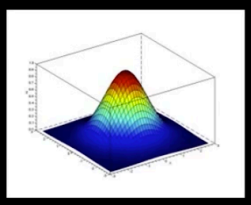
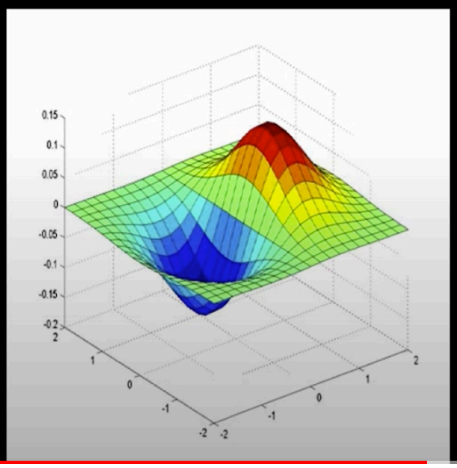
Derivative of Gaussian filter – 2D

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

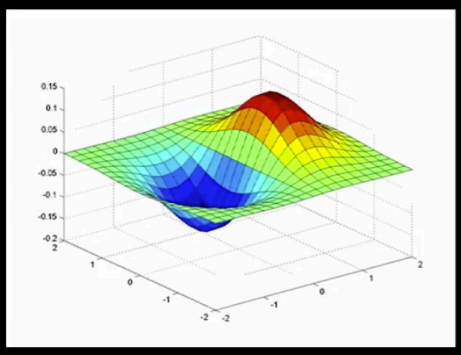


$$\begin{bmatrix} 0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \\ 0.0133 & 0.0596 & 0.0983 & 0.0596 & 0.0133 \\ 0.0219 & 0.0983 & 0.1621 & 0.0983 & 0.0219 \\ 0.0133 & 0.0596 & 0.0983 & 0.0596 & 0.0133 \\ 0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \end{bmatrix} \otimes \begin{bmatrix} -1 & 1 \end{bmatrix} =$$

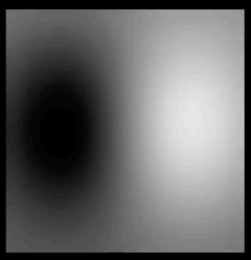


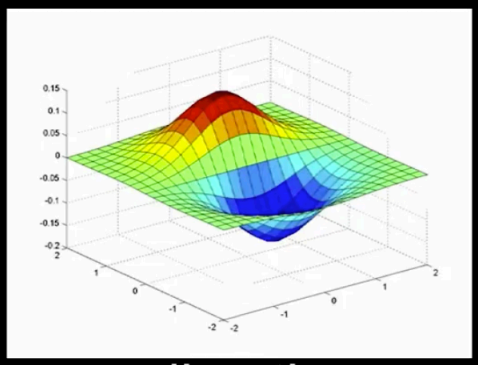
Is this preferable?

Derivative of Gaussian filter

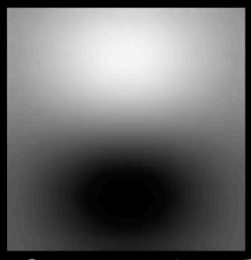


x-direction





y-direction



Correlation or convolution?

And for y it's always a problem!

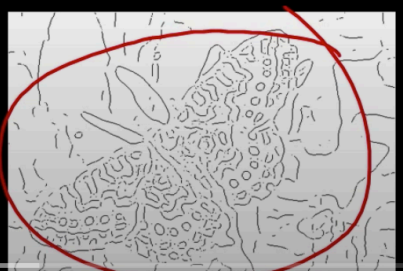
Play (k)

Source: S. Lazebnik

Gradients -> edges

Primary edge detection steps:

1. Smoothing derivatives to suppress noise and compute gradient.
2. Threshold to find regions of "significant" gradient.
3. "Thin" to get localized edge pixels



Canny edge operator

1. Filter image with derivative of Gaussian
2. Find magnitude and orientation of gradient
3. Non-maximum suppression:

   Thin multi-pixel wide “ridges” down to single pixel width

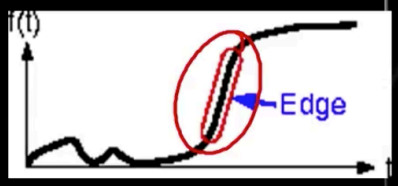
Canny edge operator

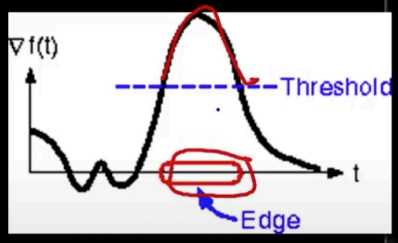
4. Linking and thresholding (hysteresis):
   - Define two thresholds: low and high
   - Use the high threshold to start edge curves and the low threshold to continue them

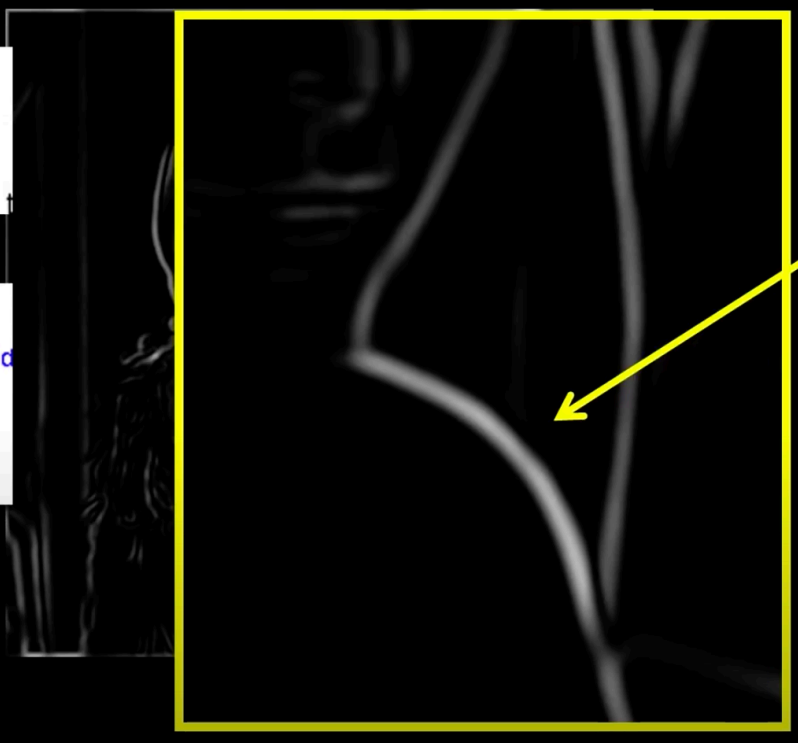
MATLAB: edge(image, 'canny');

>>doc edge *(or help edge* if doc is not supported)

The Canny edge detector

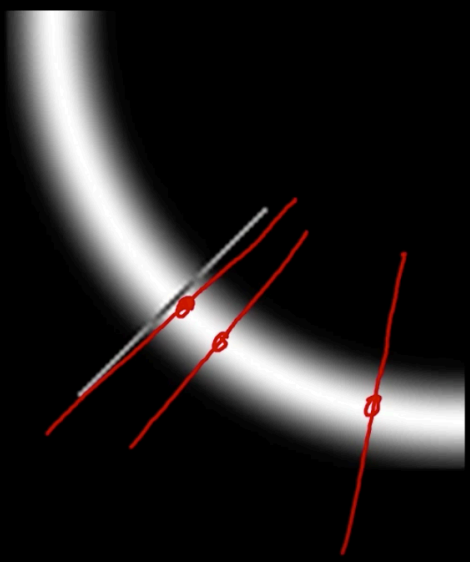


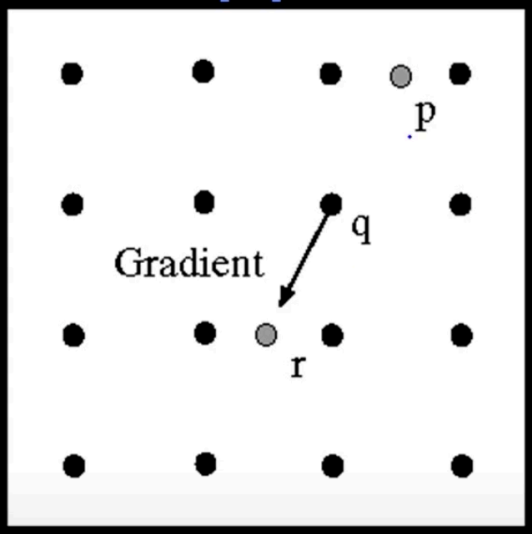




How to turn these thick regions of the gradient into curves?

Canny: Non-maximal suppression





Check if pixel is local maximum along gradient direction

can require checking interpolated pixels p and r

The Canny edge detector

thinning
(non-maximum suppression)

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

Canny threshold hysteresis

1. Apply a high threshold to detect strong edge pixels.
2. Link those strong edge pixels to form strong edges.
3. Apply a low threshold to find weak but plausible edge pixels.
4. Extend the strong edges to follow weak edge pixels.

Result of Canny

Effect of σ (Gaussian kernel spread/size)

original

Canny with $\sigma = 1$

Canny with $\sigma = 2$

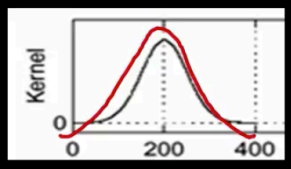
- Large σ detects large scale edges
- Small σ detects fine features

The choice of σ depends on desired behavior

Recall 1D 2nd derivative of Gaussian

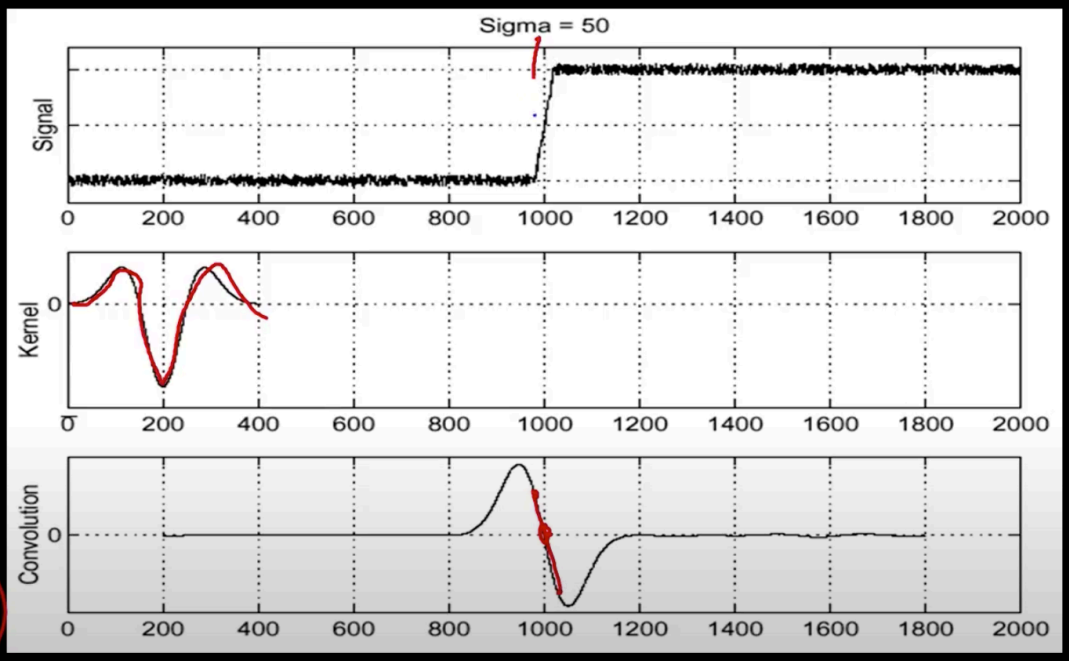
f

h



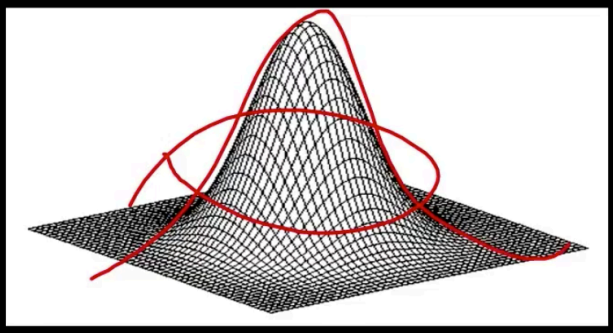
$$\frac{\partial^2}{\partial x^2} h$$

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

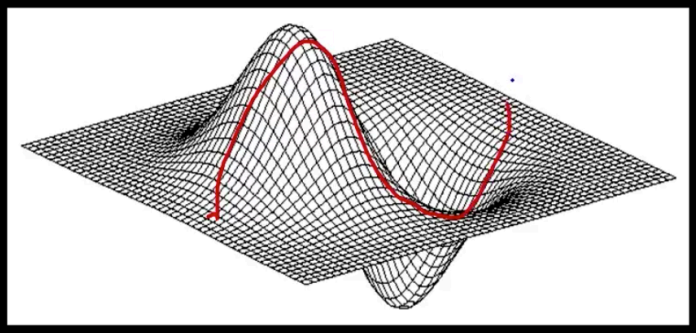


Zero-crossings of bottom graph are edges

Single 2D edge detection filter



Gaussian



derivative of Gaussian

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

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

Single 2D edge detection filter

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

∇^2 is the **Laplacian** operator,

And the zero-crossings are the edges.