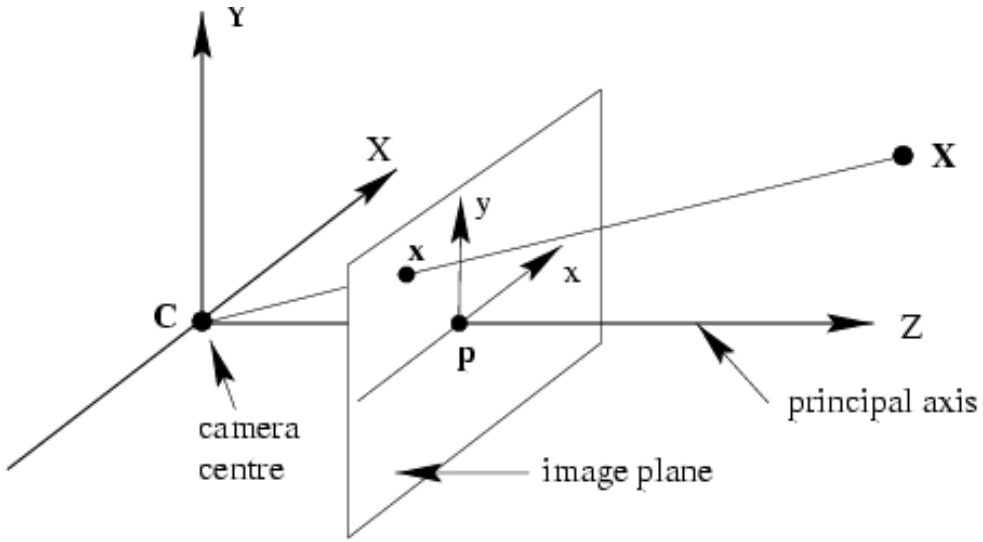
CS664 Lecture #20: Epipolar geometry, edge detection

Estimating epipolar geometry

- Assume for the moment that all intrinsic parameters are known
- *Essential Matrix* E captures the geometry
  - Mapping from a point to its epipolar line
    - i.e., tells you where that point might be!
  - For a point x with corresponding point x' on the other image, $x^T E x' = 0$
    - Using homogeneous coordinates
- When intrinsic parameters are unknown, we have the *Fundamental Matrix* F

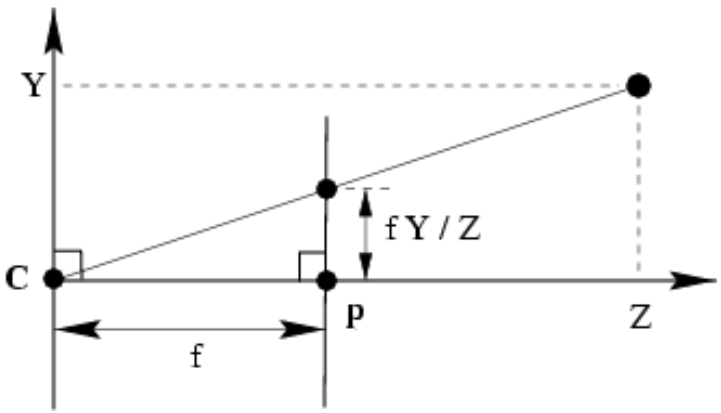
Homogeneous coordinates

- Identify a point in the image plane with ray passing through that point (pixel)
  - $(x,y) \equiv (\alpha\, x, \alpha\, y, \alpha)$ for non-zero α
  - $(X,Y,Z) \equiv (X/Z, Y/Z, 1)$ for non-zero Z



Advantages

- Many non-linear operations become linear in homogeneous coordinates
  - Example: (X,Y,Z) projects to $(fX/Z, fY/Z)$



$$\begin{bmatrix} fX \\ fY \\ Z \end{bmatrix} = \begin{bmatrix} f & & & 0 \\ & f & & 0 \\ & & f & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

2D point

3x4 camera projection

3D point

Camera projection matrix

Scale factors m_x, m_y

Optical axis coordinates (p_x, p_y)

$$\begin{bmatrix} m_x f & & p_x & 0 \\ & m_y f & p_y & 0 \\ & & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Estimating the essential matrix

- Estimate E from 5 or more correspondences

$$\hat{E} = \arg\min \sum_{j=1}^{k \geq 5} x_j^T E \mathbf{x}'_j$$

  - More correspondences make this easier
  - 8 point algorithm to find F
    - Longuet-Higgins, 1981
- Can use RANSAC or Least Median Squares
  - RANSAC can tolerate more outliers
  - LMedS is parameter-free

Detecting edges or features

- To perform various sparse image operations we need features
  - Operations: registration, recognition, etc.
  - Features: edges, corners, “interest points”
- Feature detector desiderata
  - Efficient
  - Sparse, but not too sparse
  - Invariant!
    - Affine changes in intensity (easy)
    - Changes in point of view (hard)

Edge detection

- Classical problem in vision
  - Good handout from Dan Huttenlocher now on course web page
- Easiest to think about (draw) if we try to find edges in a 1D signal
- Two obvious approaches
  - Large first derivative
  - Zero second derivative (inflection point)
- Two classes of edge detectors
  - Canny style, or Marr-Hildreth style

Detecting Edges

- Seek sudden changes in intensity
  - Various derivatives of image
- Idealized continuous image $I(x,y)$
- Gradient (first derivative), vector valued

$$\nabla I = (\partial I/\partial x, \partial I/\partial y)$$

- Squared gradient magnitude

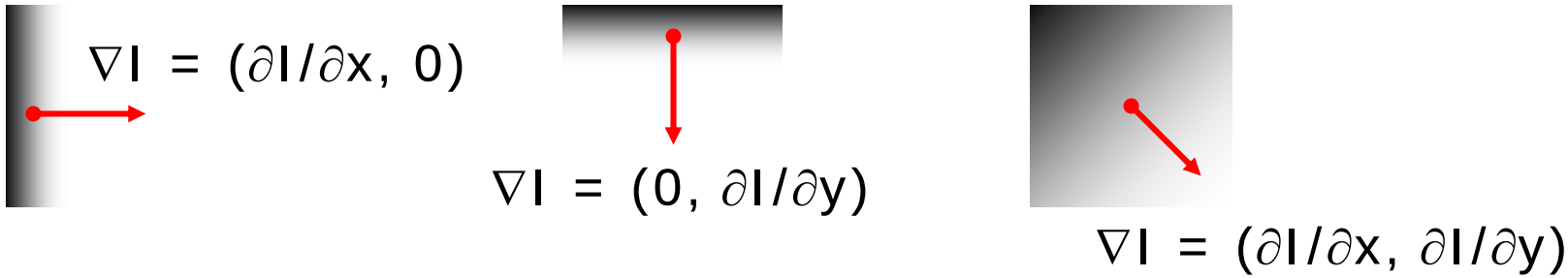
$$\|\nabla I\|^2 = (\partial I/\partial x)^2 + (\partial I/\partial y)^2$$

  - Avoid computing square root
- Laplacian (second derivative)

$$\nabla^2 I = \partial^2 I/\partial x^2 + \partial^2 I/\partial y^2$$

The Gradient

- Direction of most rapid change



- Gradient direction is atan(∂I/∂y,∂I/∂x)
  - Normal to edge
- Strength of edge given by grad magnitude
  - Often use squared magnitude to avoid computing square roots

Finite Differences

- Images are digitized
  - Idealized continuous underlying function I(x,y) realized as discrete values on a grid I[u,v]
- Approximations to derivatives (1D)

  $dF/dx \approx F[u+1] - F[u]$

  $d^2F/dx^2 \approx F[u-1] - 2F[u] + F[u+1]$

1	0	1	0	10	11	11	0	1
-1	1	-1	10	1	0	-11	1	0
-2	2	-2	11	-9	-1	-11	12	-2

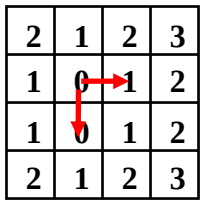
dF: edge at extremum

d^2F: edge at zero crossing

- Second derivative symmetric about edge

Discrete Gradient

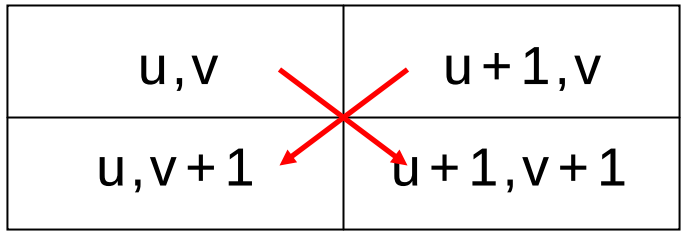
- Partial derivatives are estimated at boundaries between adjacent pixels
  - E.g., pixel and next one in x,y directions
- Yields estimates at different points in each direction if use x,y directions



- Generally use 45° directions to solve this
  - Magnitude fine, but gradient orientation needs to be rotated to correspond to axes

Estimating Discrete Gradient

- Gradient at u,v with 45° axes
  - Down-right: $\partial I/\partial x' \approx I[u+1,v+1]-I[u,v]$
  - Down-left: $\partial I/\partial y' \approx I[u,v+1]-I[u+1,v]$
- Handle image border, e.g., no change



2	1	1	2
7	6	7	6
1	6	7	7
2	2	7	6

I

4	6	5	0
-1	1	0	0
1	1	-1	0
0	0	0	0

$\partial I/\partial x'$

0	6	5	5
0	-5	-1	1
0	-4	-5	0
0	0	0	0

$\partial I/\partial y'$

16	72	50	25
1	26	1	1
1	17	26	0
0	0	0	0

$\|\nabla I\|^2$

Discrete Laplacian

- Laplacian at u,v

  $\partial^2 I/\partial x^2 = I[u-1,v]-2I[u,v]+I[u+1,v]$

  $\partial^2 I/\partial y^2 = I[u,v-1]-2I[u,v]+I[u,v+1]$

  $\nabla^2 I$ is sum of directional second derivatives:
  $I[u-1,v]+I[u+1,v]+I[u,v-1]+I[u,v+1]-4I[u,v]$

- Can view as 3x3 mask or stencil
  - Value at u,v given by sum of product with I
- Grid yields poor rotational symmetry
  - Weighted sum of two masks

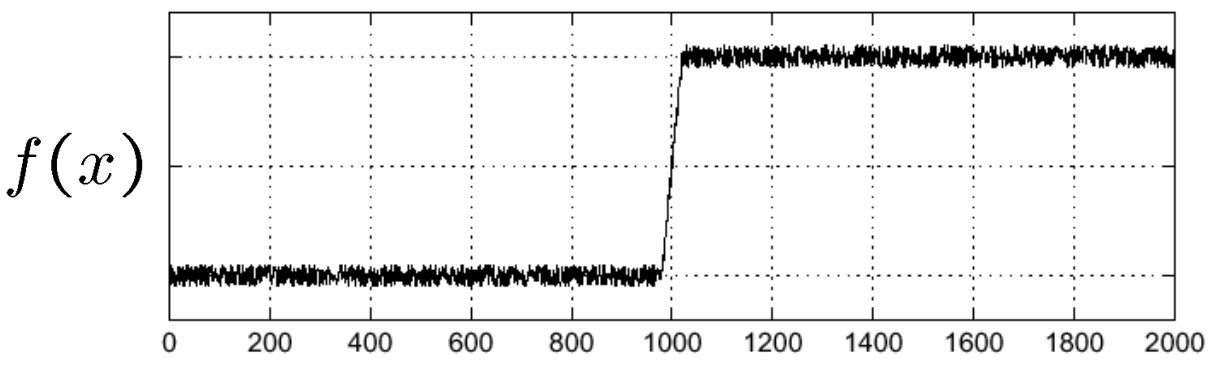
	1	
1	-4	1
	1	

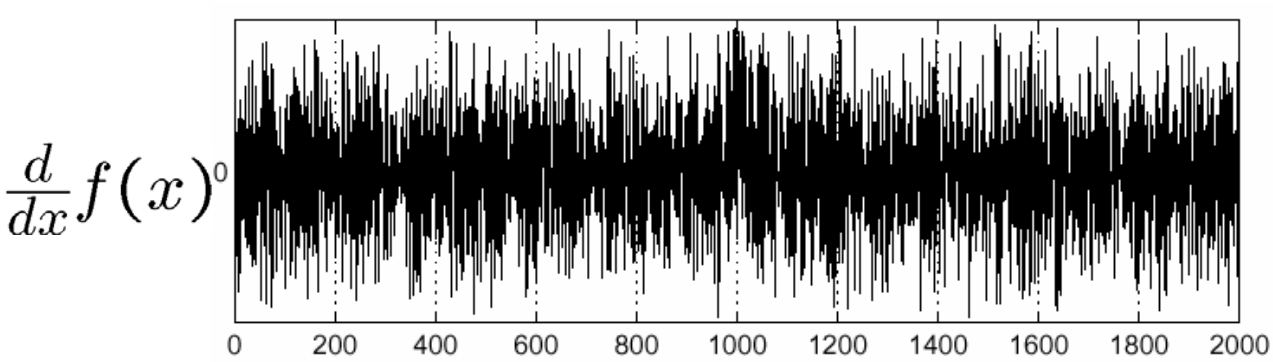
1		1
	-4	
1		1

1	4	1
4	-20	4
1	4	1

Problems With Local Detectors

- 1D example illustrates effect of noise (variation) on local measures





Regions of Support

- Desirable to have edge detectors that operate over interval or region
- Low pass filtering of an image
  - Combining certain neighboring pixel values to produce “less variable” image
  - Often referred to as “smoothing” or as “blurring” the image
- Simple idea: mean filter – average values over w by h neighborhood

$$M[u,v] = (1/wh) \sum_i \sum_j F[u+i-(w-1)/2,\ v+j-(h-1)/2]$$

3x3 Mean Filter Example

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

?	?	?	?	?	?	?	?	?	?
?	0	10	20	30	30	30	20	10	?
?	0	20	40	60	60	60	40	20	?
?	0	30	60	90	90	90	60	30	?
?	0	30	50	80	80	90	60	30	?
?	0	30	50	80	80	90	60	30	?
?	0	20	30	50	50	60	40	20	?
?	10	20	30	30	30	30	20	10	?
?	10	10	10	0	0	0	0	0	?
?	?	?	?	?	?	?	?	?	?

$$M[u,v] = (1/9) \Sigma_i \Sigma_j F[u+i-1, v+j-1]$$

Border Pixels

- As usual with image operations the border cases need to be handled somehow
  - Produce smaller image by summing only when entire w by h window fits inside image
  - Sum only value inside image but produce full size image
    - In effect summing zeroes outside image
  - Assume value outside image some non-zero value
    - E.g., reflected copy of the image
- No right answer, reflection often least bad

Weighted Average Filter

- Sum of product with weights H

$$G[u,v] = \Sigma_i \Sigma_j H[i,j]F[u+i-(w-1)/2, v+j-(h-1)/2]$$

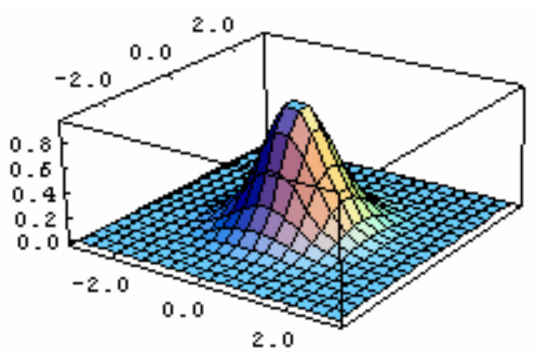
- Mean filter simply has H[I,j]=1/wh
  - Uniform weighting
- Note that entries of H should sum to 1
  - Otherwise performs overall scaling of the image
    - Consider 3x3 mask of 1's instead of 1/9's
- When averaging generally give central pixel most weight

Gaussian Filter

- Gaussian in two-dimensions

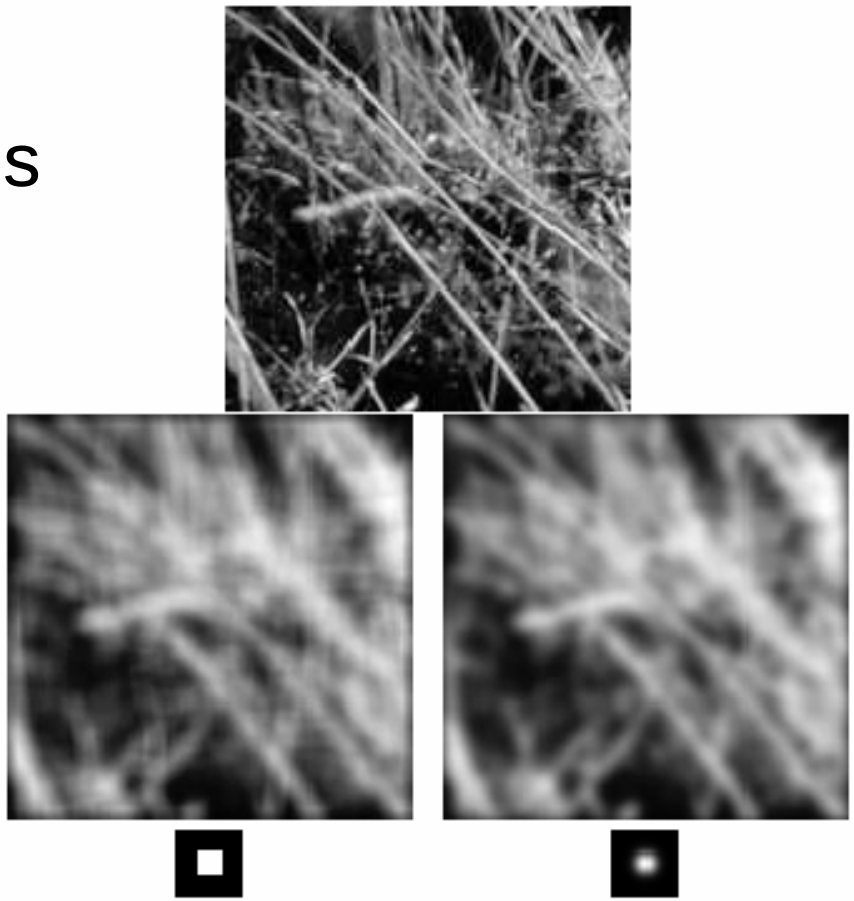
$$h(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{\sigma^2}}$$

- Weights center more
- Falls off smoothly
- Integrates to 1
- Larger σ produces more equal weights (blurs more)
- Normal distribution



Gaussian Versus Mean Filter

- Mean filter blurs but sharp changes remain as well
  - "Blocky"
- Gaussian not blocky looking
- Same area masks
  - But Gaussian small at borders



Convolution

- Same as weighted average if H symmetric

  $G[u,v] = \sum_i \sum_j H[i,j]F[u-i+(w-1)/2, v-j+(h-1)/2]$

  – Written as $G = H \star F$

    • Notation not always consistent

- Convolution has nice properties

  – Commutative: $A \star B = B \star A$

  – Associative: $A \star (B \star C) = (A \star B) \star C$

  – Distributive: $A \star (B + C) = (A \star B) + (A \star C)$

Back to Edges: Derivatives

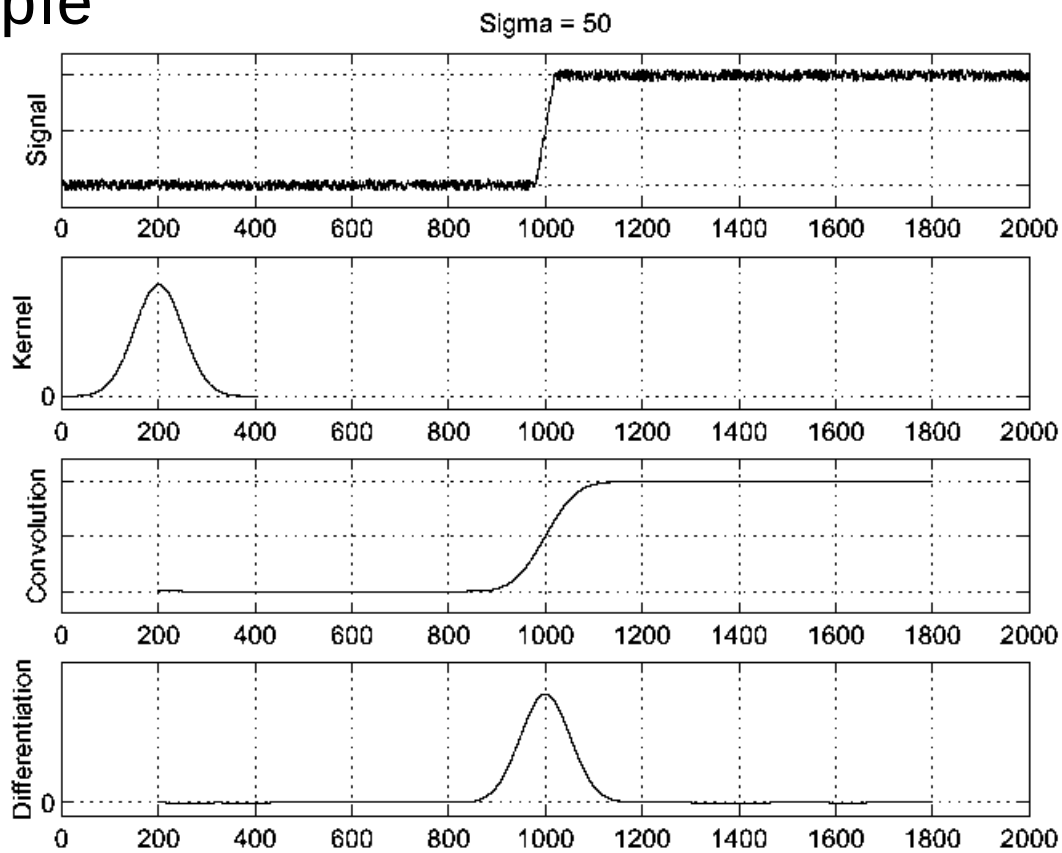
- Smooth and then take derivative
  - 1D example

f

h

$h \star f$

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



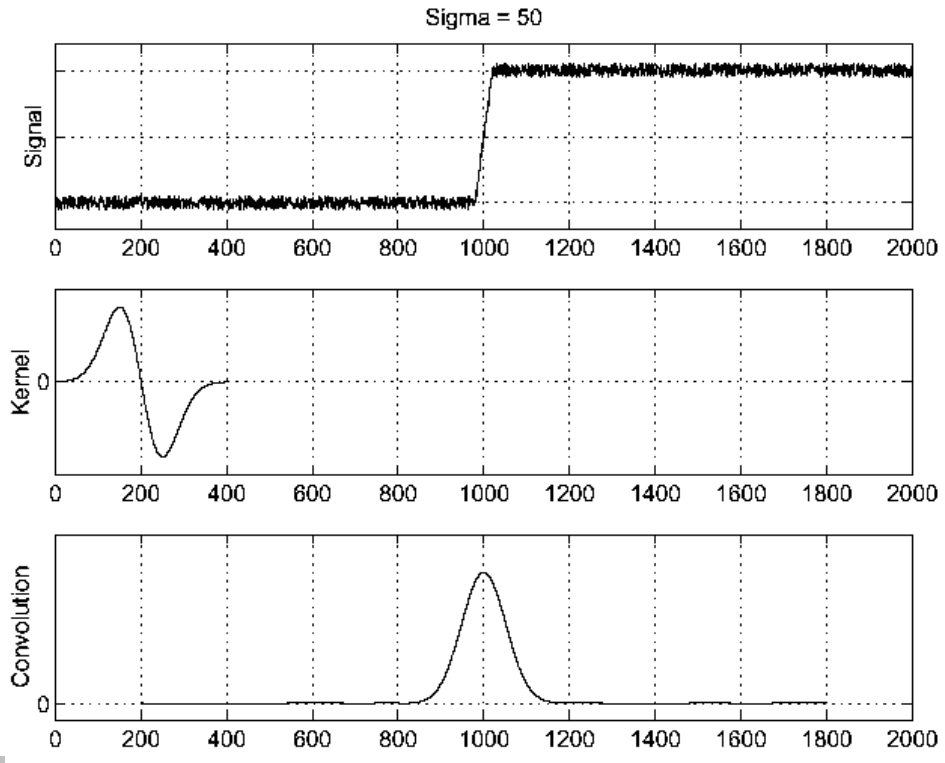
Derivatives and Convolutions

- Another useful identity for convolution is d/dx(A∗B)= (d/dx A)∗B = A∗(d/dx B)
  - Use to skip one step in edge detection

f

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

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



Derivatives Using Convolution

- When smoothing all weights of mask h are positive
  - Sum to 1
  - Maximum weight at center of mask
- Weights do not have to all be positive
  - Negative weights compute differences (derivatives)
  - E.g., Laplacian h =

1	4	1
4	-20	4
1	4	1

  - $h \star f = \nabla^2 f$

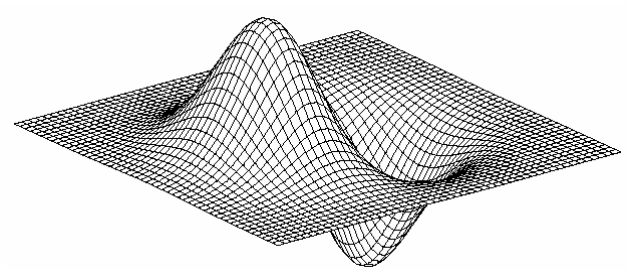
Linear Operators

- Linear shift invariant (LSI) system
  - Given a "black box" h: $f \rightarrow \boxed{h} \rightarrow g$
  - Linearity: $af_1 + bf_2 \rightarrow \boxed{h} \rightarrow ag_1 + bg_2$
  - Shift invariance: $f(x-u) \rightarrow \boxed{h} \rightarrow g(x-u)$
- Convolution with arbitrary h equivalent to these properties
  - Not hard to show this
- Linearity is "simple to understand" but real world not always linear
  - E.g., saturation effects

Area of Support for 2D Operators

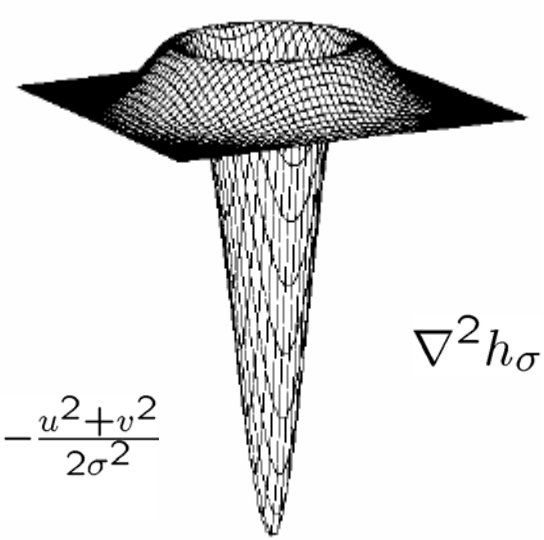
- Directional first derivatives and second derivative (Laplacian) of Gaussian
  - Sigma controls scale, larger yields fewer edges

Derivative of Gaussian



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

Laplacian of Gaussian



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

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

Gradient Magnitude

- Also use smoothed image

$$\|\nabla(I * h_\sigma)\| = ((\partial(I * h_\sigma)/\partial x)^2 + (\partial(I * h_\sigma)/\partial y)^2)^{.5}$$

Edge Detection by Subtraction

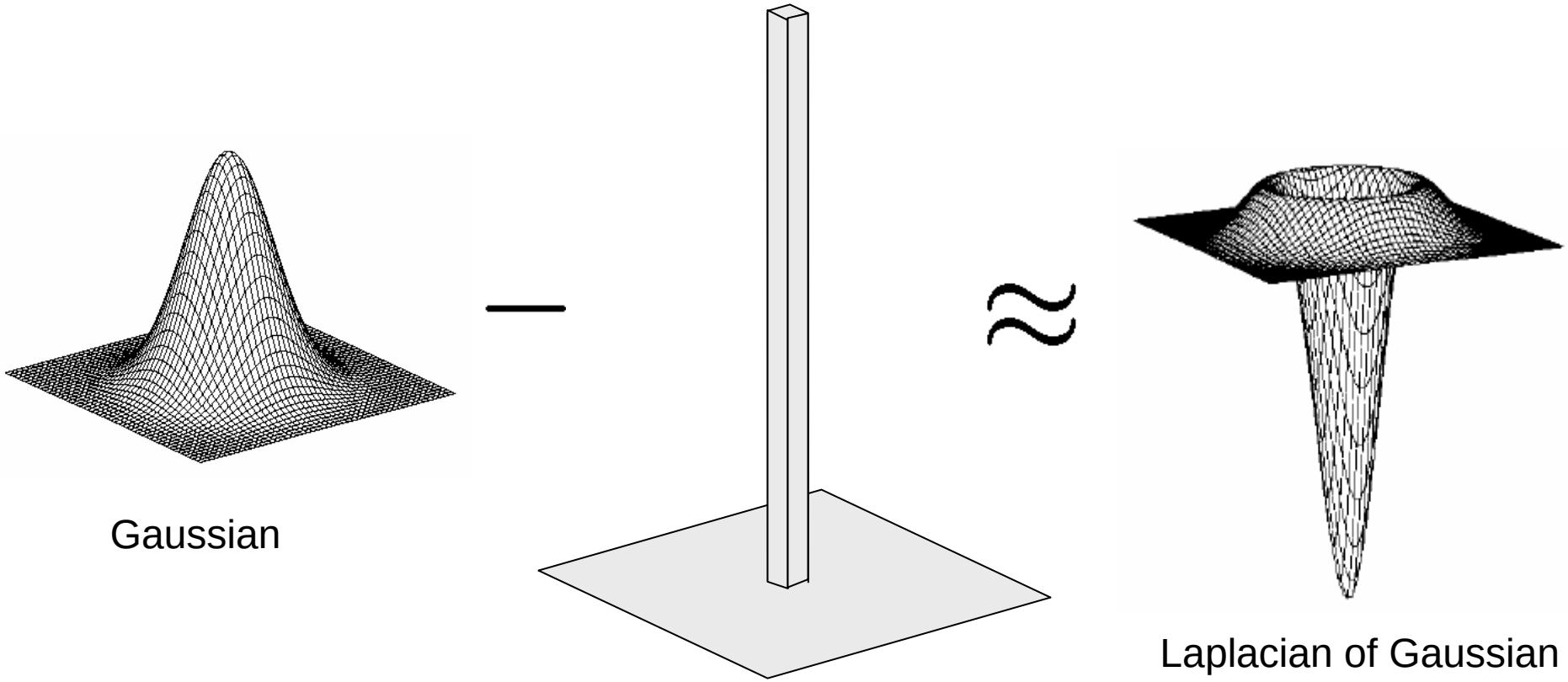
- Difference of image and smoothed version

Original

Smoothed

Difference
(brightened)

What Does This Do?



- More generally $(I \star h_{\sigma 1}) - (I \star h_{\sigma 2}) \approx \nabla^2 I (I \star h_{\sigma 3})$