0133 & 0.0596 & 0.0983 & 0.0596 & 0.0133 \\ 0.0219 & 0.0983 & 0.1621 & 0.0983 & 0.0219 \\ 0.0133 & 0.0596 & 0.0983 & 0.0596 & 0.0133 \\ 0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \end{bmatrix}$$

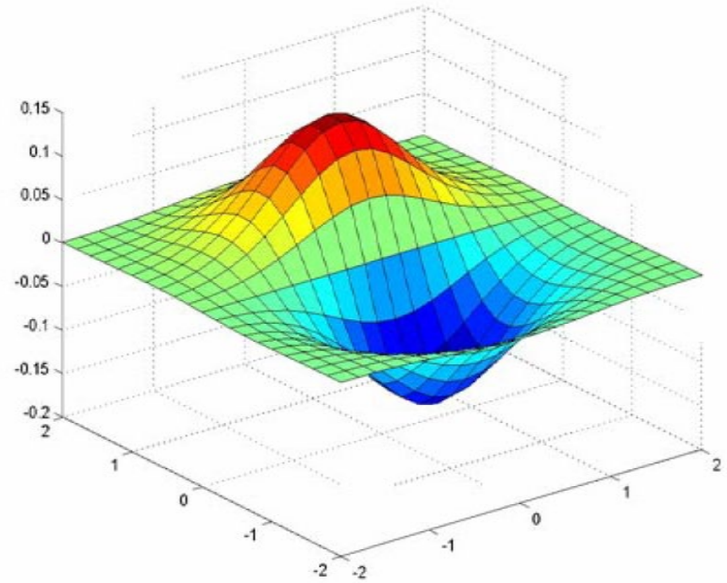
$$\otimes \begin{bmatrix} 1 & -1 \end{bmatrix}$$



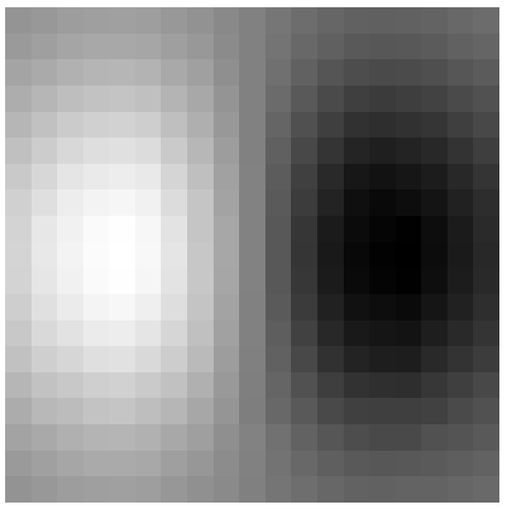
Derivative of Gaussian filters

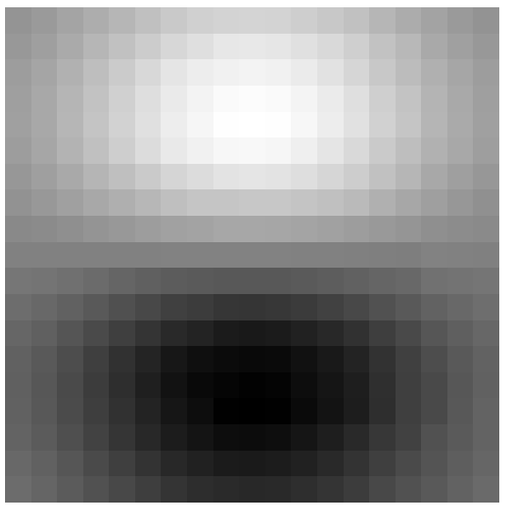


x-direction



y-direction





- Nice (noise robust) filters for horizontal and vertical edges!
- But what about other edge directions?

Source: L. Lazebnik

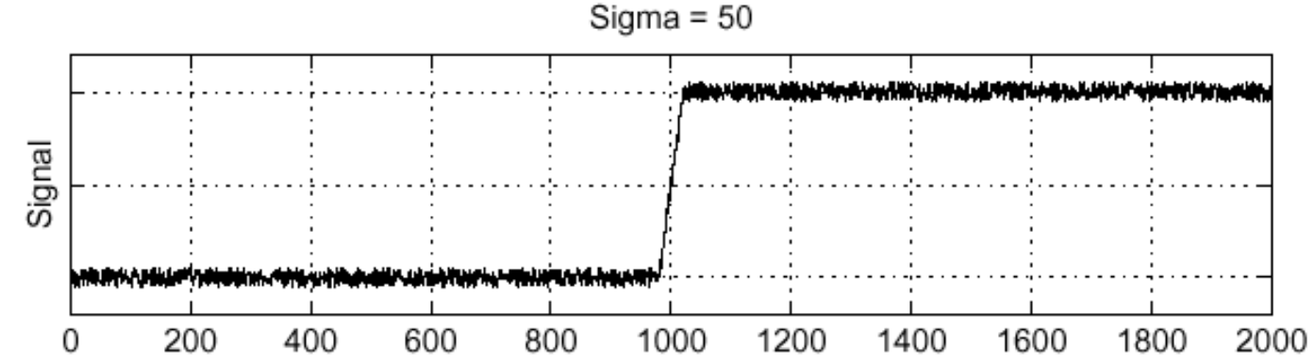
Derivative theorem of convolution

- Combine the filters first, i.e. take Gaussian and compute 1st derivation

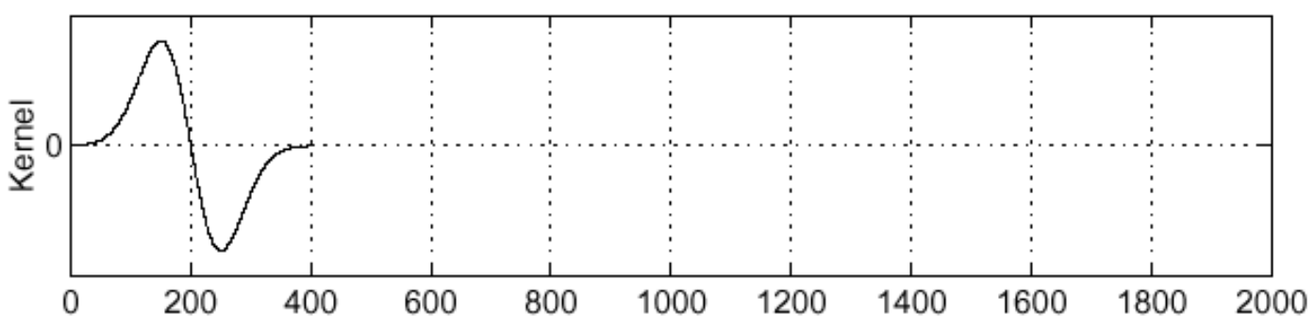
$$\frac{\partial}{\partial x}(h \star f) = (\frac{\partial}{\partial x}h) \star f$$

- And filter then → same result!

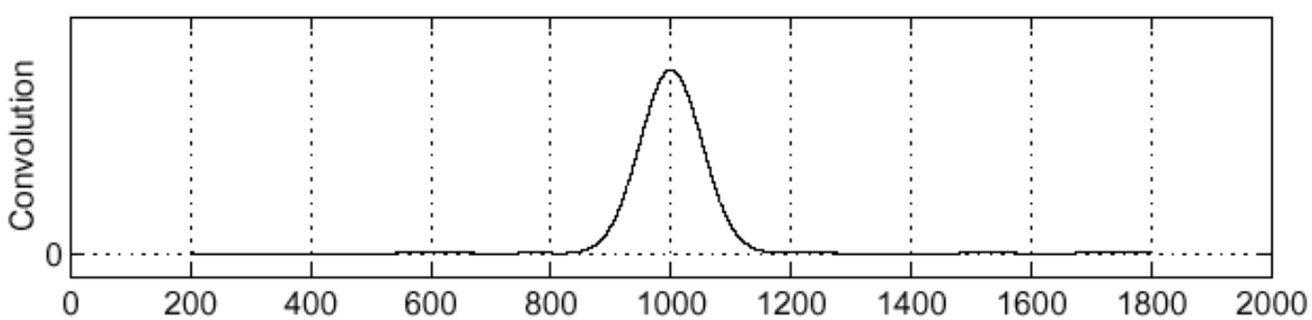
f



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



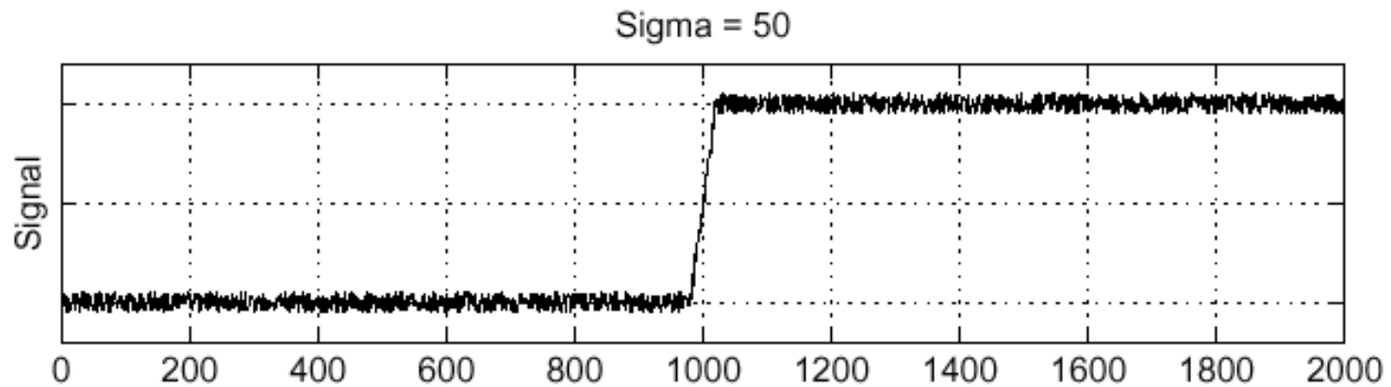
$(\frac{\partial}{\partial x}h) \star f$



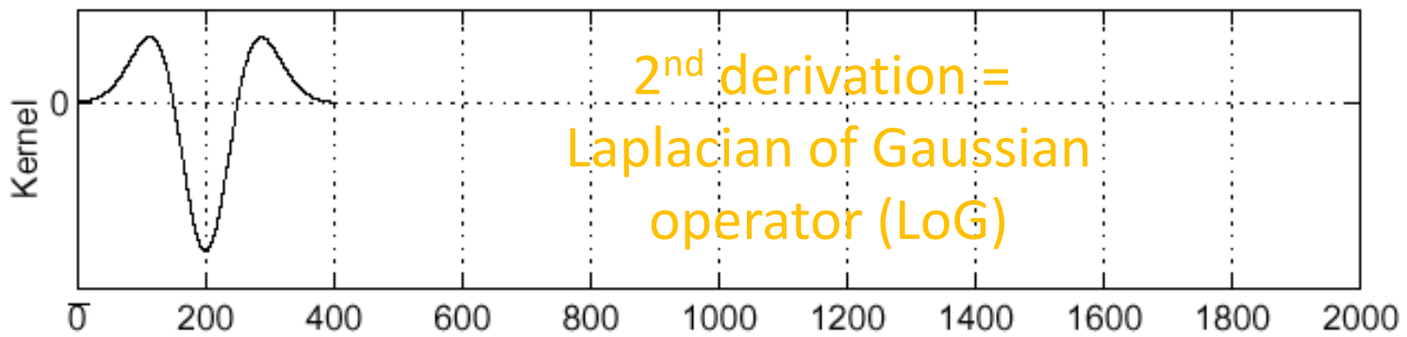
In 1D: Laplacian of Gaussian

- Consider $\frac{\partial^2}{\partial x^2}(h \star f)$

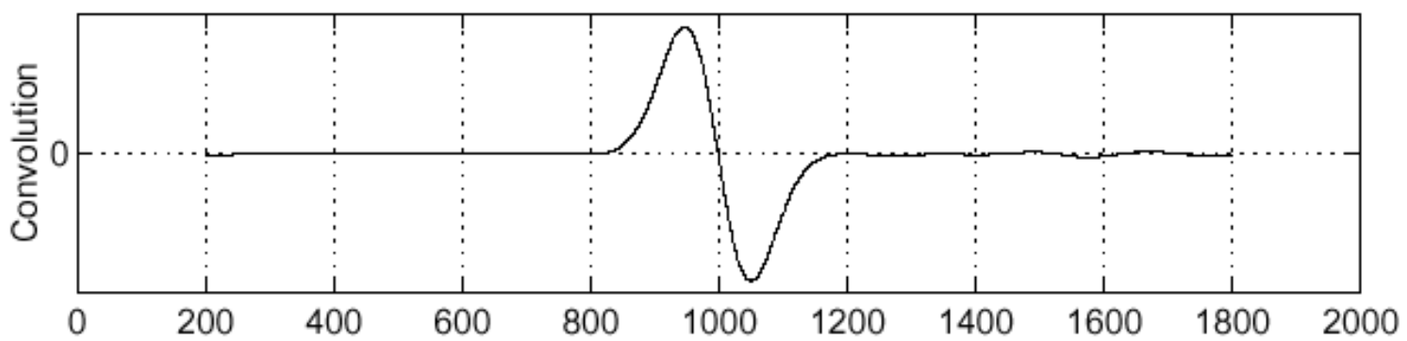
f



$\frac{\partial^2}{\partial x^2}h$

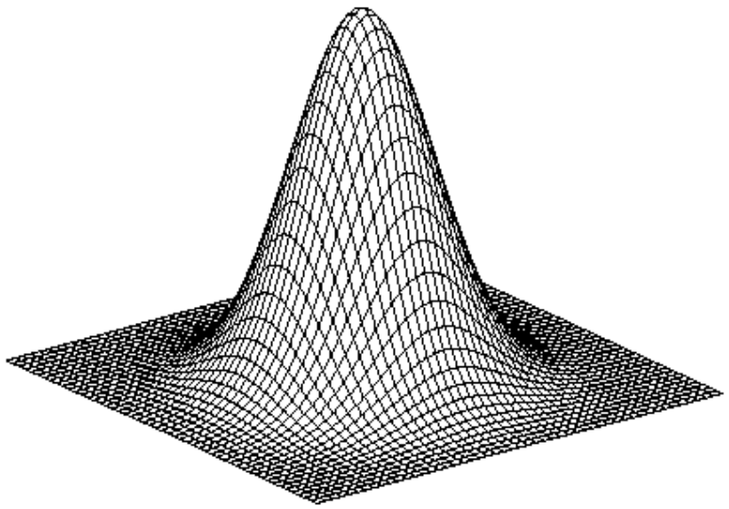


$(\frac{\partial^2}{\partial x^2}h) \star f$



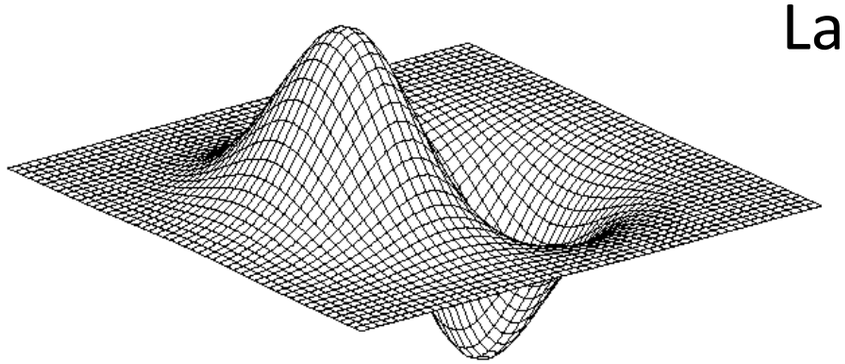
Where is the edge? Zero-crossings of bottom graph

2D edge detection filters



Gaussian

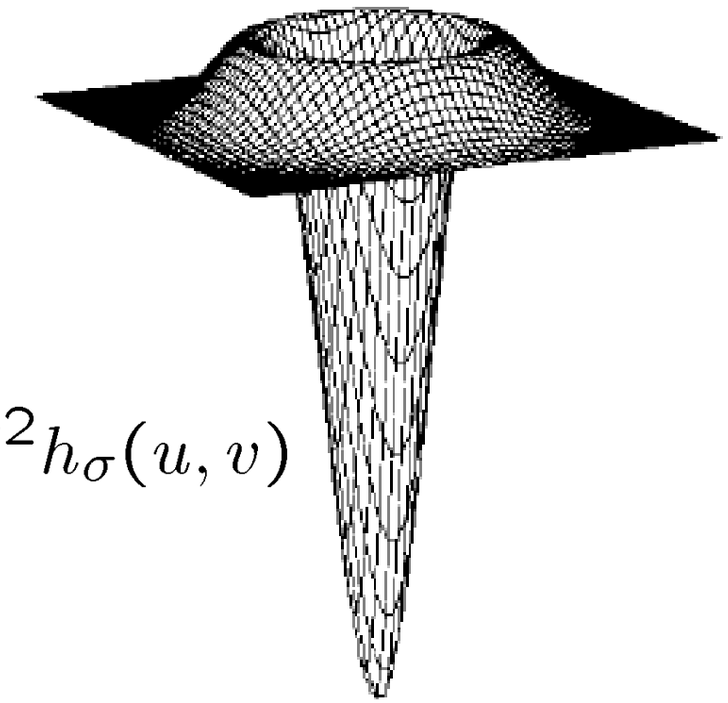
$$h_\sigma(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$



derivative of Gaussian

$$\frac{\partial}{\partial x} h_\sigma(u,v)$$

Laplacian of Gaussian (LoG)



$$\nabla^2 h_\sigma(u,v)$$

∇^2s the Laplacian operator:

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

3x3 LoG:

0	1	0
1	-4	1
0	1	0

Filter properties

- Smoothing
  - All values positive
  - Sum to 1 → constant regions same as input
  - Amount of smoothing proportional to filter size
  - Remove “high-frequency” components → “low-pass” filter

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

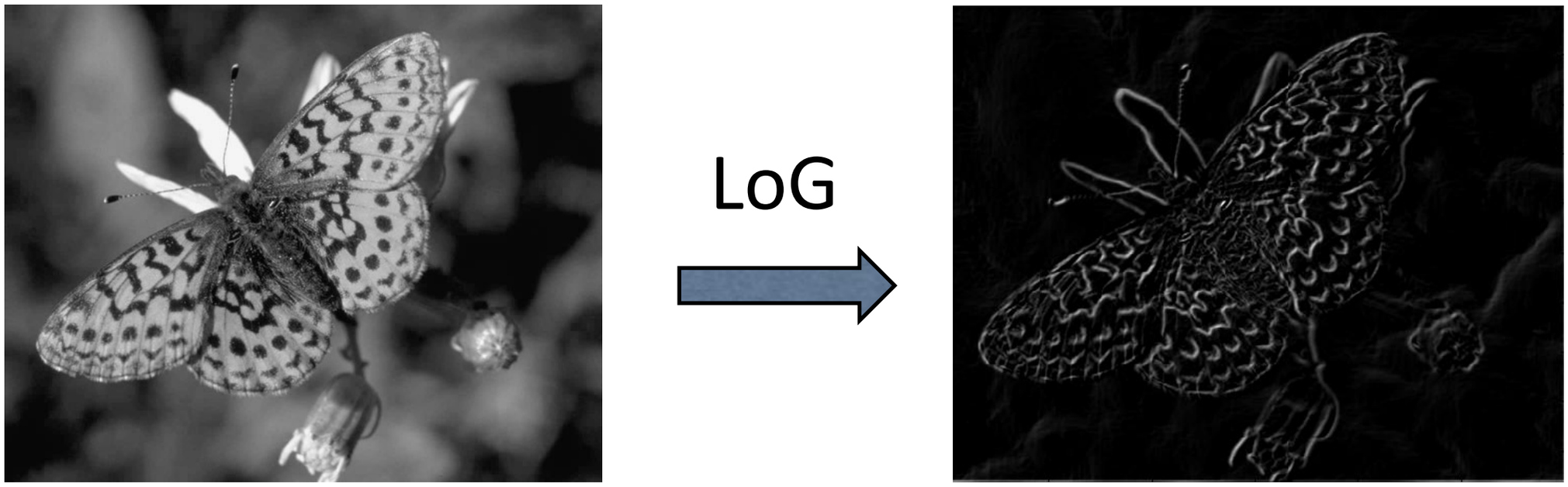
- Edges
  - Opposite signs used to get high response in regions of high contrast
  - Sum to 0 → no response in constant regions
  - High absolute value at points of high contrast
  - Remove “low-frequency” components → “high-pass” filter

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

- Filters act as templates
  - Highest response for regions that “look the most like the filter”
  - Dot product as correlation

Computation of Gradients

- Apply the LoG filter to the image
- Result is an intensity map
  - High values: strong gradients → potentially important edges (high contrast regions)
  - Low values: weak or no gradients → rather homogenenous regions

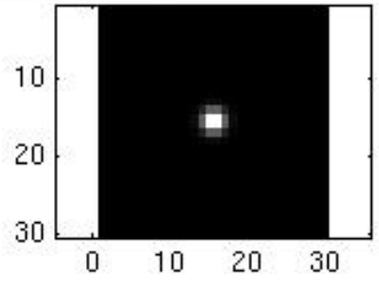


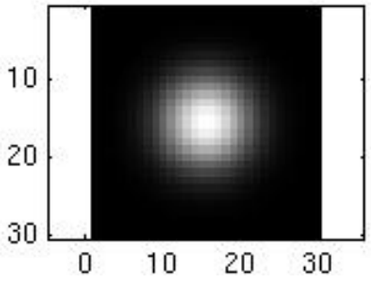
Input image

Gradients

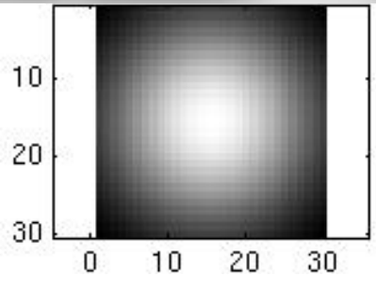
Influence on Amount of Smoothing on Edge Detection (LoG Filter)

- Recall: parameter σ is the “scale” / “width” / “spread” of the Gaussian kernel, and controls the amount of smoothing.

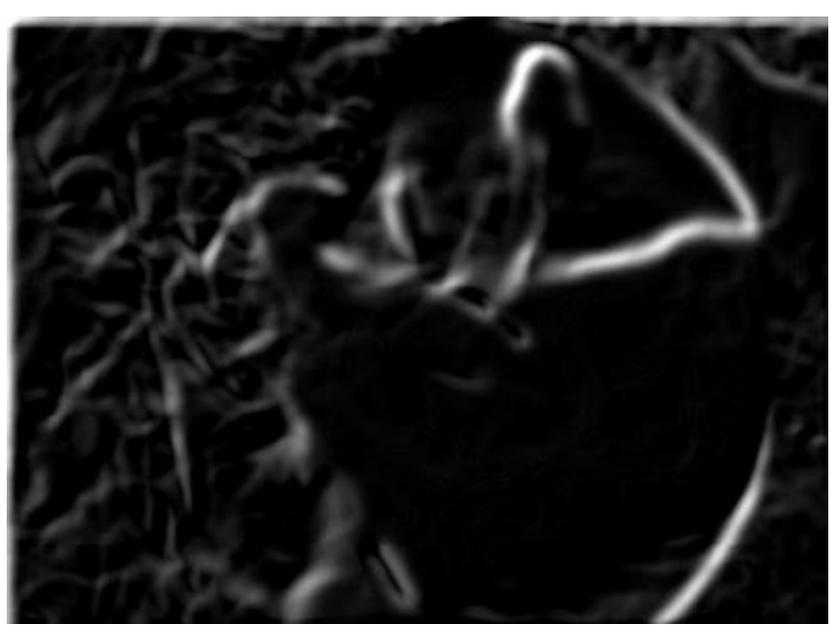




...



Influence on Amount of Smoothing on Edge Detection (LoG Filter)



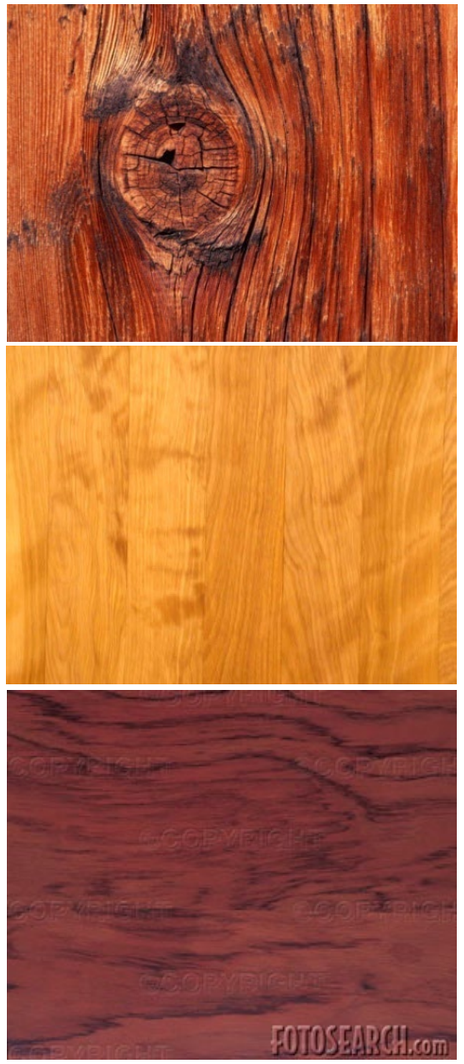
σ = 1 pixel

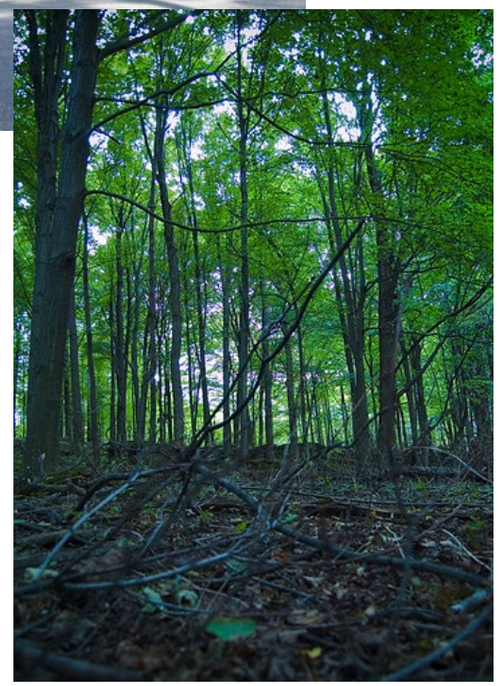
σ = 3 pixels

- The apparent structures differ depending on Gaussian's scale parameter.
- Larger values: larger scale edges detected
- Smaller values: finer features detected

So, what scale to choose?

- It depends what we are looking for…





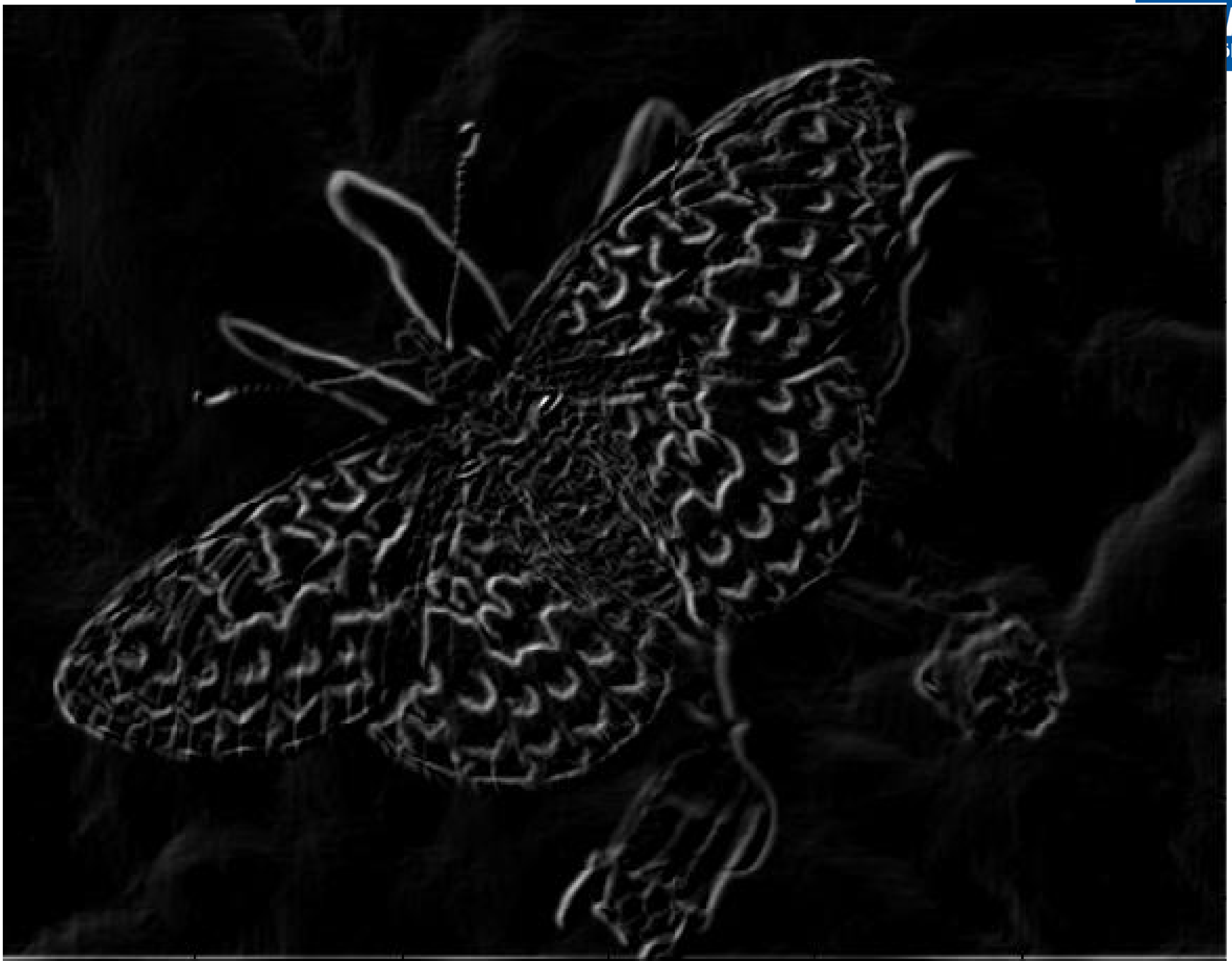
- Too fine of a scale… can’t see the forest for the trees.
- Too coarse of a scale…can’t tell the maple grain from the cherry.

A Simple Edge Detection:

- Choose a threshold value t
- Set any pixels less than t to zero (off)
- Set any pixels greater than or equal to t to one (on)

Original image

Gradient magnitude image



Thresholding gradient with a lower threshold

Thresholding gradient with a higher threshold

Are we happy with that?

Canny edge detector

- Filter image with derivative of Gaussian (e.g. LoG)
- Find magnitude and orientation of gradient
- **Non-maximum suppression**:
  - Thin multi-pixel wide “ridges” down to single pixel width
- Linking and thresholding (**hysteresis**):
  - Define two thresholds: low and high
  - Use the high threshold to start edge curves and the low threshold to continue them
- Python:
  - `edges2 = feature.canny(im, sigma=3)`
  - see: http://scikit-image.org/docs/dev/auto_examples/edges/plot_canny.html

Source: D. Lowe, L. Fei-Fei

The Canny edge detector

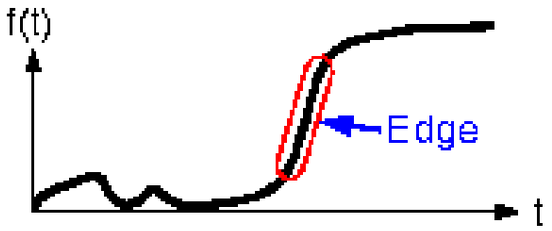
The Canny edge detector

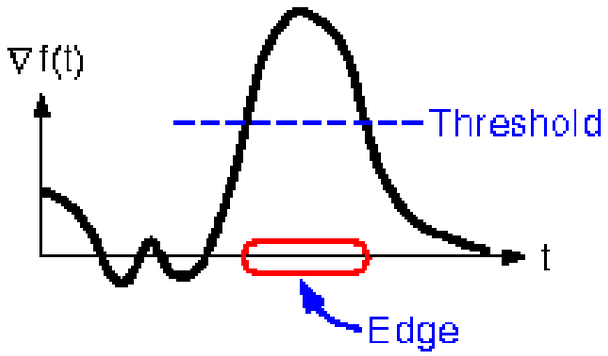
gradient map

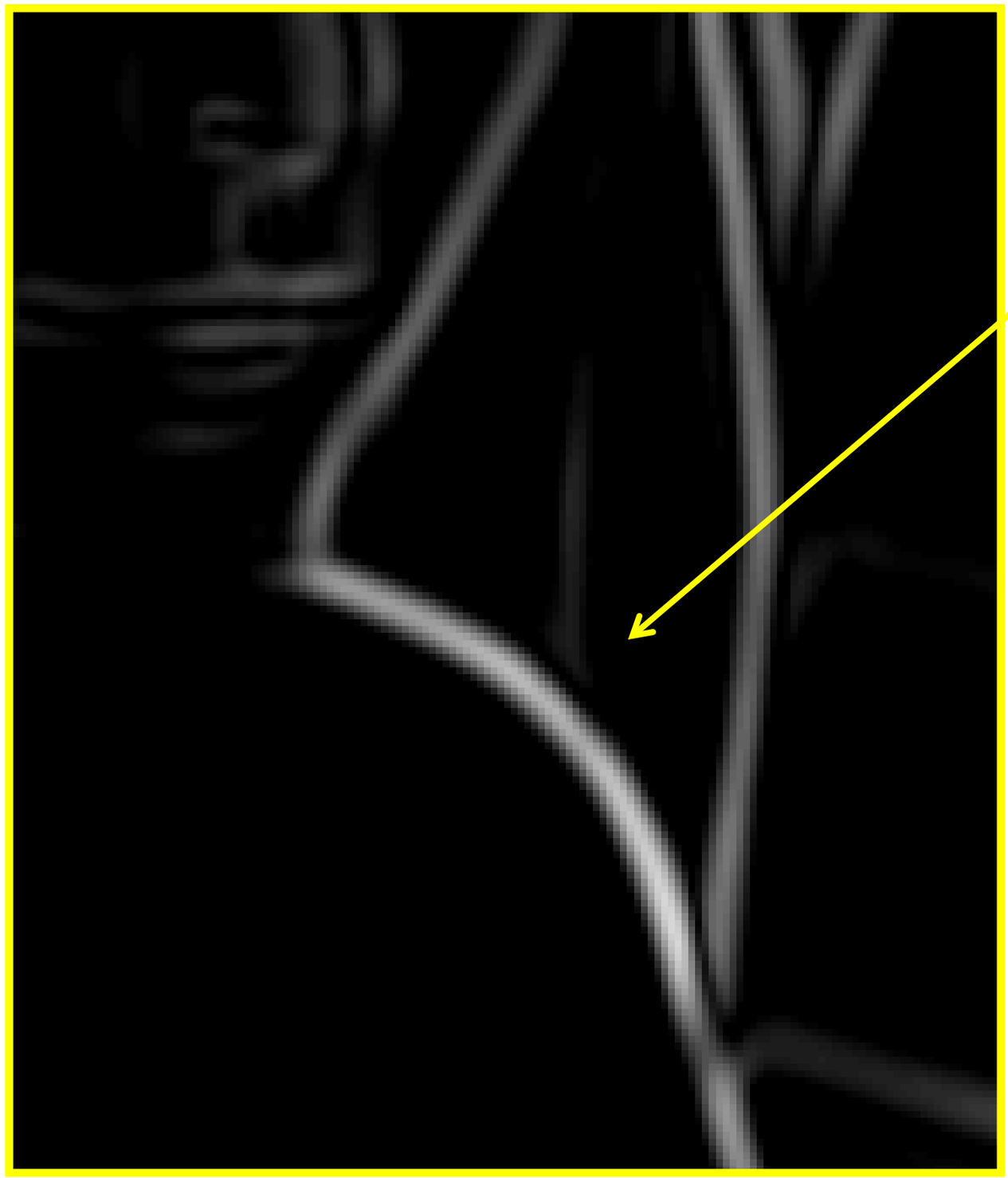
The Canny edge detector

thresholding

The Canny edge detector

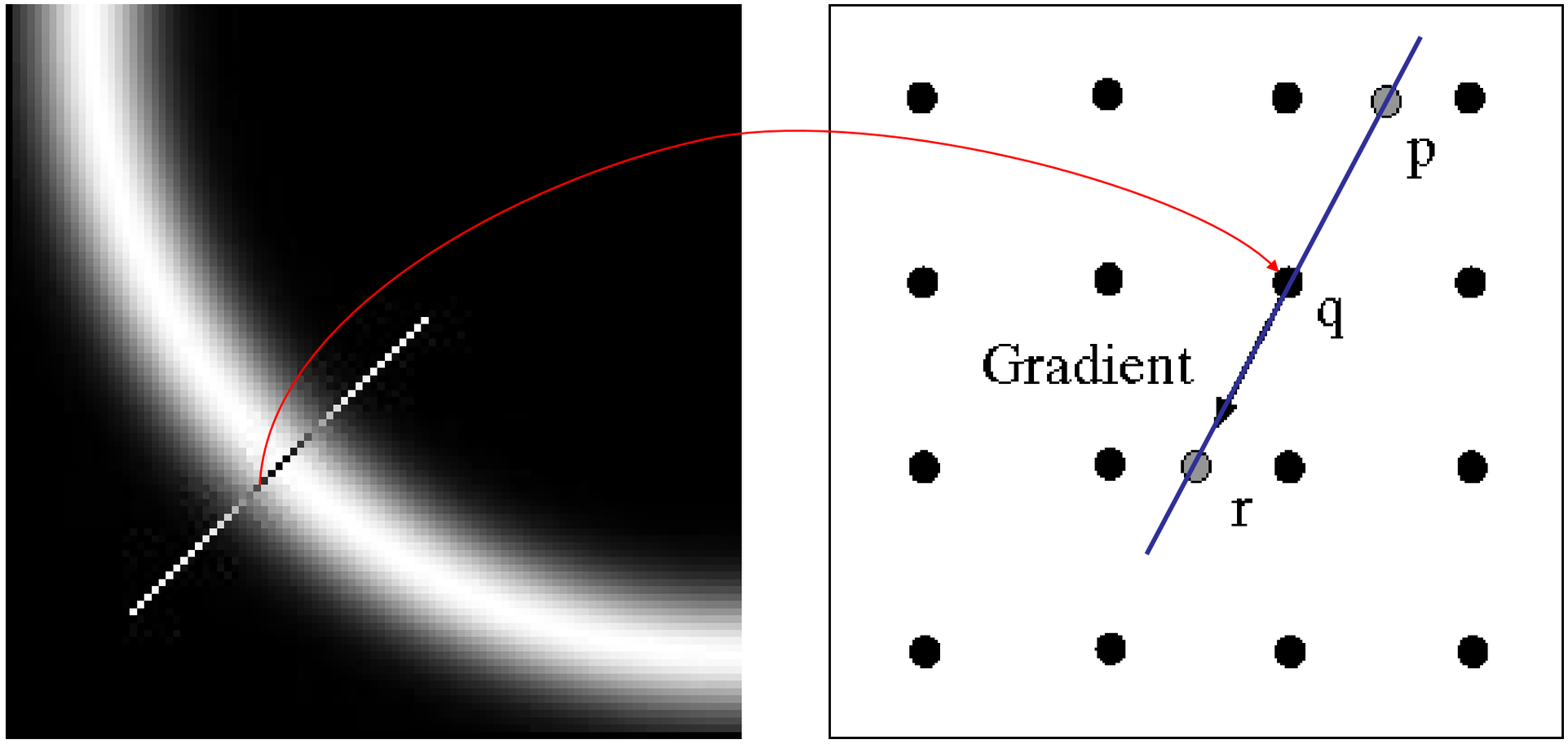






How to turn these thick regions of the gradient into curves?

Non-maximum suppression



- Check if a given pixel q is local maximum along gradient direction, select single max across width of the edge
  - The “gradient line” does not intersect with the pixels! Points p and r are not valid pixels.
  - What are the values or p and r? Estimate via bilinear interpolation
  - If q>r and q>p then declare q is a local maximum

The Canny edge detector

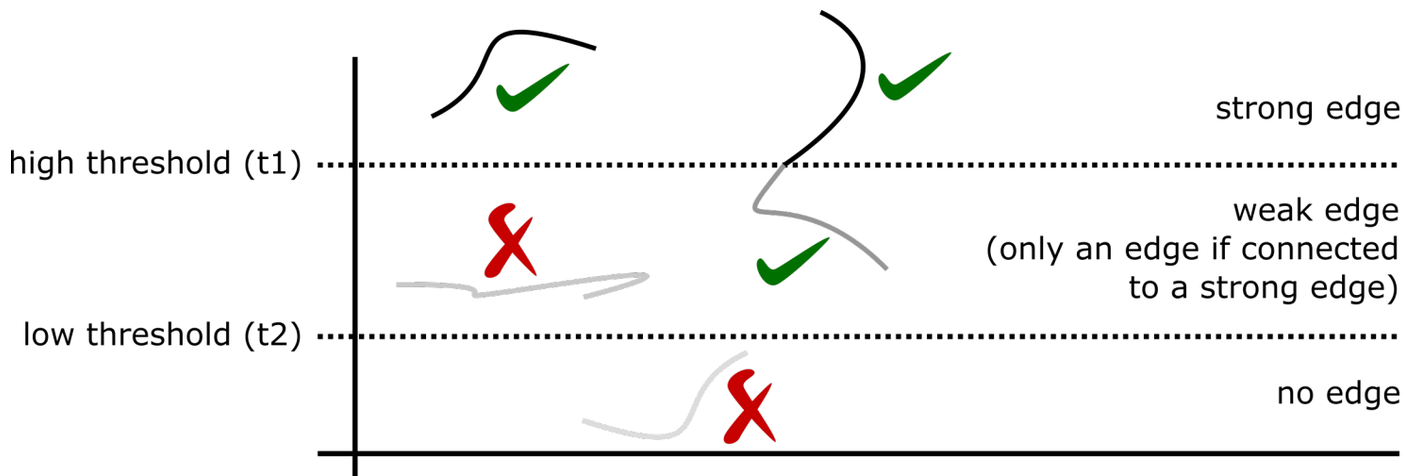
Problem: pixels along this edge didn’t survive the thresholding

thinning

(non-maximum suppression)

Hysteresis thresholding

- Check that maximum value of gradient value is sufficiently large
  - Use “**Hysteresis”:**
    - Define two thresholds t1 and t2, where t1 is larger than t2
    - Look for strong edge pixels which are larger than t1
    - Go along edge until edge strengths falls under t2 → stop edge
- Idea:
  - Weak edges that are not connected to strong edge pixels and considered noise and are removed
  - Weak edges that are connected to strong edge pixels are kept



Hysteresis thresholding

original image

high threshold
(strong edges)

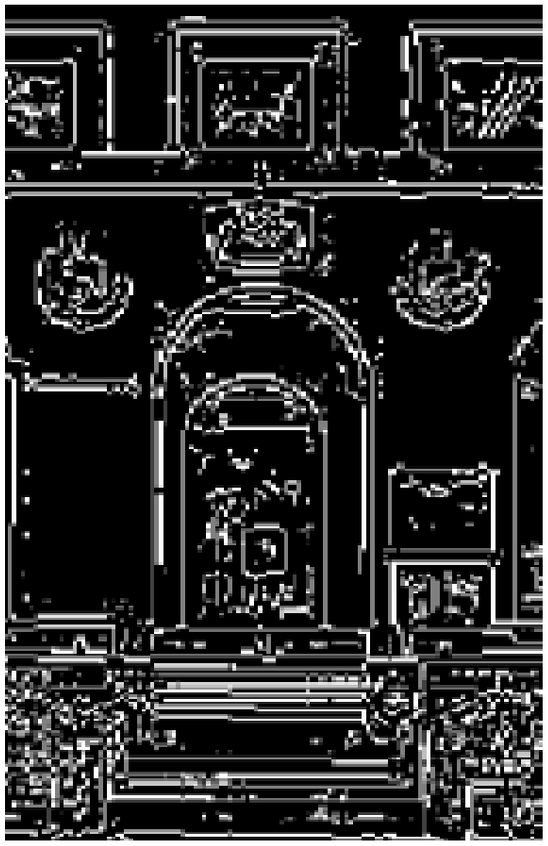
low threshold
(+weak edges)

hysteresis threshold

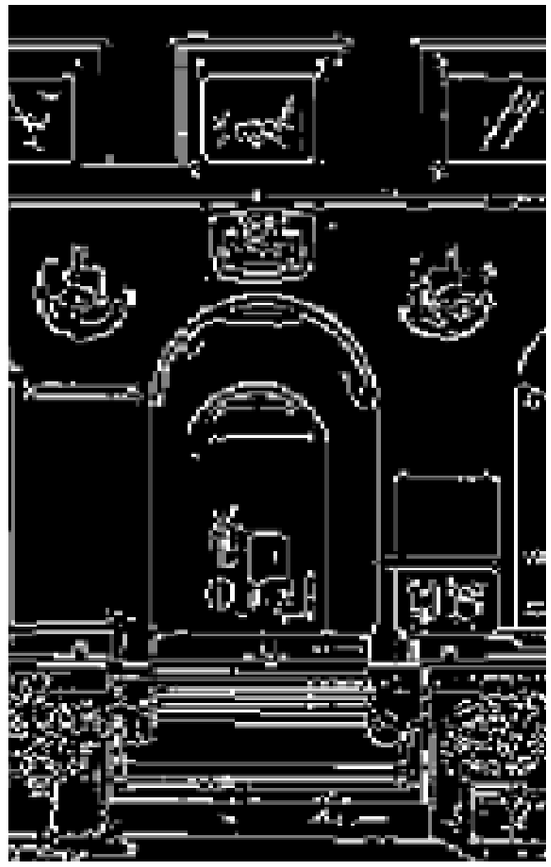
Source: L. Fei-Fei

Hysteresis thresholding

high threshold
(strong edges)



low threshold
(weak edges)



hysteresis threshold

Source: L. Fei-Fei

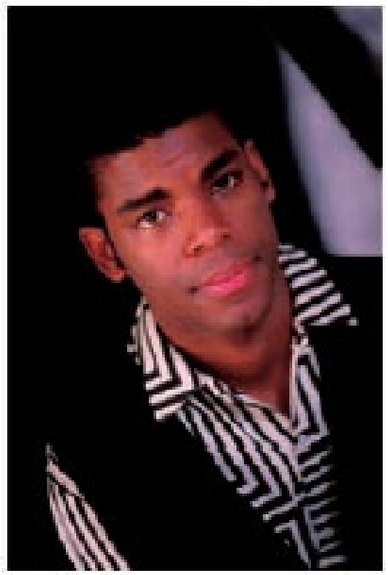
Effect of σ (Gaussian kernel spread/size)

original Canny with $\sigma = 1$ Canny with $\sigma = 2$

The choice of σ depends on desired behavior

- large σ detects large scale edges
- small σ detects fine features

Object boundaries vs. edges

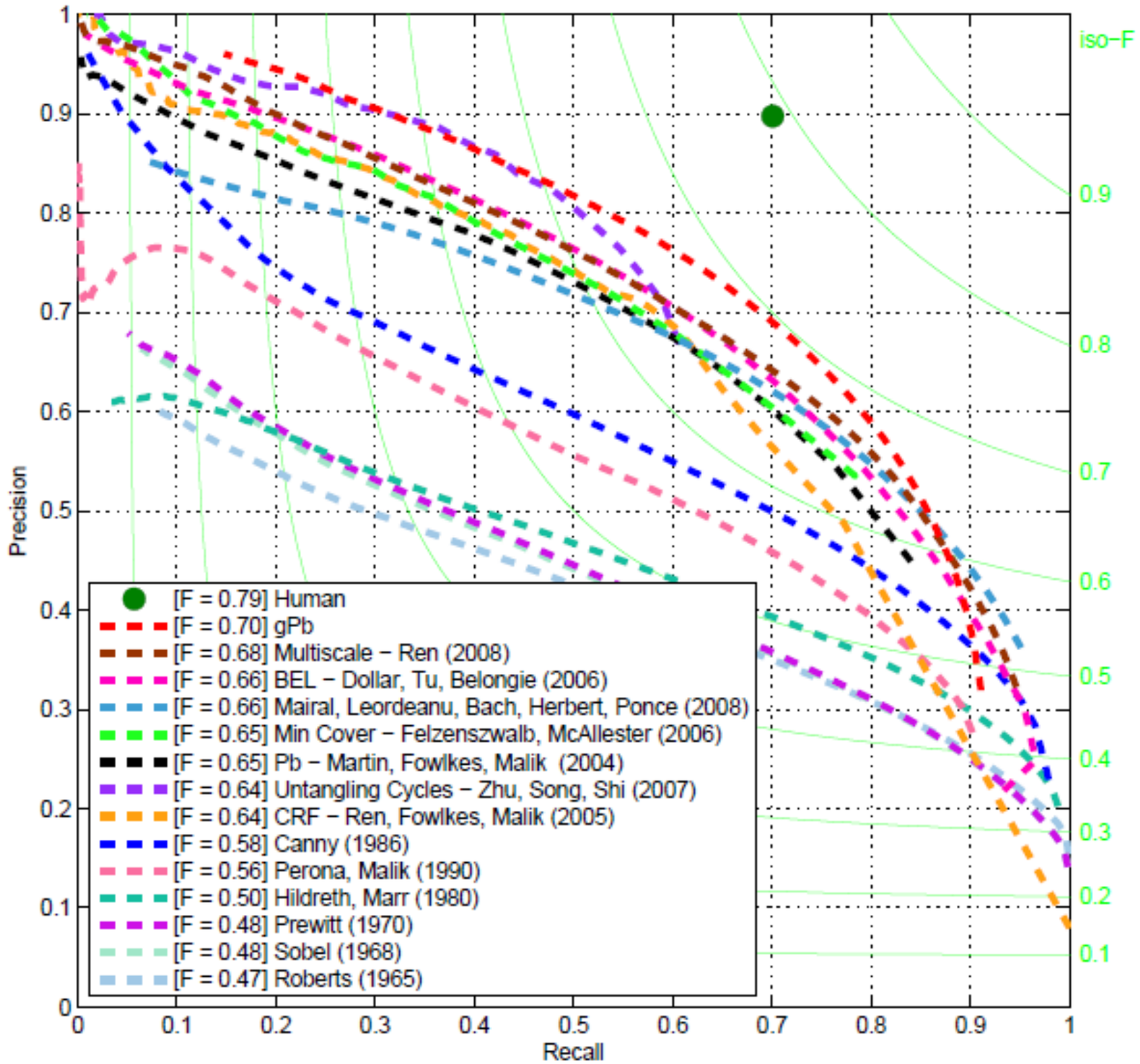


Background

Texture

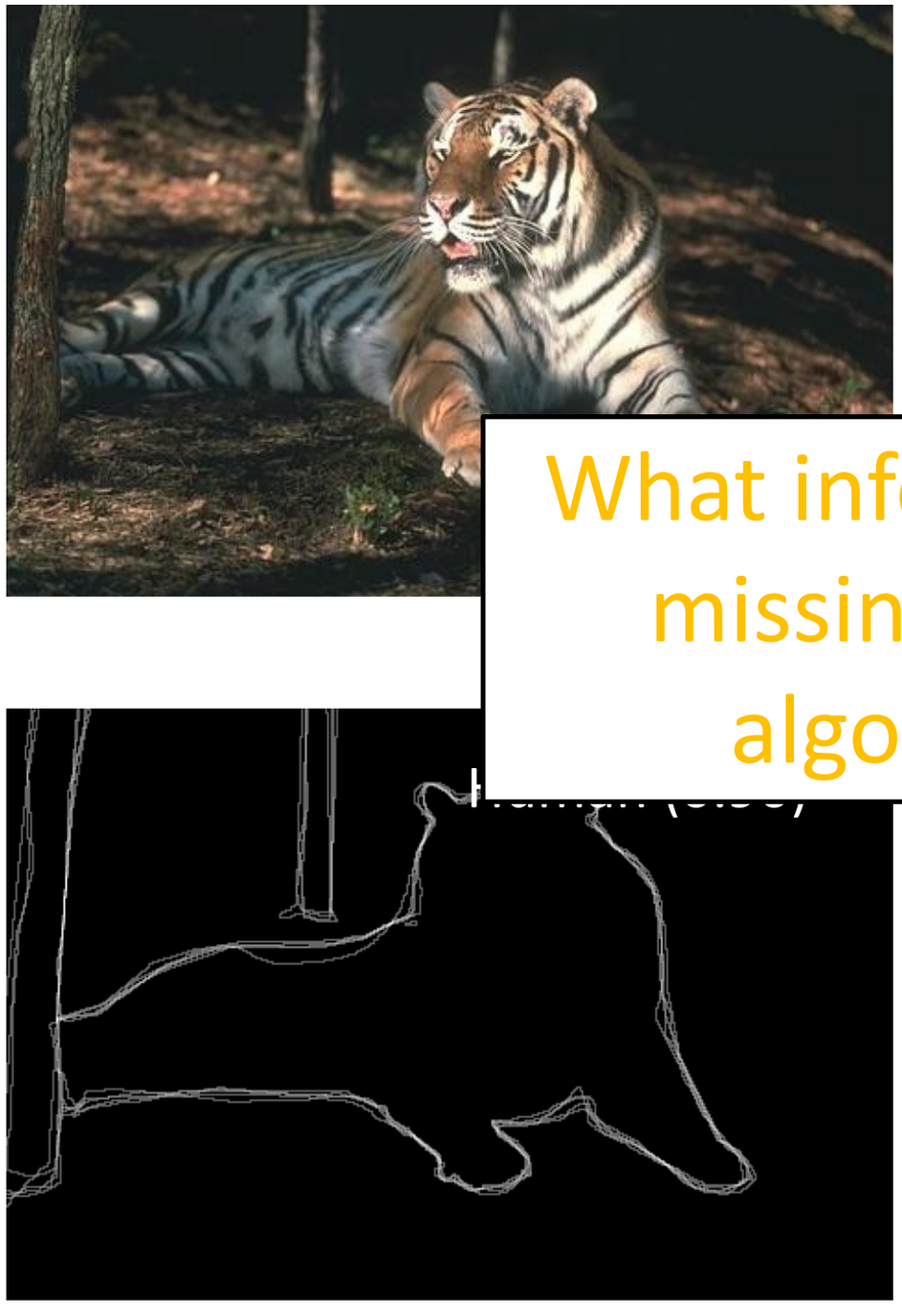
Shadows

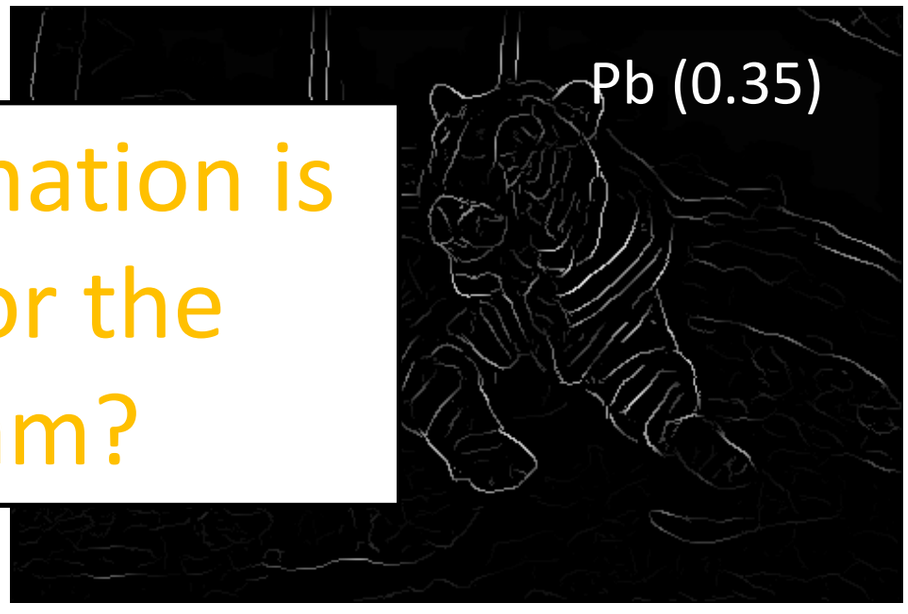
45 years of edge/boundary detection



Source: Arbelaez, Maire, Fowlkes, and Malik. TPAMI 2011

There are limits…





What information is missing for the algorithm?

For more:
http://www.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/bench/html/108082-color.html

Semantic Segmentation (by Facebook Research)

- Segmentation by Deep Networks (SharpMask Algorithm)

https://code.facebook.com/posts/561187904071636/segmenting-and-refining-images-with-sharpmask/, ECCV 2016

Summary on Edge Detection

- Filters allow local image neighborhood to influence our description and features
  - Smoothing to reduce noise
  - Derivatives to locate high contrast areas and gradients
- Filters have highest response on neighborhoods that “look like” it; can be thought of as template matching.
- Edge detection uses the image gradient to find curves, or chains of edges.