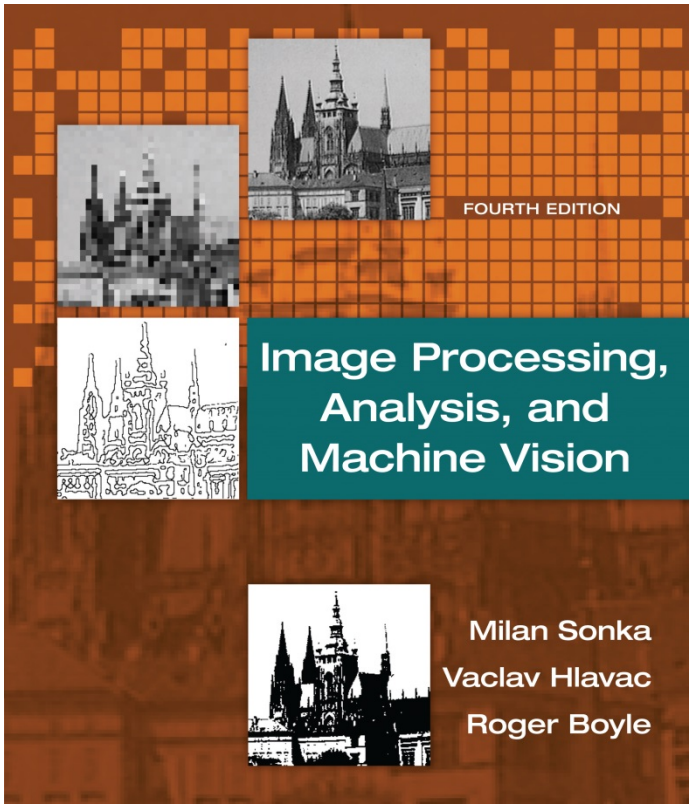


Chapter 5.3.2 – 5.3.5 Convolution, Image Filtering

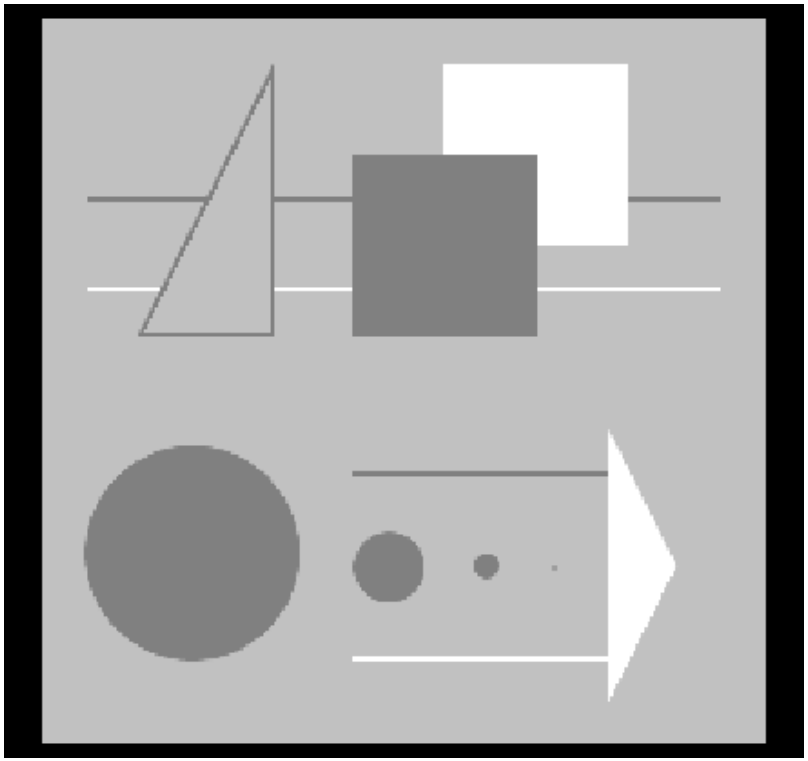
Sources:

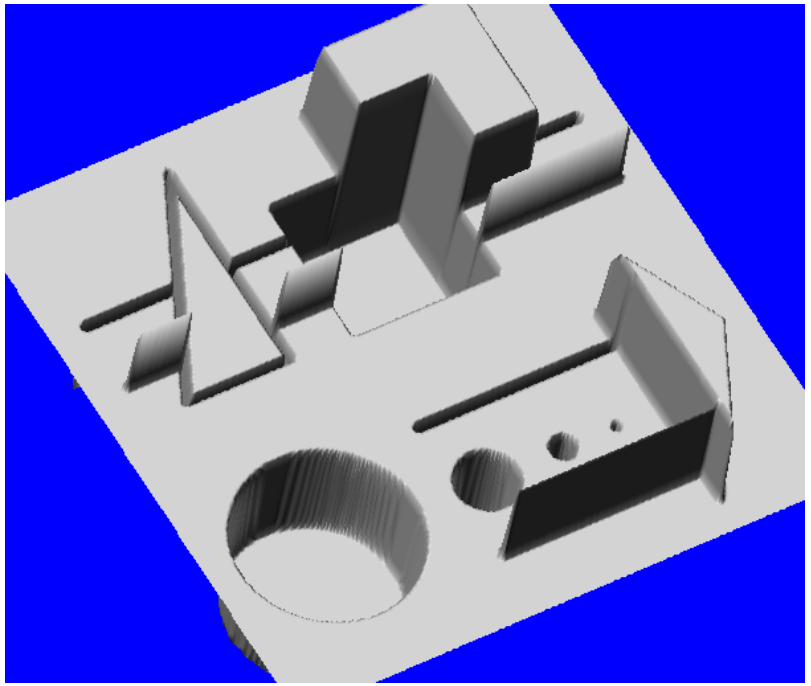
- Sonka Textbook
- Gonzalez/Woods DIP textbook

Overview

- Linear filtering
  - Edge Detection
  - Derivatives
- Template matching equivalent to cross-correlation
- Canny edge detection

Digital Images: Boundaries are "Lines" or "Discontinuities"



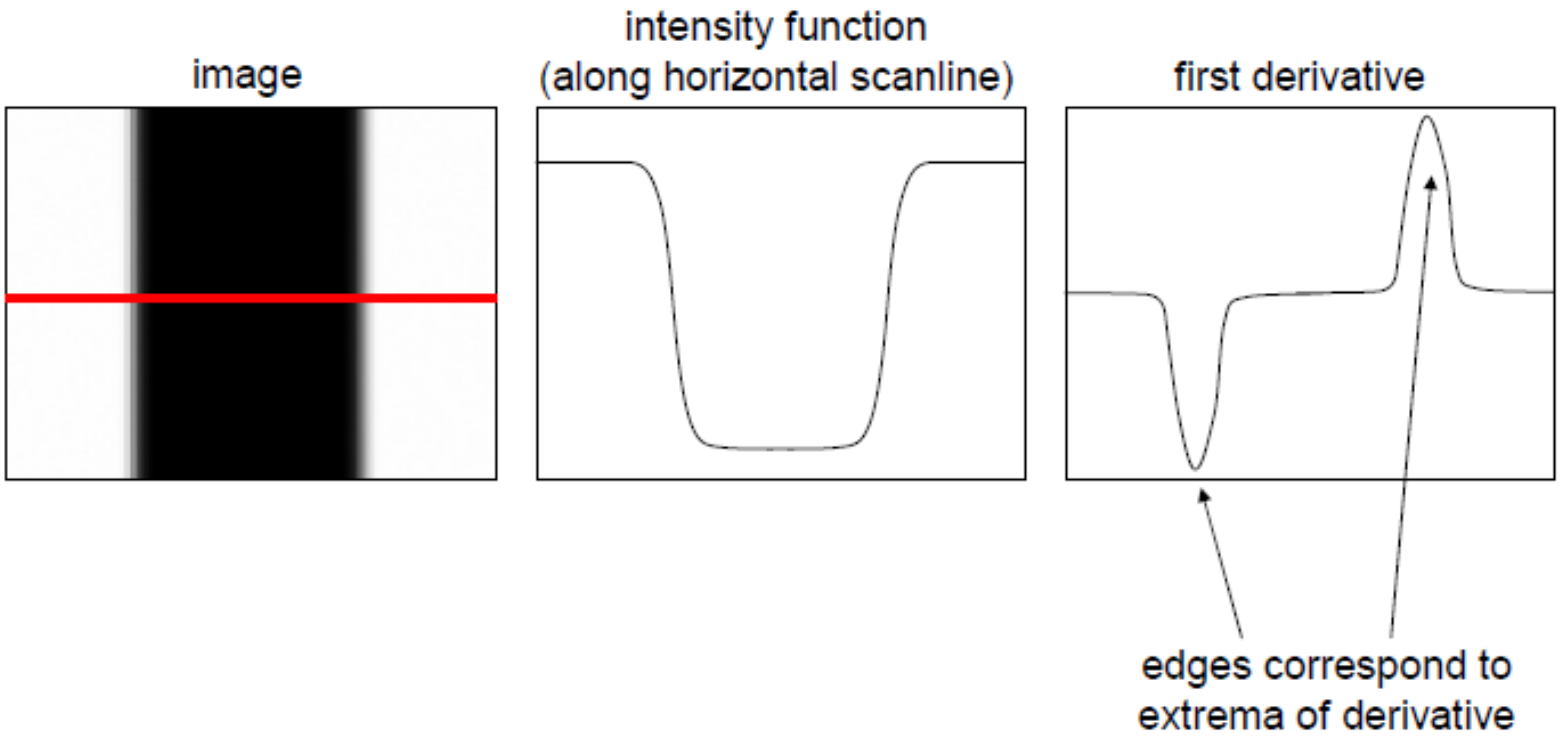


Example: Characterization of discontinuities?

Source: http://web.media.mit.edu/~maov/classes/vision09/lect/09_Image_Filtering_Edge_Detection_09.pdf

Characterizing edges

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



Source: http://web.media.mit.edu/~maov/classes/vision09/lect/09_Image_Filtering_Edge_Detection_09.pdf

Differentiation and convolution

- Recall, for 2D function, f(x,y):

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

- We could approximate this as

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

- This is linear and shift invariant, so must be the result of a convolution.

- (which is obviously a convolution)

-1	1

Source: D. Forsyth, D. Lowe

Source: http://web.media.mit.edu/~maov/classes/vision09/lect/09_Image_Filtering_Edge_Detection_09.pdf

Edges are not only horizontal or vertical

2D functions:

- The first derivate of an image can be computed using the gradient:

$$\nabla f \qquad grad(f) = \begin{pmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{pmatrix}$$

Gradient Representation

- The gradient is a vector which has magnitude and direction:

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

$$direction(grad(f)) = \tan^{-1}(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x})$$

$$|\frac{\partial f}{\partial x}| + |\frac{\partial f}{\partial y}|$$

- **Magnitude:** indicates edge strength.
- **Direction:** indicates edge direction.
  - i.e., perpendicular to edge direction

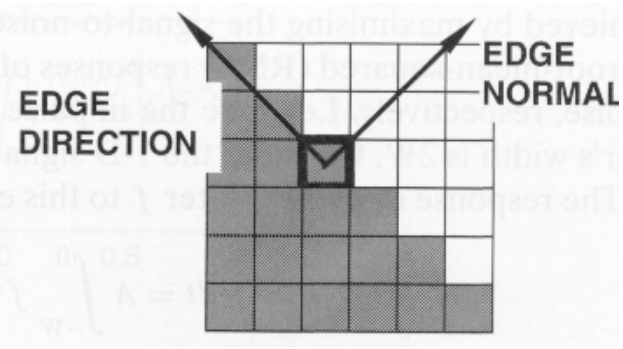
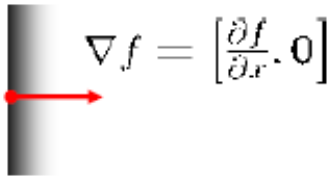


Image gradient

- 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 is given by:

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

  - how does this relate to the direction of the edge? *perpendicular*
- The *edge strength* is given by the gradient magnitude

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

Source: http://web.media.mit.edu/~maov/classes/vision09/lect/09_Image_Filtering_Edge_Detection_09.pdf

Approximate Gradient

- Approximate gradient using finite differences:

$$\frac{\partial f}{\partial x} = \lim_{h->0} \frac{f(x+h,y) - f(x,y)}{h} \qquad \frac{\partial f}{\partial y} = \lim_{h->0} \frac{f(x,y+h) - f(x,y)}{h}$$

$$\frac{\partial f}{\partial x} = \frac{f(x+h_x,y) - f(x,y)}{h_y} = f(x+1,y) - f(x,y), \quad (h_x=1)$$

$$\frac{\partial f}{\partial y} = \frac{f(x,y+h_y) - f(x,y)}{h_y} = f(x,y+1) - f(x,y), \quad (h_y=1)$$

Derivatives: Finite Differences

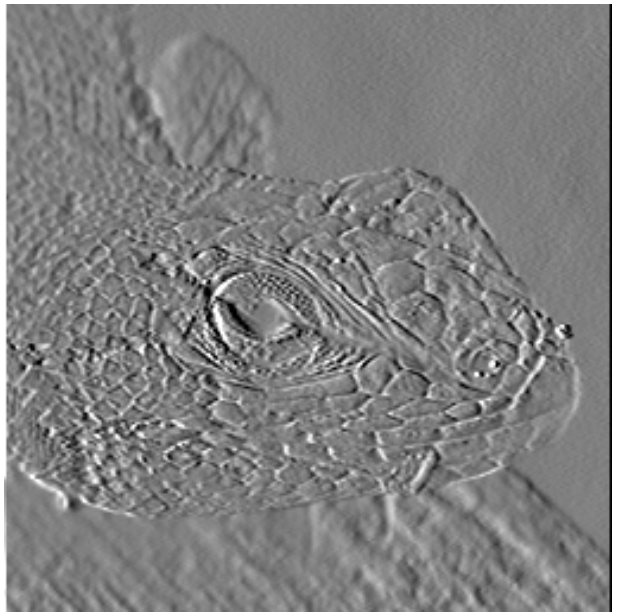
$$\frac{\partial f}{\partial x} \approx \frac{1}{2h}\left(f(x+1,y) - f(x-1,y)\right)$$

$$\frac{\partial f}{\partial x} \approx w_{dx} \circ f \qquad w_{dx} = \begin{array}{|c|c|c|} \hline -\frac{1}{2} & 0 & \frac{1}{2} \\ \hline \end{array}$$

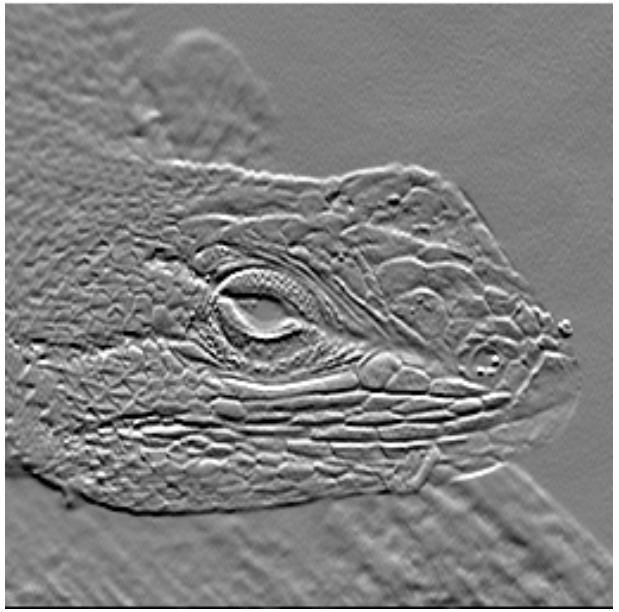
$$\frac{\partial f}{\partial y} \approx w_{dy} \circ f \qquad w_{dy} = \begin{array}{|c|} \hline -\frac{1}{2} \\ \hline 0 \\ \hline \frac{1}{2} \\ \hline \end{array}$$

Derivative Example

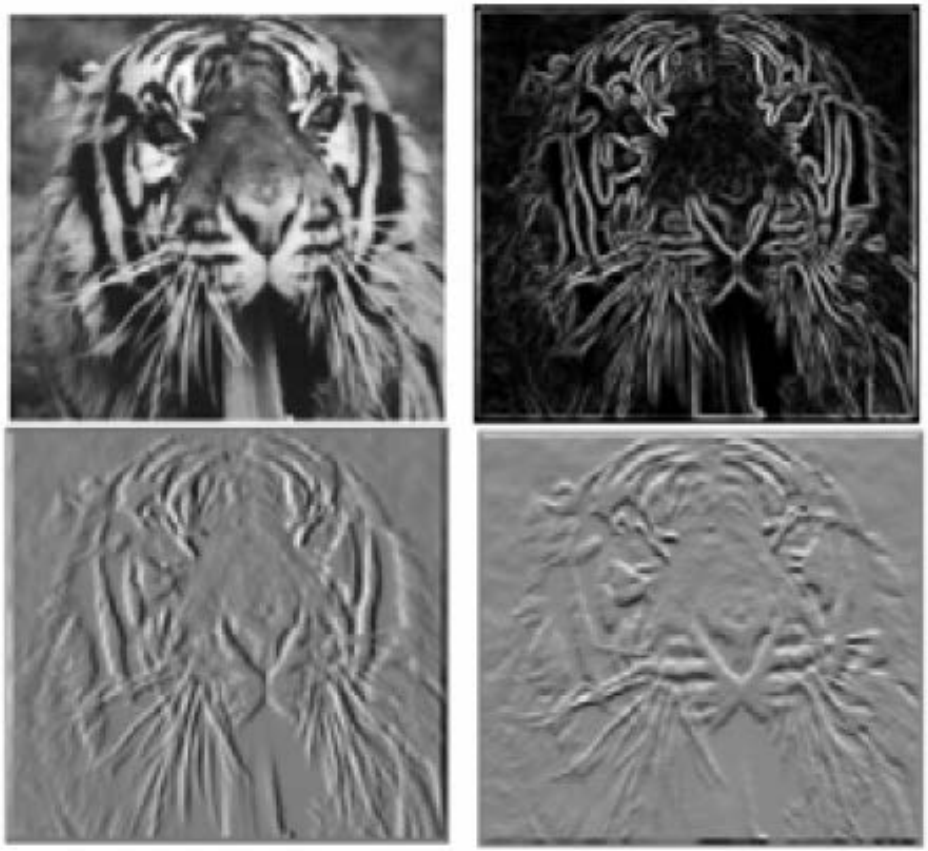
$$\begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$



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



Finite differences: example



- Which one is the gradient in the x-direction (resp. y-direction)?

Source: http://web.media.mit.edu/~maov/classes/vision09/lect/09_Image_Filtering_Edge_Detection_09.pdf

Another Approximation

- Consider the arrangement of pixels about the pixel (*i*, *j*):

3 x 3 neighborhood:

$$\begin{matrix} a_0 & a_1 & a_2 \\ a_7 & [i,j] & a_3 \\ a_6 & a_5 & a_4 \end{matrix}$$

- The partial derivatives $\frac{\partial f}{\partial x}$ $\frac{\partial f}{\partial y}$ can be computed by:

$$M_x = (a_2 + ca_3 + a_4) - (a_0 + ca_7 + a_6)$$
$$M_y = (a_6 + ca_5 + a_4) - (a_0 + ca_1 + a_2)$$

- The constant *c* implies the emphasis given to pixels closer to the center of the mask.

Prewitt Operator

- Setting $c = 1$, we get the Prewitt operator:

$$M_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \qquad M_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

M_x and M_y are approximations at (i, j)

Sobel Operator

- Setting $c = 2$, we get the Sobel operator:

$$M_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad M_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

M_x and M_y are approximations at (i, j)

Finite difference filters

- Other approximations of derivative filters exist:

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

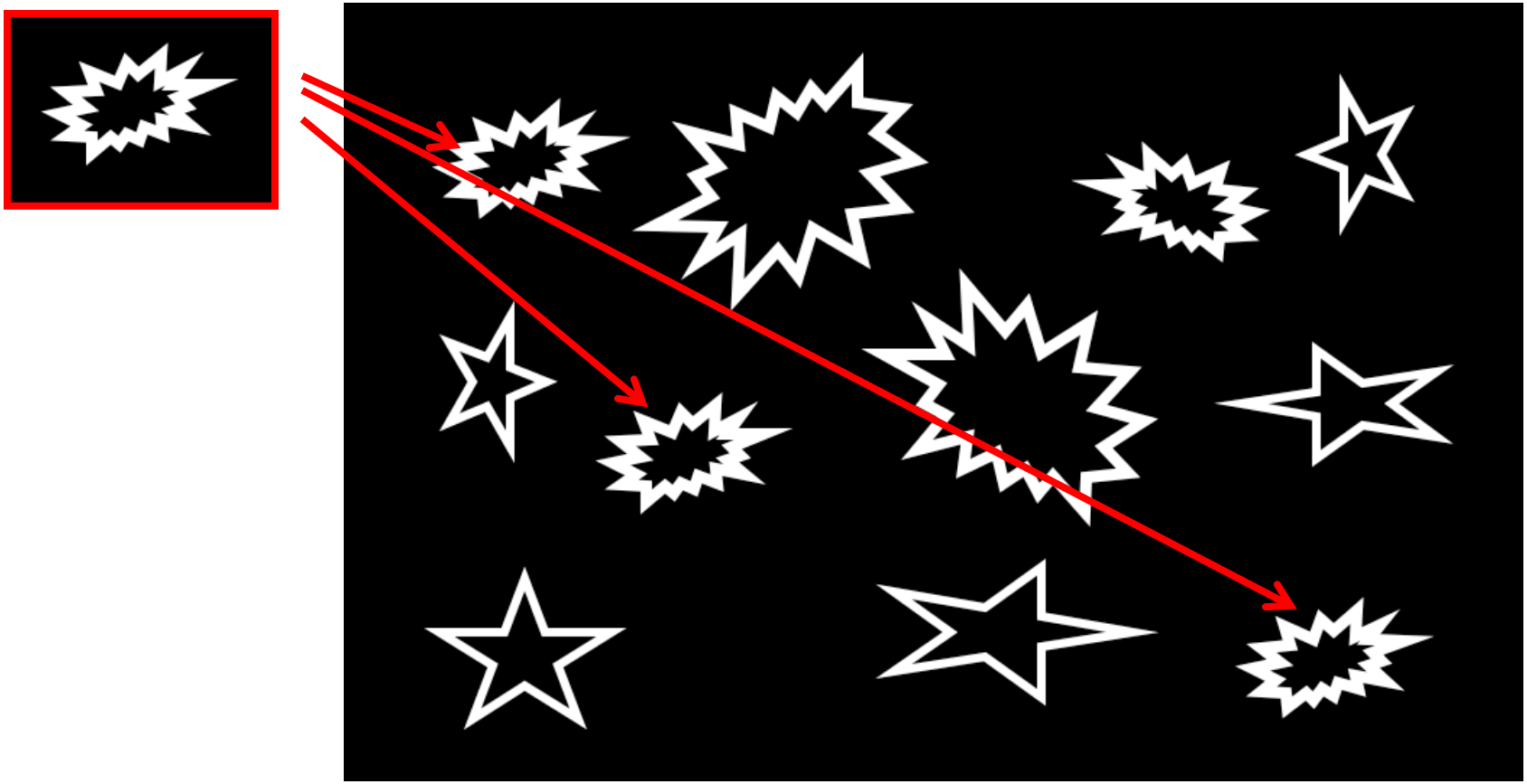
; $M_y =$

1	0
0	-1

Source: K. Grauman

Source: http://web.media.mit.edu/~maov/classes/vision09/lect/09_Image_Filtering_Edge_Detection_09.pdf

Pattern Matching



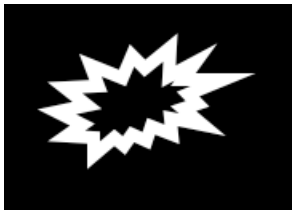
Pattern Matching/Detection

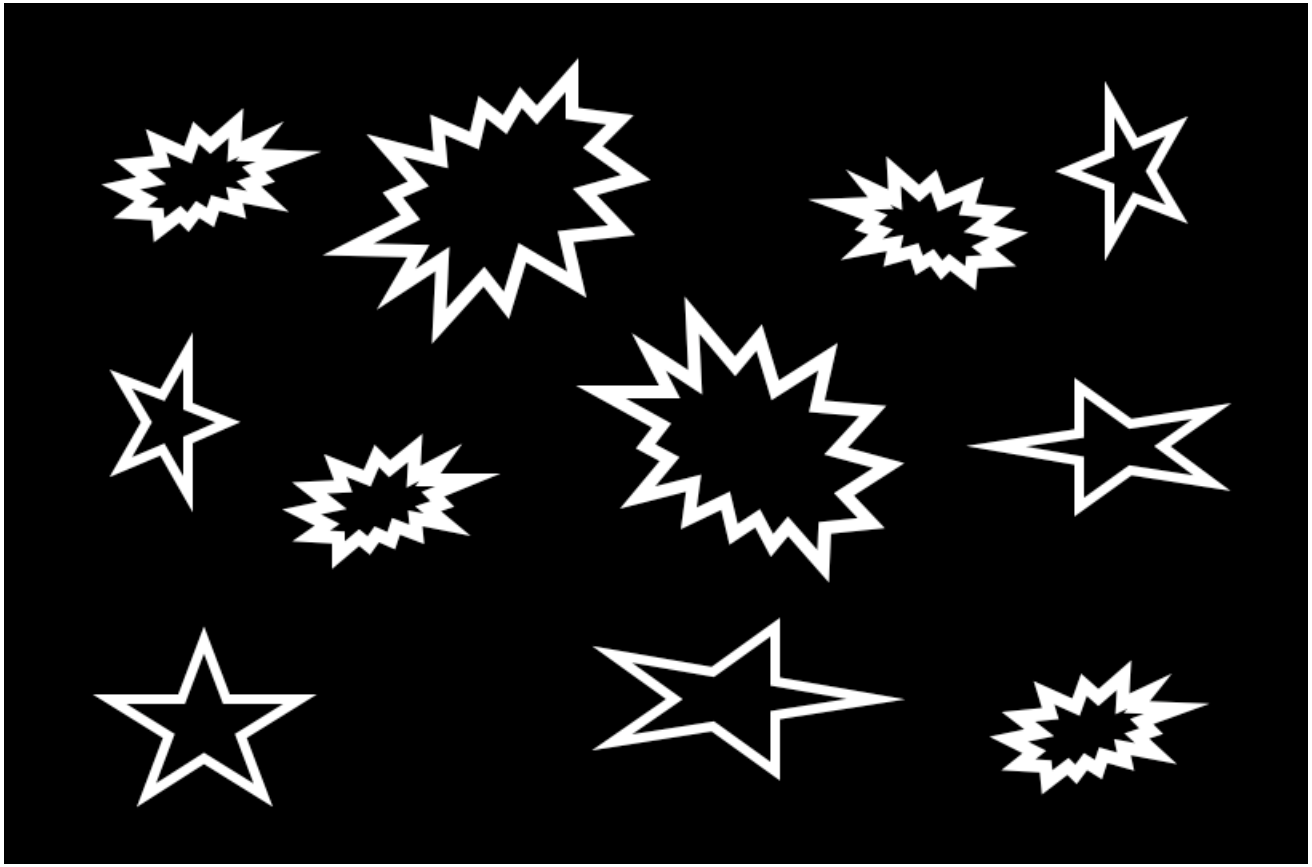
- The optimal (highest) response from a filter is the autocorrelation evaluated at position zero

$$\max_{\bar{x}} C_{ff}(\bar{x}) = C_{ff}(0) = \int f(\bar{s}) f(\bar{s}) d\bar{s}$$

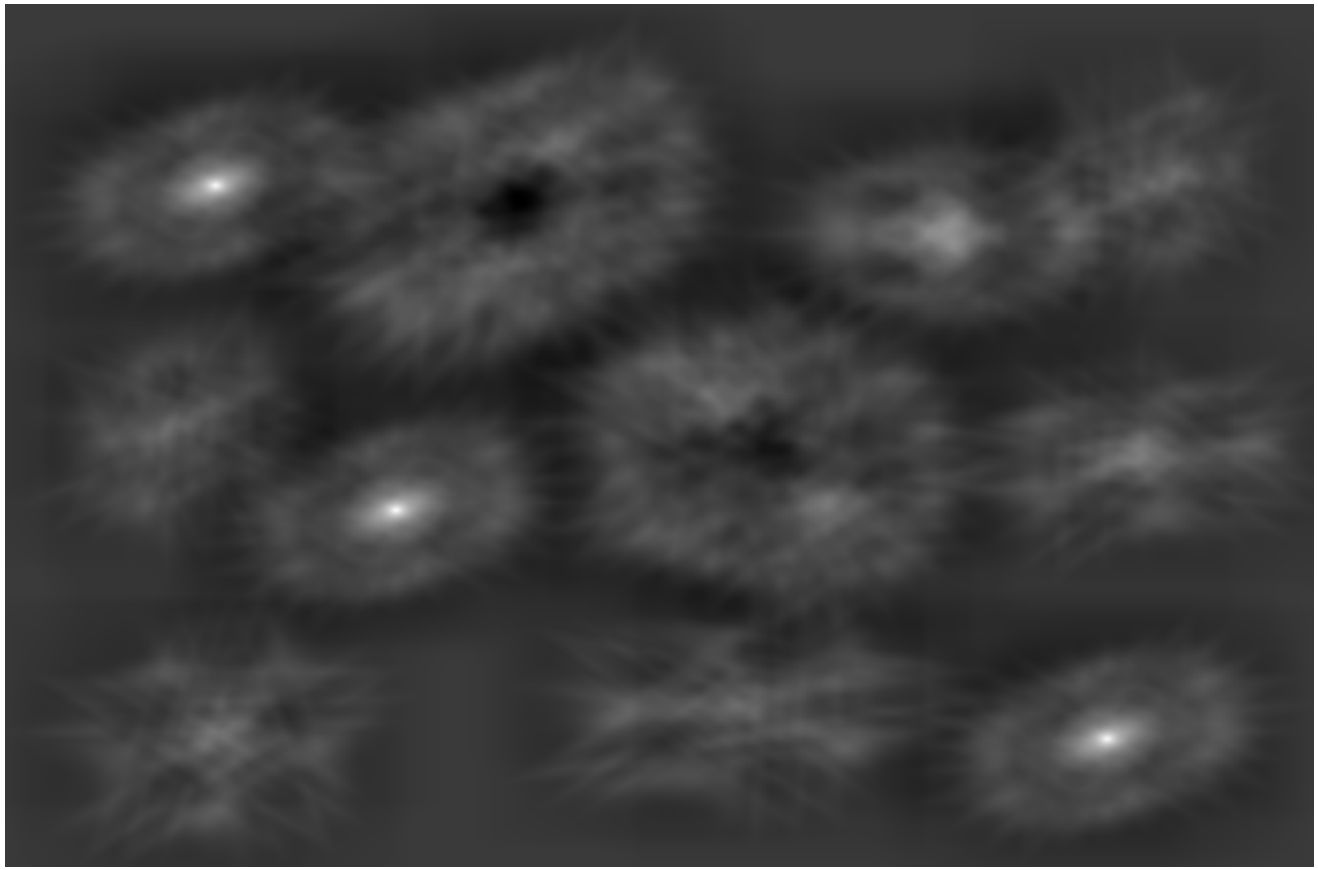
- A filter responds best when it matches a pattern that looks itself
- Strategy
  - Detect objects in images by correlation with "matched" filter

Matched Filter Example

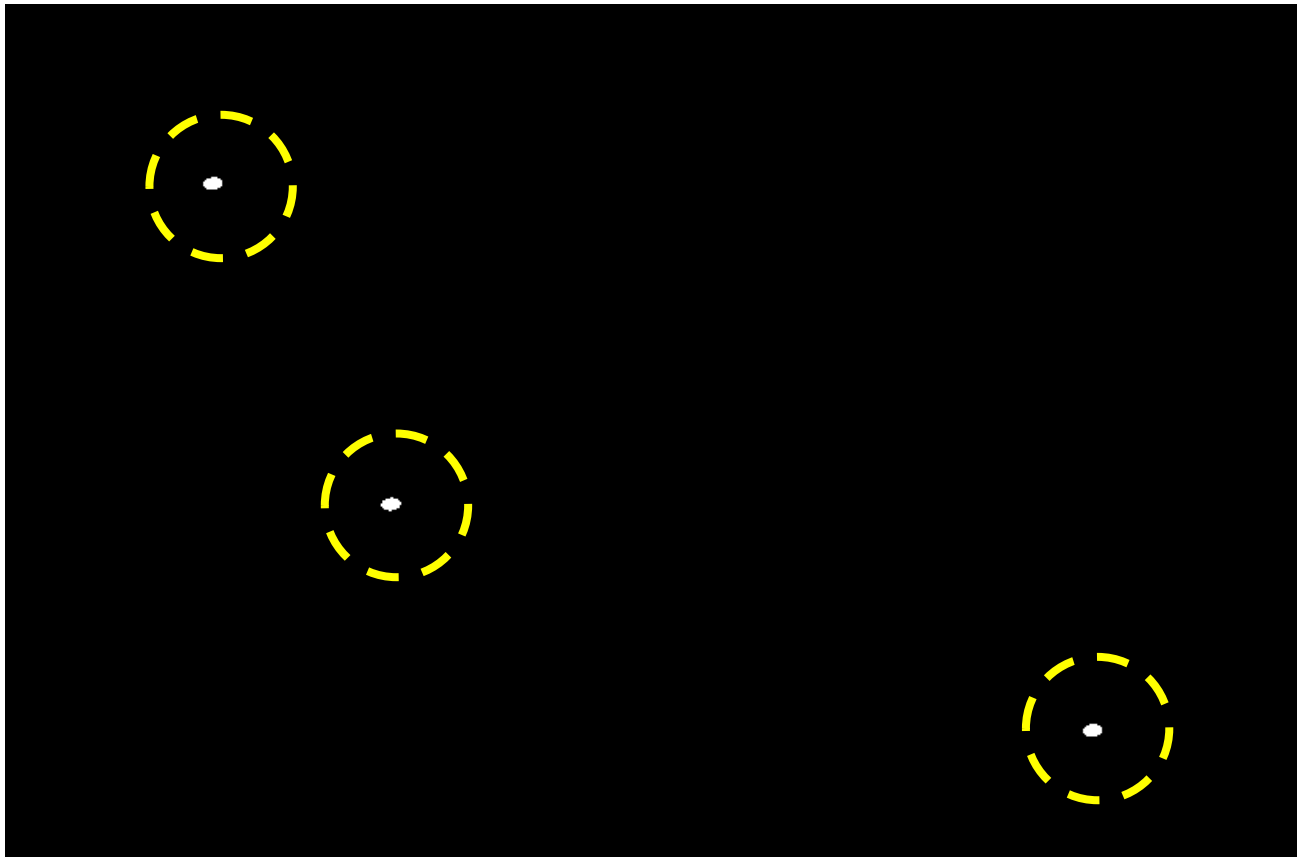




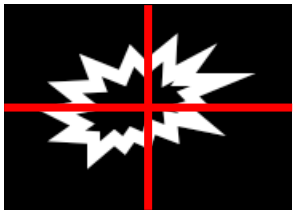
Matched Filter Example: Correlation of template with image

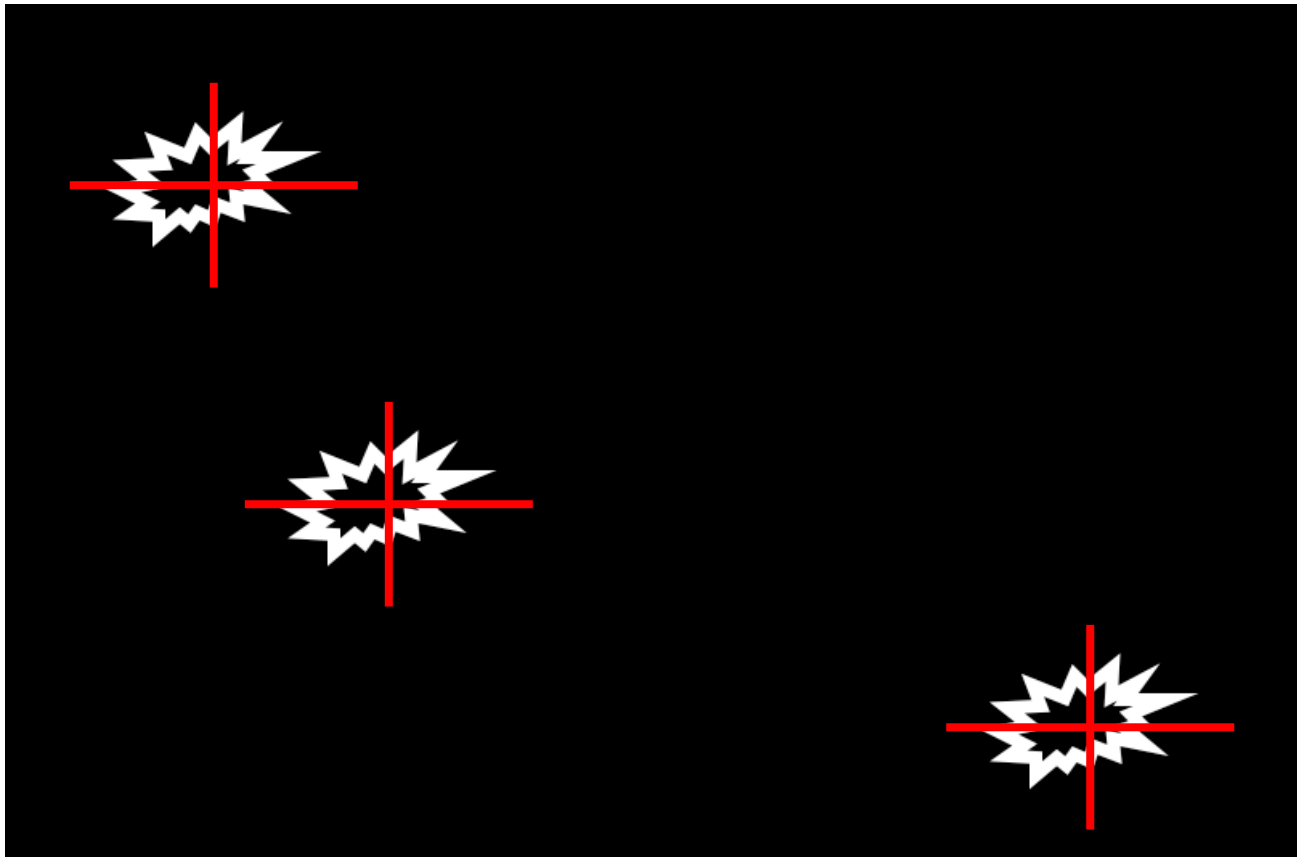


Matched Filter Example: Thresholding of correlation results

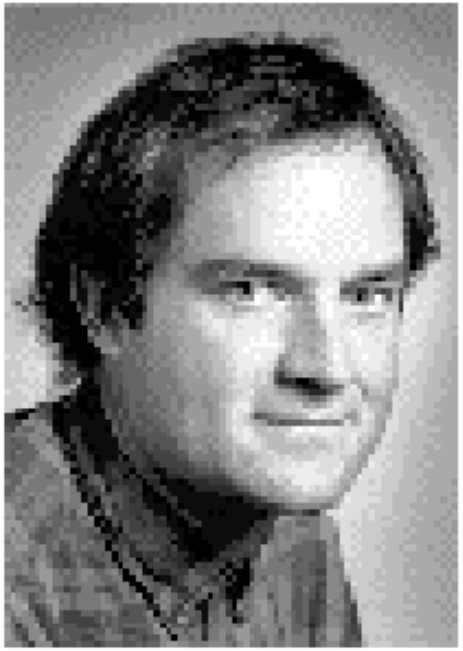


Matched Filter Example: High correlation → template found





Canny Edge Detector

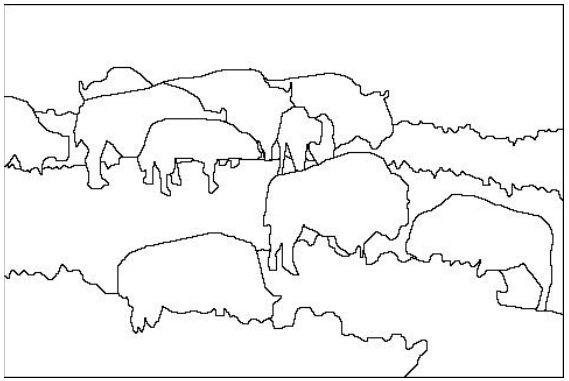


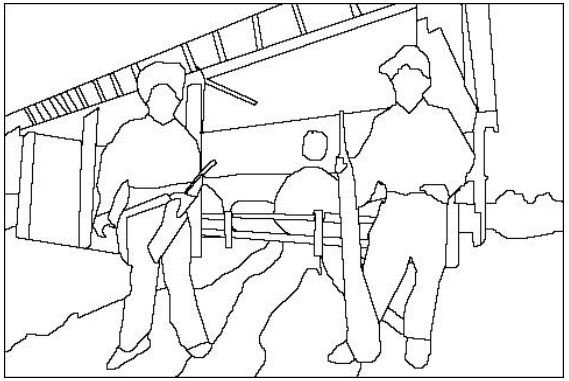
- Canny (1984) introduces several good ideas to help.
- **References:** Canny, J.F. A computational approach to edge detection. IEEE Trans Pattern Analysis and Machine Intelligence, 8(6): 679-698, Nov 1986.

J. Canny, ***A Computational Approach To Edge Detection***, IEEE Trans. Pattern Analysis and Machine Intelligence, 8:679-714, 1986.

Edge detection is part of segmentation

image | human segmentation | gradient magnitude

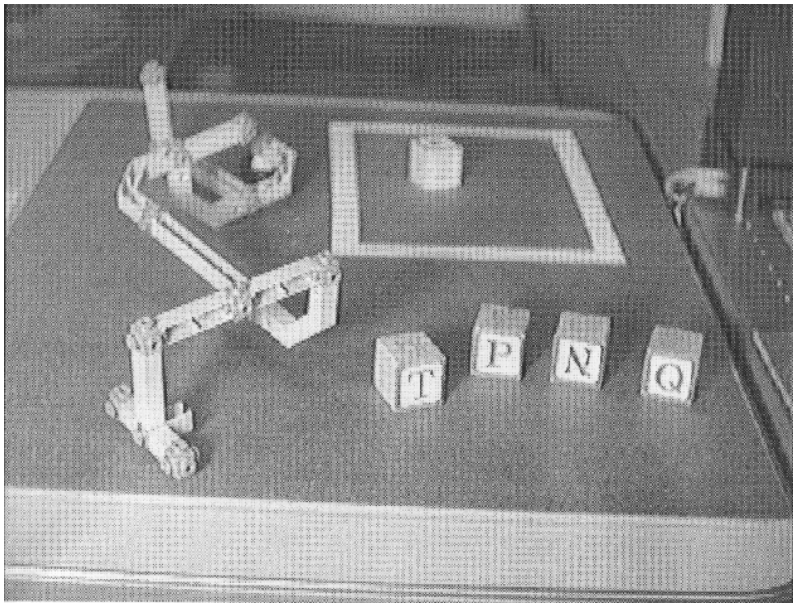



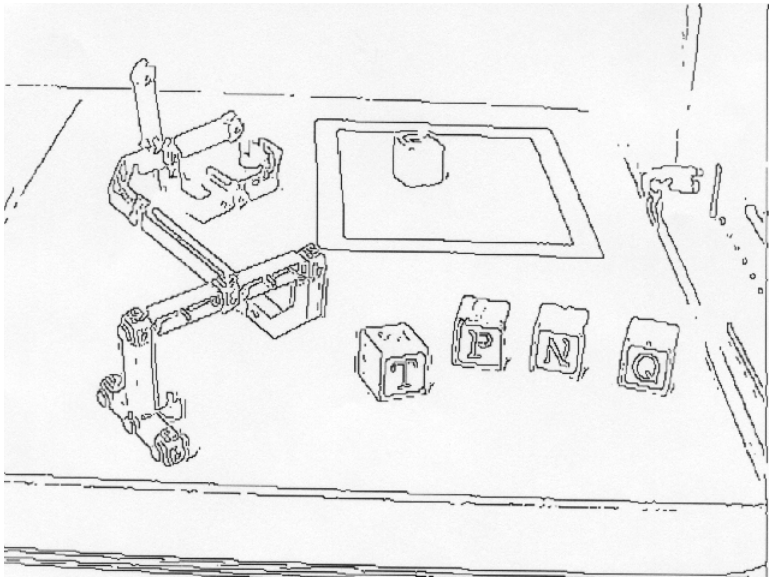



- Berkeley segmentation database:
  http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/

Goal of Edge Detection

- Produce a line "drawing" of a scene from an image of that scene.





Optimal Edge Detector Design

- Canny derives his filter by optimizing a certain performance index that favors true positive, true negative and accurate localization of detected edges
- Analysis is restricted to linear shift invariant filter that detect unblurred 1D continuous step
- Other justifiable performance criteria are possible and will lead to different filters.

Criteria for Optimal Edge Detection

- **(1) Good detection**
  - Minimize the probability of false positives (i.e., spurious edges).
  - Minimize the probability of false negatives (i.e., missing real edges).
- **(2) Good localization**
  - Detected edges must be as close as possible to the true edges.
- **(3) Single response**
  - Minimize the number of local maxima around the true edge.

What are Canny's Criteria?

- **Good detection**: low probability of not marking real edge points, and falsely marking non-edge points.

$$SNR = \frac{\left| \int_{-w}^{w} G(-x) f(x) dx \right|}{n_o \sqrt{\int_{-w}^{w} f^2(x) dx}}$$

- f is the filter, G is the edge signal, denominator is the root-mean-squared response to noise $n(x)$ only.

Localization Criterion

- Good localization: close to center of the true edge

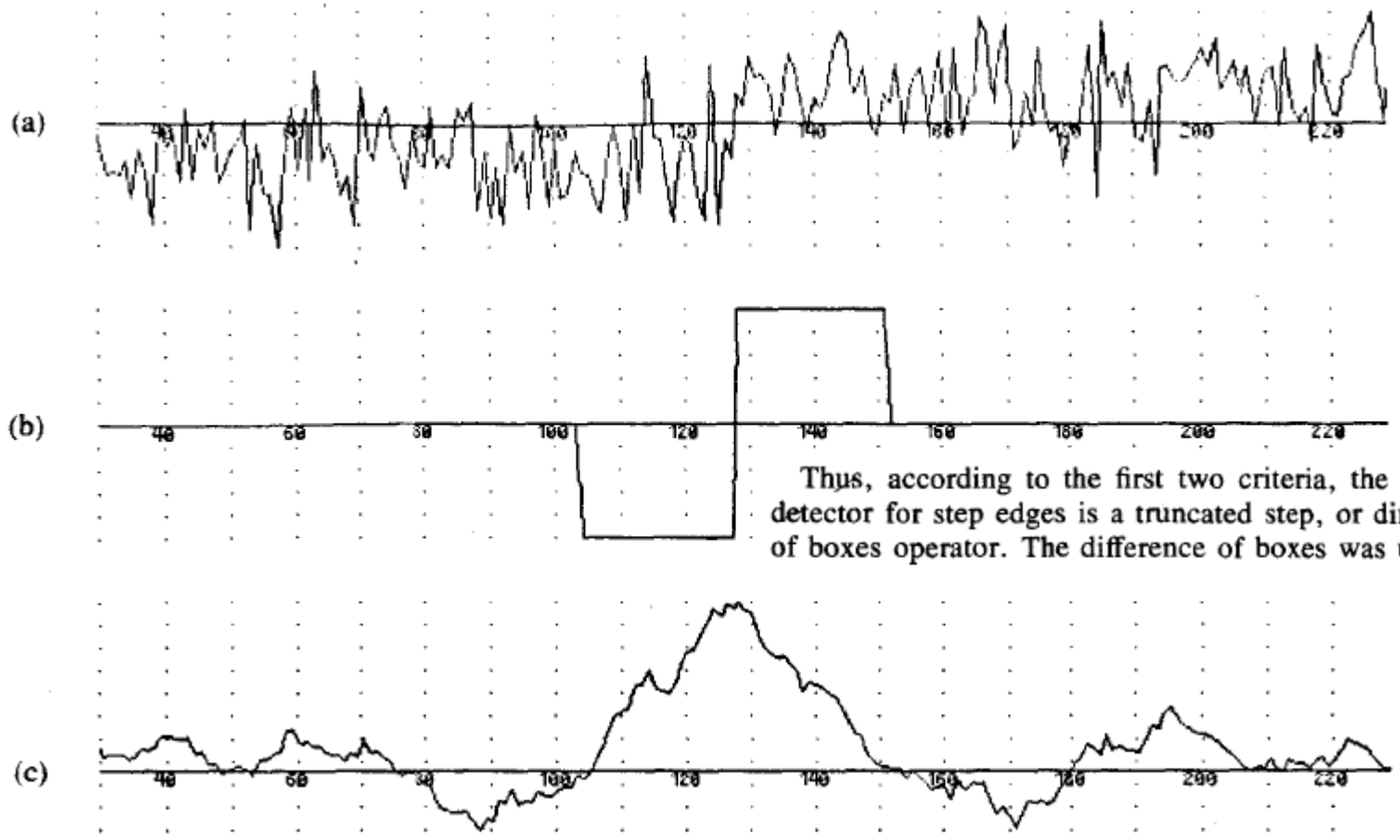
$$Localization = \frac{1}{\sqrt{E[x_0^2]}} = \frac{\left| \int_{-w}^{w} G'(-x) f'(x) dx \right|}{n_o \sqrt{\int_{-w}^{w} f'^2(x) dx}}$$

- a measure that increases as localization improves.
- Use reciprocal of the rms distance of the marked edge from the center of the true edge.

Eliminating Multiple Response

- Only one response to a single edge: implicit in first criterion, but make explicit to eliminate multiple response.
- The first two criteria can be trivially maximized by setting $f(x)=G(-x)!$
- What is this? This is a truncated step (difference of box operator).
- What is its problem?

“Optimal Operator” for Noisy Step Edge: SNR*LOC



Thus, according to the first two criteria, the optimal detector for step edges is a truncated step, or difference of boxes operator. The difference of boxes was used by

Inter-maximum Spacing

- Ideally, want to make the distance between peaks in the noise response approximate the width of the response of the operator to a single step.
- The mean distance between two adjacent maxima in the filtered response (or zero-crossing of their derivatives) can be derived as:

$$x_{zc}(f) = \pi \left(\frac{\int_{-\infty}^{\infty} f'^2(x)dx}{\int_{-\infty}^{\infty} f''^2(x)dx} \right)^{1/2}$$

- Set this distance a fraction k of the operator width W, Seek f satisfies this constraint with a fixed k. $x_{zc}(f) = kW$

Optimization

Filter Parameters						
n	x_{max}	$\Sigma\Lambda$	r	α	ω	β
1	0.15	4.21	0.215	24.59550	0.12250	63.97566
2	0.3	2.87	0.313	12.47120	0.38284	31.26860
3	0.5	2.13	0.417	7.85869	2.62856	18.28800
4	0.8	1.57	0.515	5.06500	2.56770	11.06100
5	1.0	1.33	0.561	3.45580	0.07161	4.80684
6	1.2	1.12	0.576	2.05220	1.56939	2.91540
7	1.4	0.75	0.484	0.00297	3.50350	7.47700

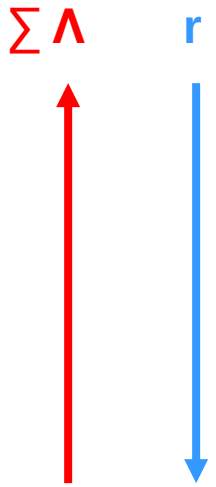


Fig. 4. Filter parameters and performance measures for the filters illustrated in Fig. 5.

∑: SNR

Λ: Localization (how close to true position)

X_max: distance between adjacent maxima (fraction of operator width)

r: multiple response performance

Optimal Operators

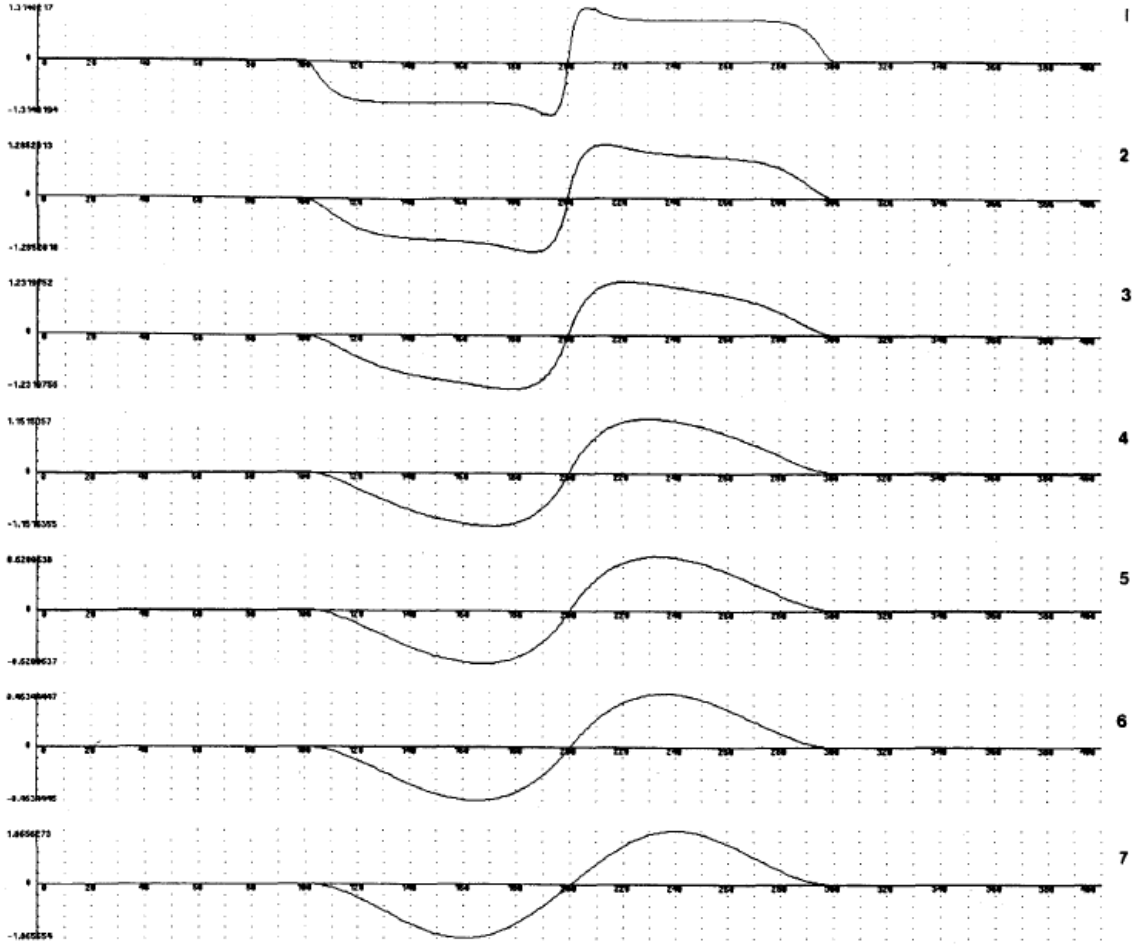
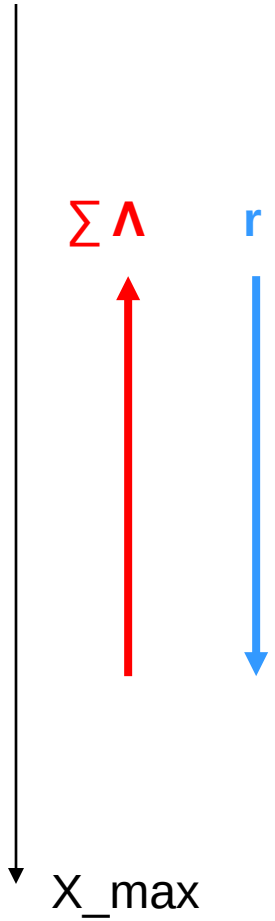


Fig. 5. Optimal step edge operators for various values of x_{max}. From top to bottom, they are $x_{max} = 0.15, 0.3, 0.5, 0.8, 1.0, 1.2, 1.4$.



Optimal Operator versus First Derivative of Gaussian

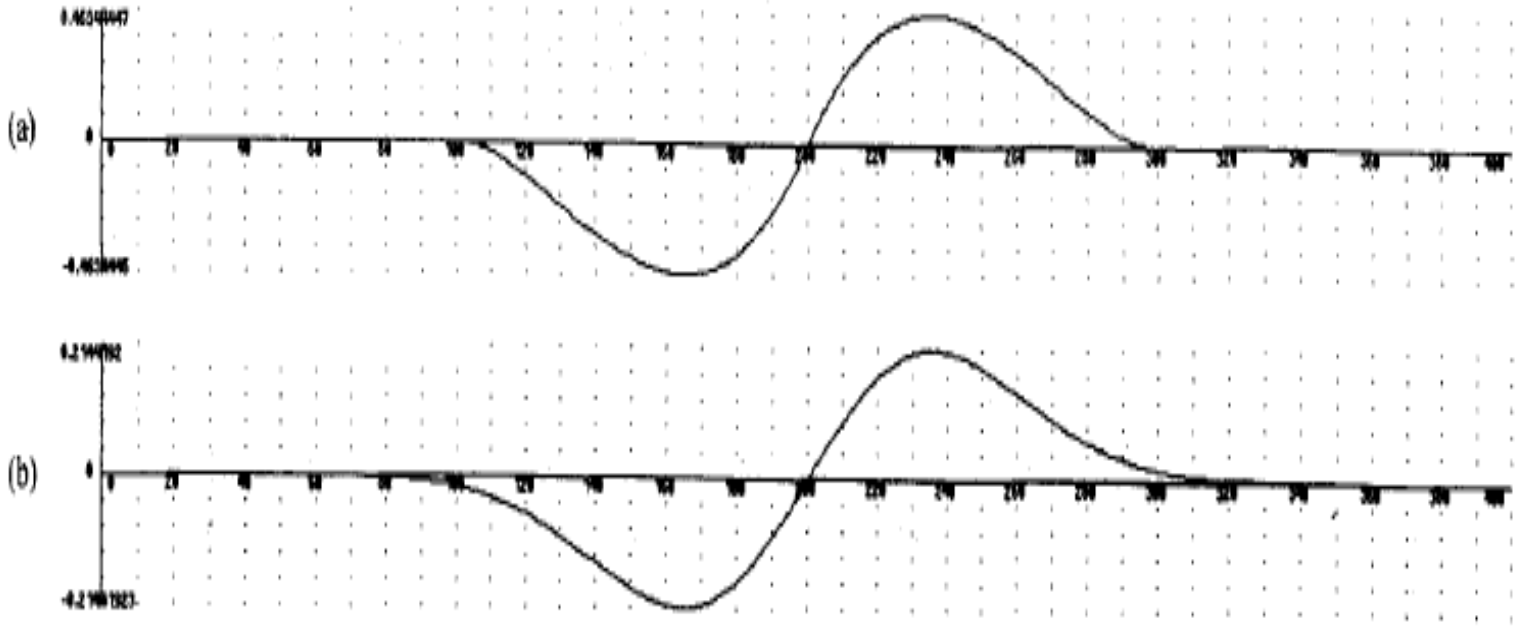


Fig. 6. (a) The optimal step edge operator. (b) The first derivative of a Gaussian.

“Optimal Operator” for Noisy Step Edge: SNR*LOC & MULT

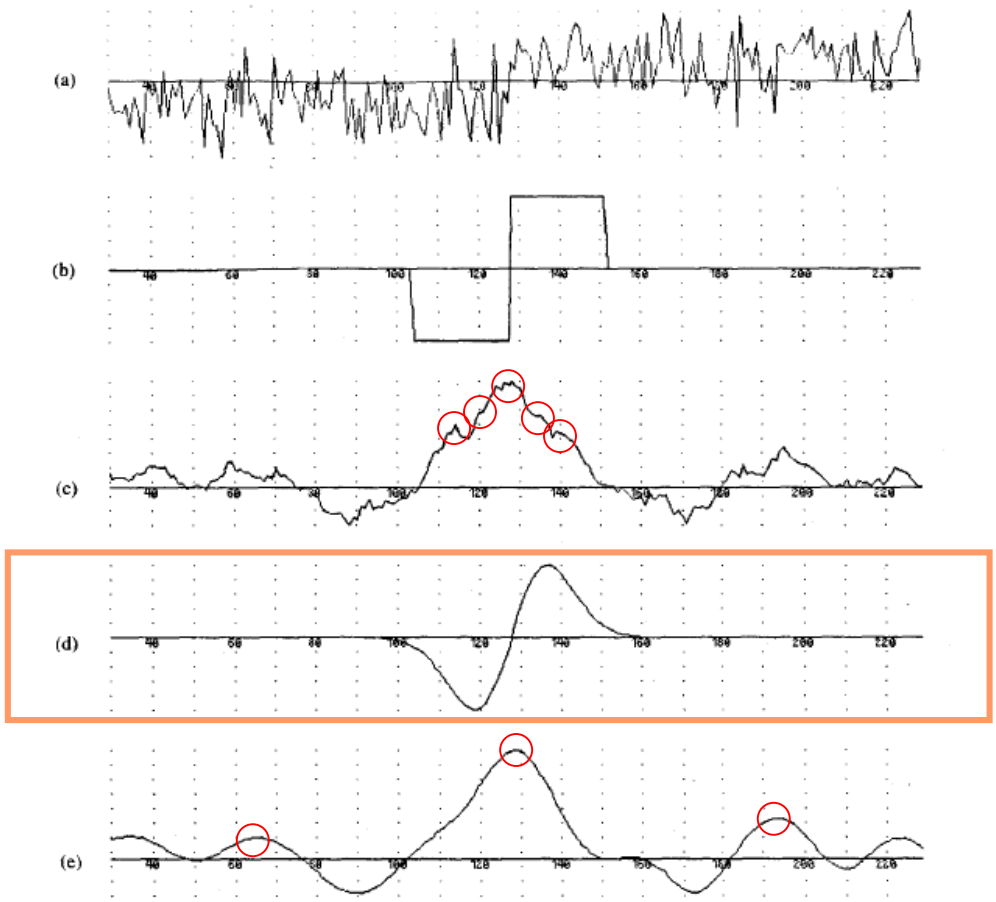
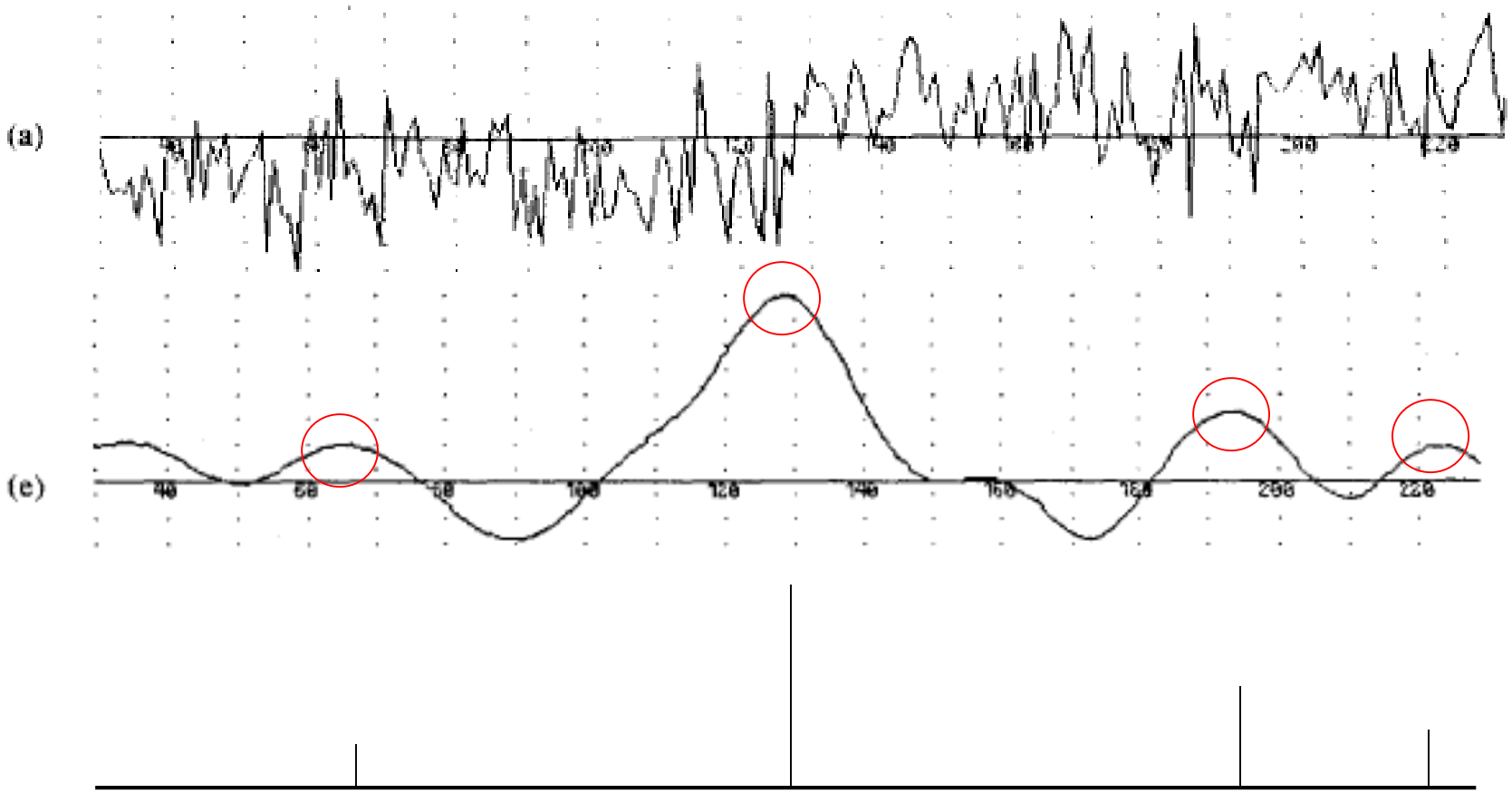


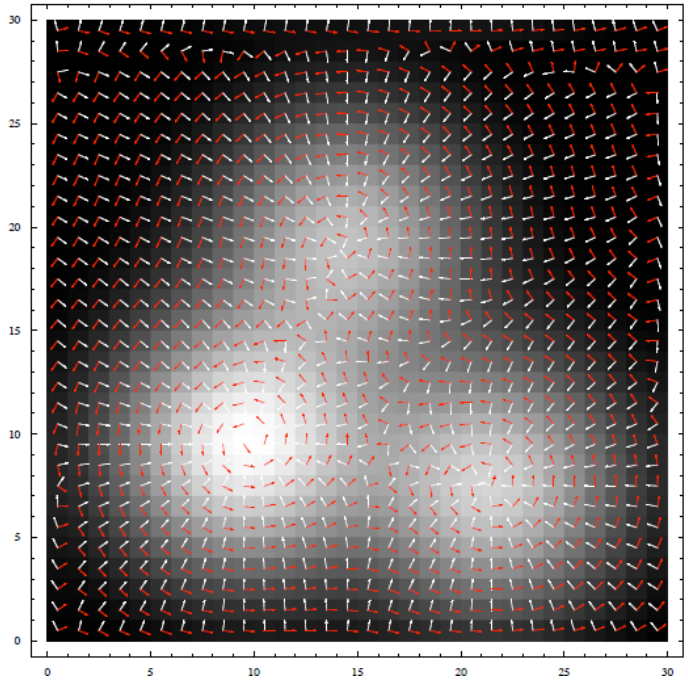
Fig. 1. (a) A noisy step edge. (b) Difference of boxes operator. (c) Difference of boxes operator applied to the edge. (d) First derivative of Gaussian operator. (e) First derivative of Gaussian applied to the edge.

Non-Maximum Suppression

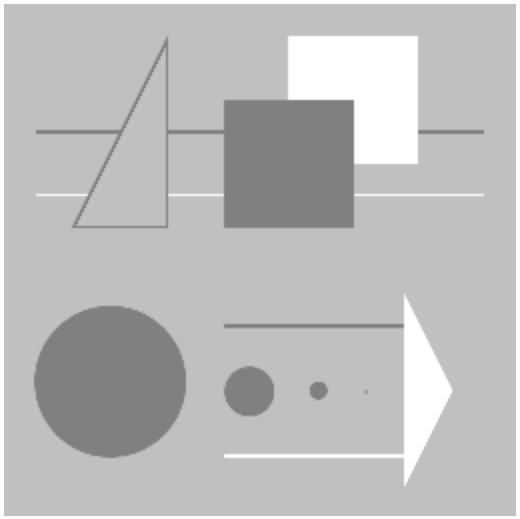


Detect local maxima and suppress all other signals.

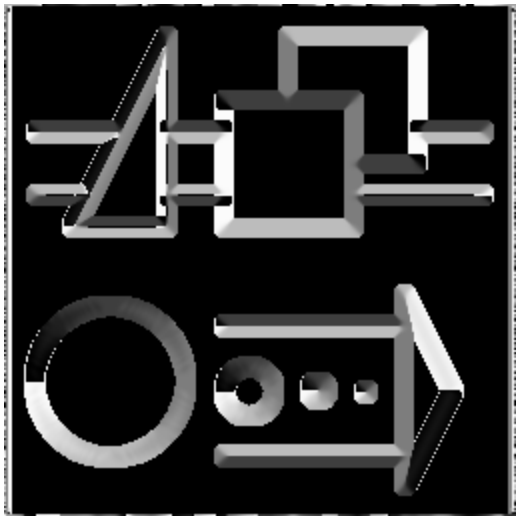
What about 2D?



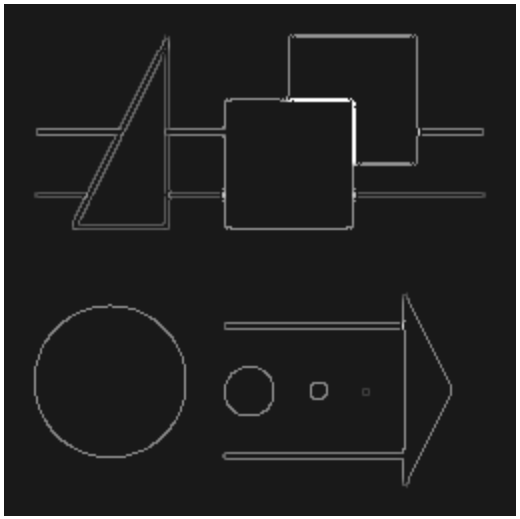
At every position in the edge-magnitude output, there is a coordinate system with normal and tangent.



original



edge orientation



edge positions coded by edge magnitude

Steps of Canny edge detector

Algorithm

1. Compute f_x and f_y

$$f_x = \frac{\partial}{\partial x}(f * G) = f * \frac{\partial}{\partial x} G = f * G_x$$

$$f_y = \frac{\partial}{\partial y}(f * G) = f * \frac{\partial}{\partial y} G = f * G_y$$

$G(x, y)$ is the Gaussian function

$G_x(x, y)$ is the derivate of $G(x, y)$ with respect to x: $G_x(x, y) = \frac{-x}{\sigma^2} G(x, y)$

$G_y(x, y)$ is the derivate of $G(x, y)$ with respect to y: $G_y(x, y) = \frac{-y}{\sigma^2} G(x, y)$

Steps of Canny edge detector ctd.

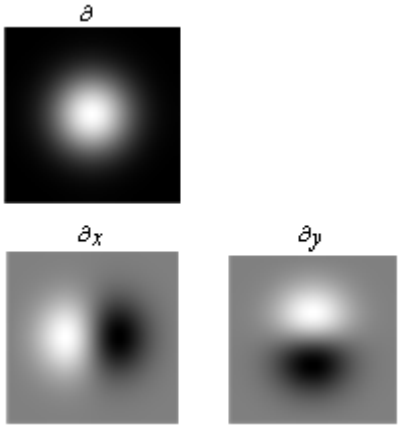
2. Compute the gradient magnitude (and direction)

$$magnitude(grad(f)) = \sqrt{\frac{\partial f}{\partial x}^2 + \frac{\partial f}{\partial y}^2} \qquad dir(x, y) = \tan^{-1}(\hat{f}_y / \hat{f}_x)$$

3. Apply non-maxima suppression.

4. Apply hysteresis thresholding/edge linking.

2D Edge Filter: Output at different scales



1st order Gaussian Derivatives

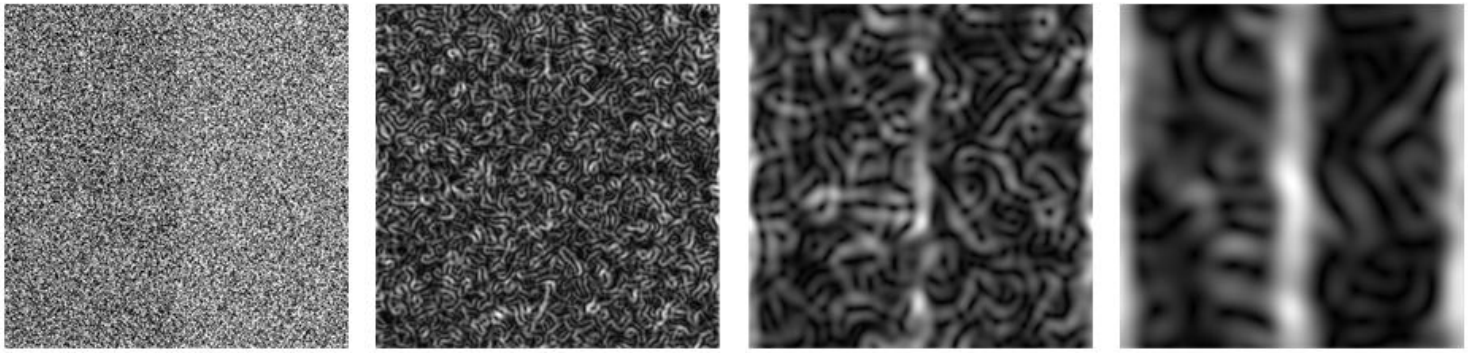


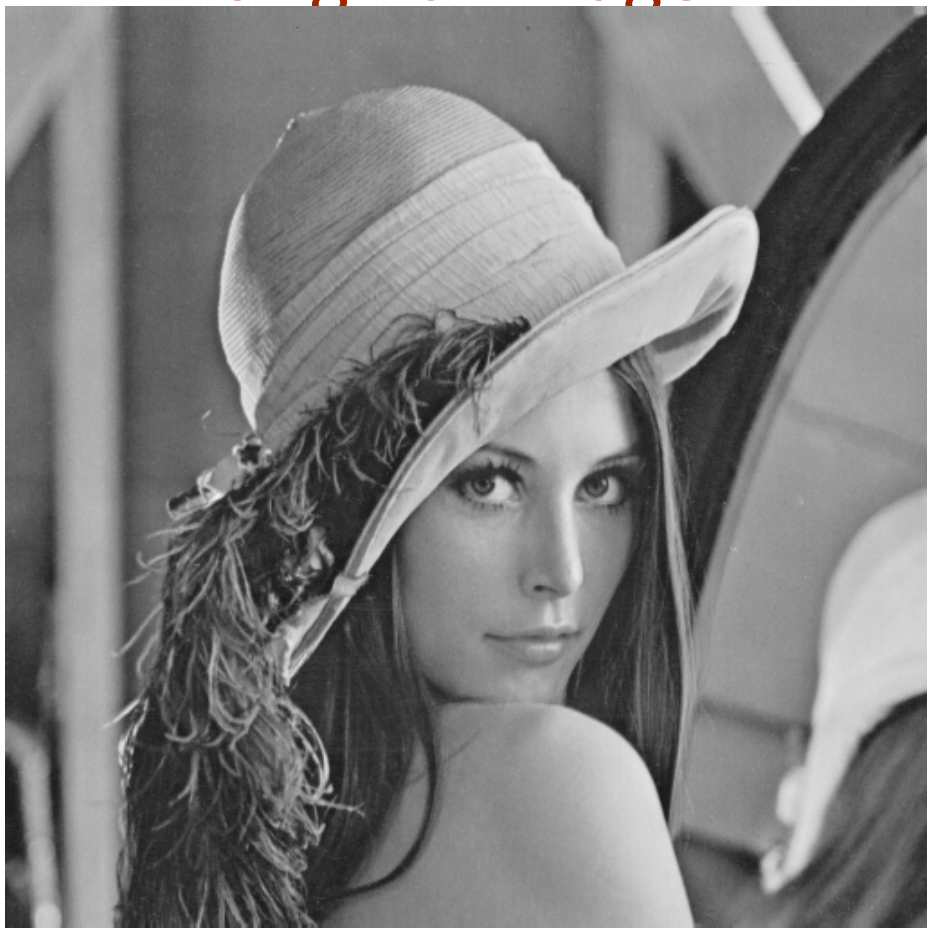
Figure 5.11 Detection of a very low contrast step-edge in noise. Left: original image, the step-edge is barely visible. At small scales (second image, $\sigma = 2$ pixels) the edge is not detected. We see the edges of the noise itself, cluttering the edge of the step-edge. Only at large scale (right, $\sigma = 12$ pixels) the edge is clearly found. At this scale the large scale structure of the edge emerges from the small scale structure of the noise.

Response at different scales

Figure 5.11 Gradient edges detected at different scales (σ = 0.5, 2, 5 pixels resp.). coarser edges (right) indicate hierarchically more 'important' edges.

Canny edge detector - example

original image



Canny edge detector – example (cont'd)

Gradient magnitude

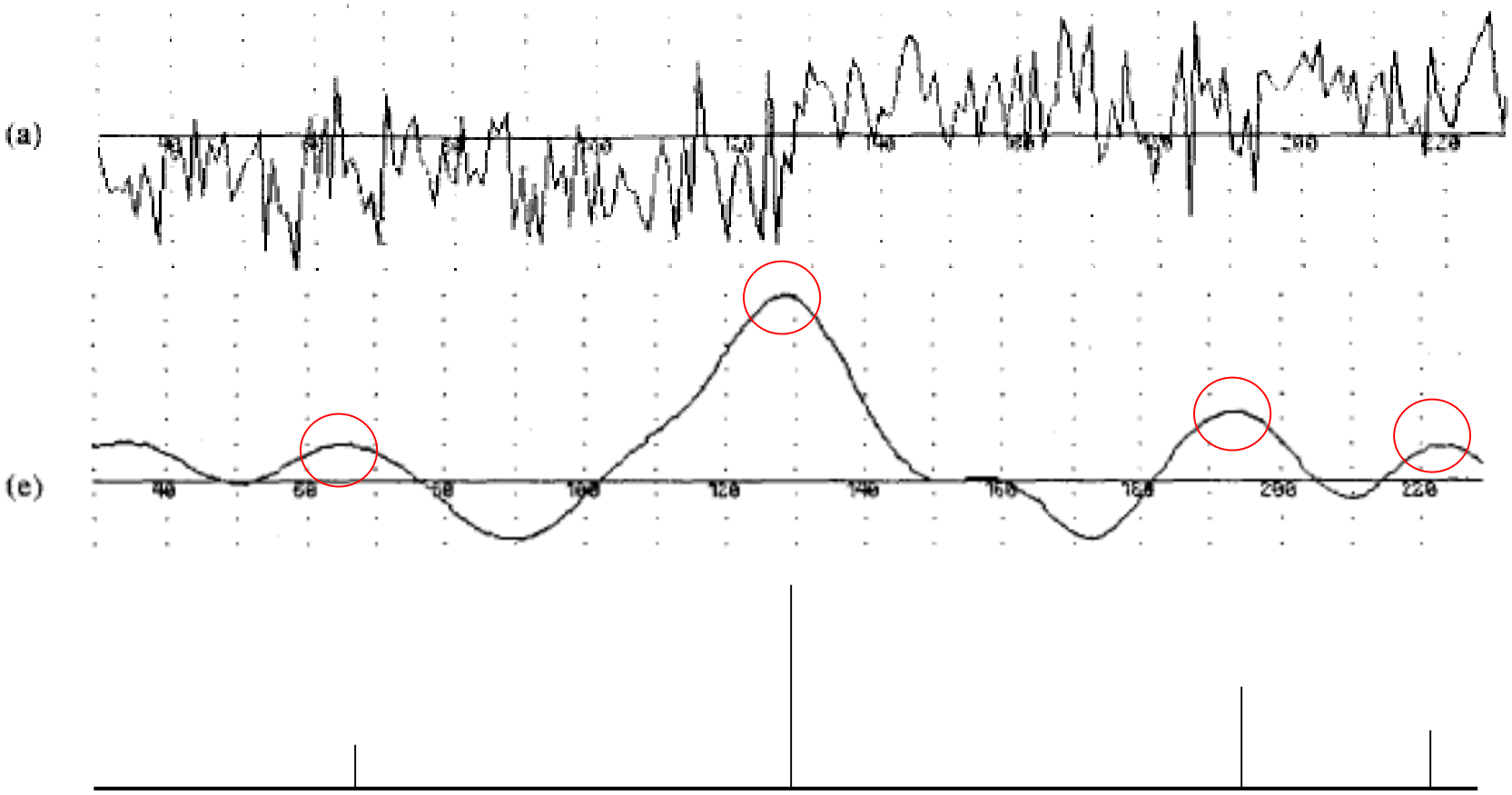
Canny edge detector – example (cont'd)

Thresholded gradient magnitude

Canny edge detector – example (cont'd)

Thinning (non-maxima suppression)

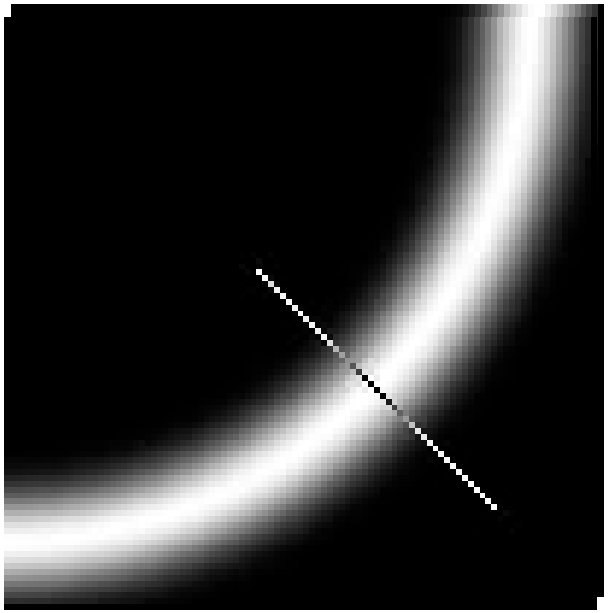
Non-Maximum Suppression



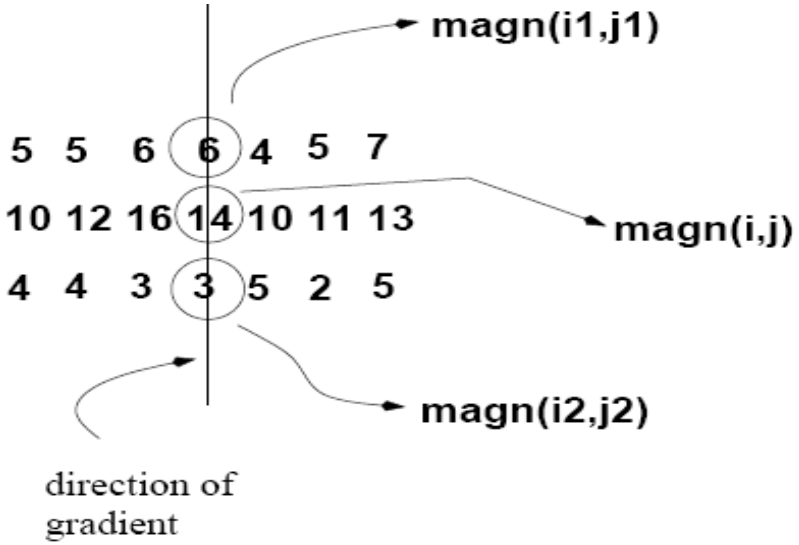
Detect local maxima and suppress all other signals.

Non-maxima suppression

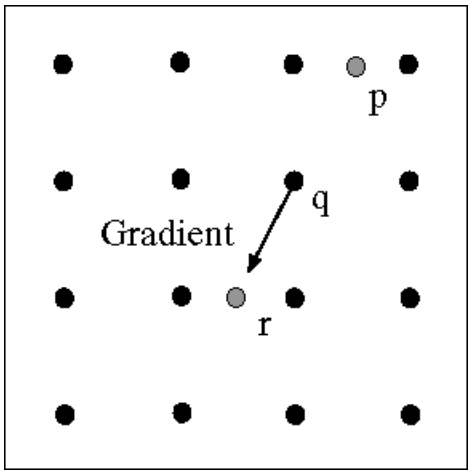
- Check if gradient magnitude at pixel location (i,j) is local maximum along gradient direction



Non-maxima suppression (cont'd)



Warning: requires checking interpolated pixels p and r



Algorithm

For each pixel (i,j) do:

if $magn(i, j) < magn(i_1, j_1)$ or $magn(i, j) < magn(i_2, j_2)$
then $I_N(i, j) = 0$
else $I_N(i, j) = magn(i, j)$

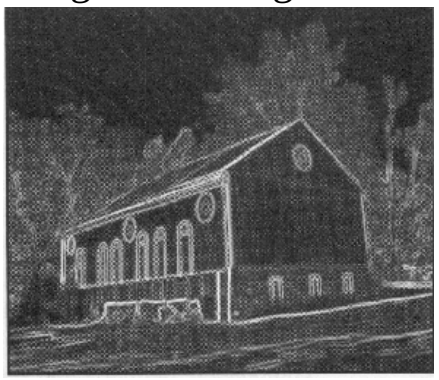
Hysteresis thresholding

- Standard thresholding:

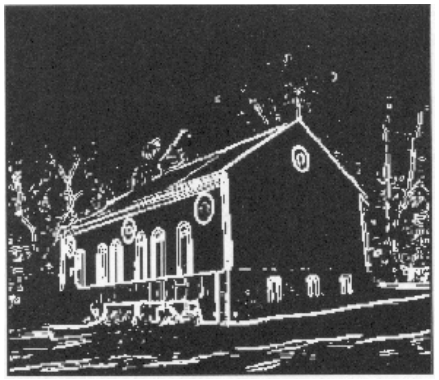
$$E(x,y) = \begin{cases} 1 & \text{if } \|\nabla f(x,y)\| > T \text{ for some threshold } T \\ 0 & \text{otherwise} \end{cases}$$

- Can only select “strong” edges.
- Does not guarantee “continuity”.

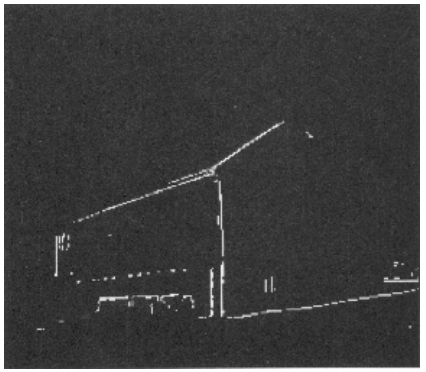
gradient magnitude



low threshold



high threshold



Hysteresis thresholding (cont'd)

- Hysteresis thresholding uses two thresholds:

  - low threshold t_l
  - high threshold t_h (usually, $t_h = 2t_l$)

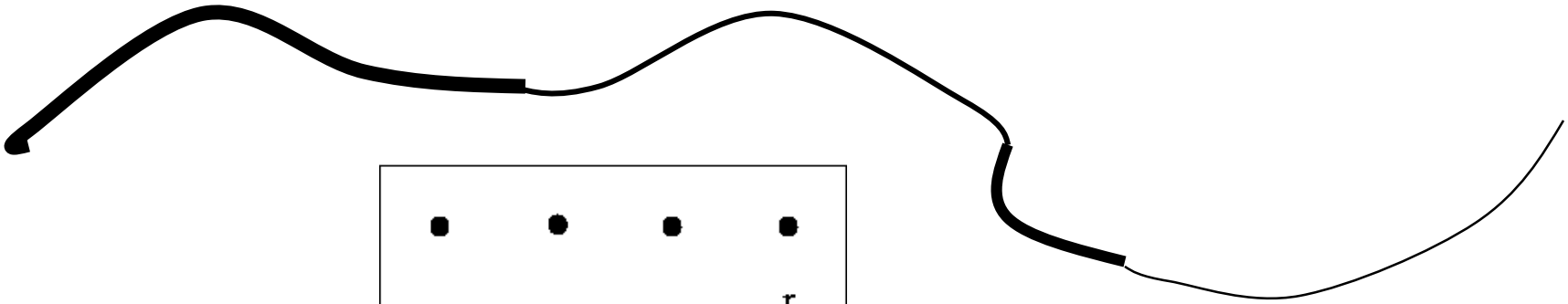
$$\begin{array}{rrl} & \|\nabla f(x,y)\| \geq & t_h \\ t_l \geq & \|\nabla f(x,y)\| < & t_h \\ & \|\nabla f(x,y)\| < & t_l \end{array}$$

definitely an edge
maybe an edge, depends on context
definitely not an edge

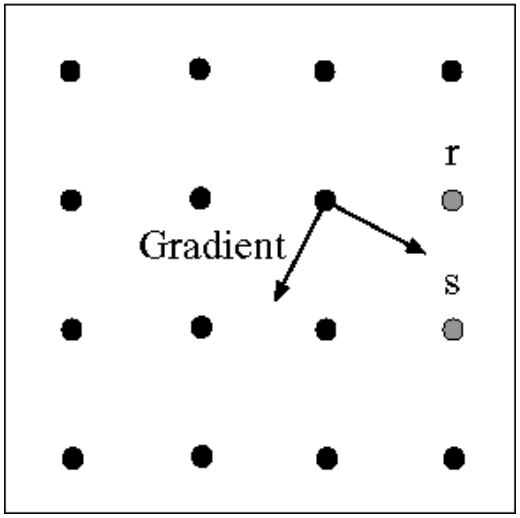
- For "maybe" edges, decide on the edge if neighboring pixel is a strong edge.

Hysteresis thresholding/Edge Linking

Idea: use a **high** threshold to start edge curves and a **low** threshold to continue them.



Use edge "direction" for linking edges



Hysteresis Thresholding/Edge Linking (cont'd)

Algorithm

1. Produce two thresholded images $I_1(i, j)$ and $I_2(i, j)$. (using t_l and t_h)

(note: since $I_2(i, j)$ was formed with a high threshold, it will contain fewer false edges but there might be gaps in the contours)

2. Link the edges in $I_2(i, j)$ into contours

    2.1 Look in $I_1(i, j)$ when a gap is found.

    2.2 By examining the 8 neighbors in $I_1(i, j)$, gather edge points from $I_1(i, j)$ until the gap has been bridged to an edge in $I_2(i, j)$.

Note: large gaps are still difficult to bridge.
(i.e., more sophisticated algorithms are required)

Effect of scale (i.e., σ)

original

$\sigma = 1$

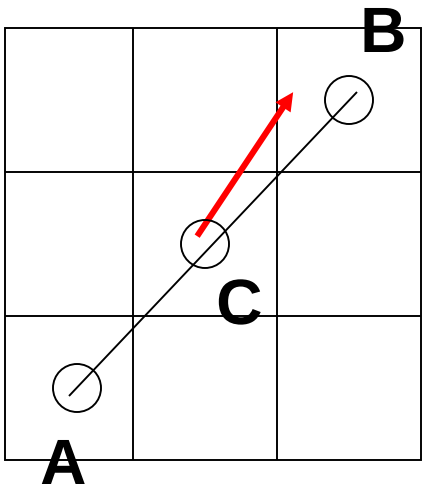
$\sigma = 2$

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

ADDITIONAL SLIDES CANNY THESIS

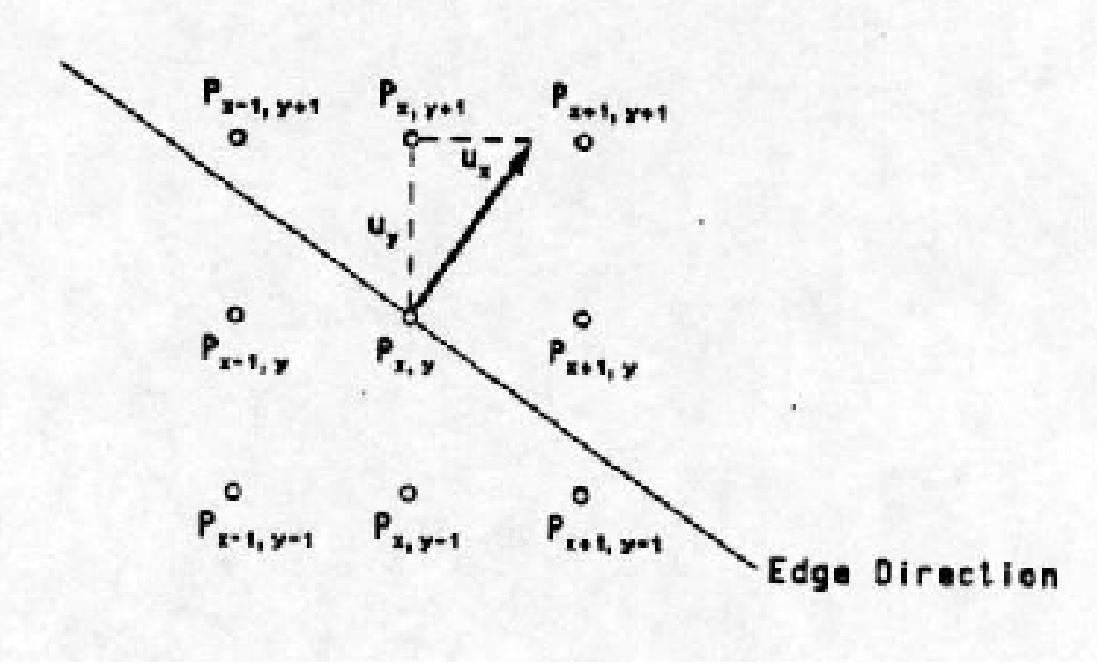
Non-Maximum Suppression

- Canny: Interpolate Gradient along gradients (plus and minus a certain distance) and check if center is larger than neighbors.
- Simplified: Test for each Gradient Magnitude pixel if neighbors along gradient direction (closest neighbors) are smaller than center: Mark C as maximum if A<C and B<C



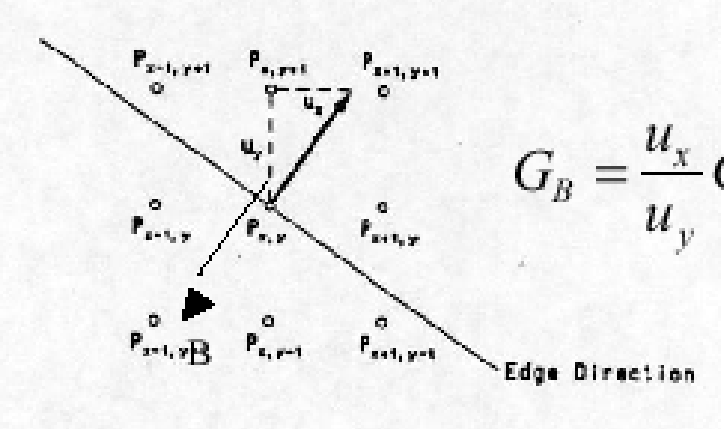
Non-maximum suppression

At each point, compute its edge gradient, compare with the gradients of its neighbors along the gradient direction. If smaller, turn 0; if largest, keep it.



Estimation of Gradient

- The interpolated gradient on the other side is given by:



$$G_B = \frac{u_x}{u_y} G(x-1, y-1) + \frac{u_y - u_x}{u_y} G(x, y-1)$$

- Mark $P_{x,y}$ as a maximum if $G(x,y) > G_A$ and $G(x,y) > G_B$
- Interpolation always involve 1 diagonal and 1 non-diagonal point. Avoid division by multiplying through by u_y .

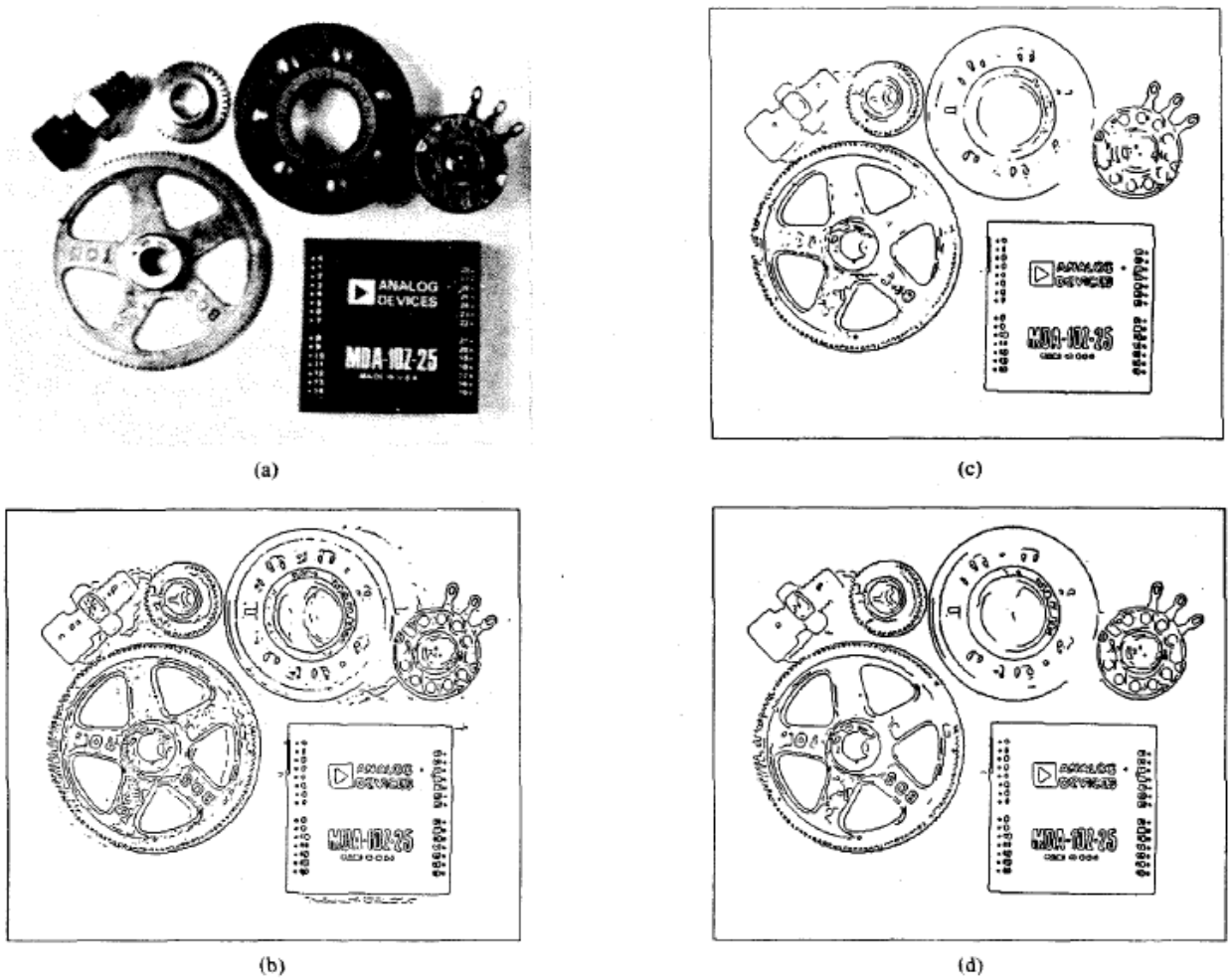
Non-maximum suppression

- This scheme involves 4 multiplication per point, but it is not excessive.
- Works better than simpler scheme which compares the points $P_{x,y}$ with two of its neighbors.

Canny: Hysteresis Thresholding

- Thresholding/binarization of edge map:
  - Noise and image structures have different structure
  - Simple thresholding: If too low, too many structures appear, if too high, contours are broken into pieces
  - **Idea**: Hysteresis: Upper and lower threshold, keep all connected edges (d_m metric) that are connected to upper but above lower threshold

Hysteresis Thresholding



too little

too many

combi nation

Fig. 7. (a) Parts image, 576 by 454 pixels. (b) Image thesholded at T_1. (c) Image thresholded at 2 T_1. (d) Image thresholded with hysteresis using both the thresholds in (a) and (b).

Multidimensional Derivatives

- Nabla operator: $\vec{\nabla} \equiv \left\{ \frac{\partial}{\partial x}, \frac{\partial}{\partial y} \right\}$
- Gradient: $\vec{\nabla} L$ is the gradient of L
- Laplacian: $\vec{\nabla} . \left(\vec{\nabla} L \right) = \frac{\partial^2 L}{\partial x^2} + \frac{\partial^2 L}{\partial y^2}$
- Hessian: $\vec{\nabla} \left(\vec{\nabla} L \right) = \begin{pmatrix} \frac{\partial^2 L}{\partial x^2} & \frac{\partial^2 L}{\partial x \partial y} \\ \frac{\partial^2 L}{\partial x \partial y} & \frac{\partial^2 L}{\partial y^2} \end{pmatrix}$

  (matrix of 2nd derivatives, gradient of gradient of L)

Optimal Operators for Other Structures

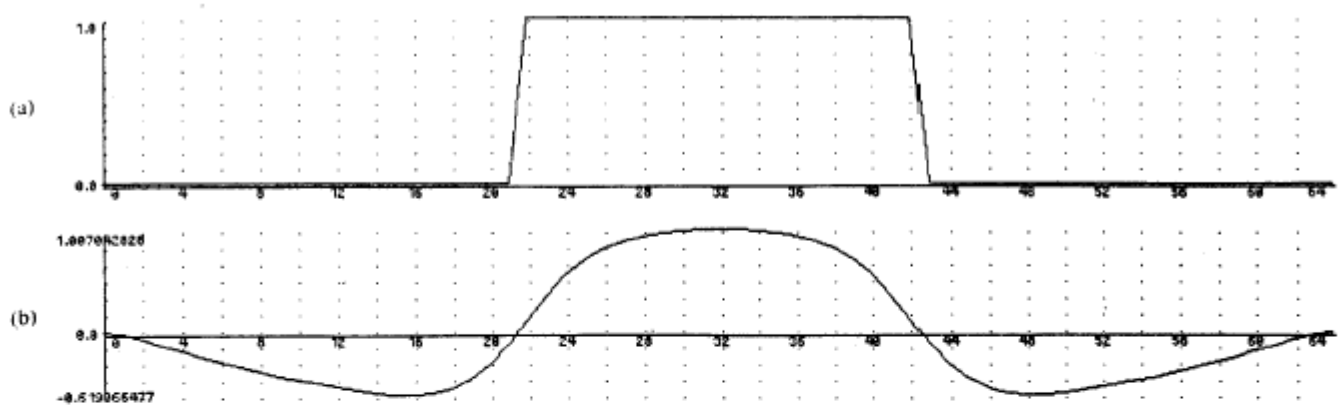


Fig. 2. A ridge profile and the optimal operator for it.

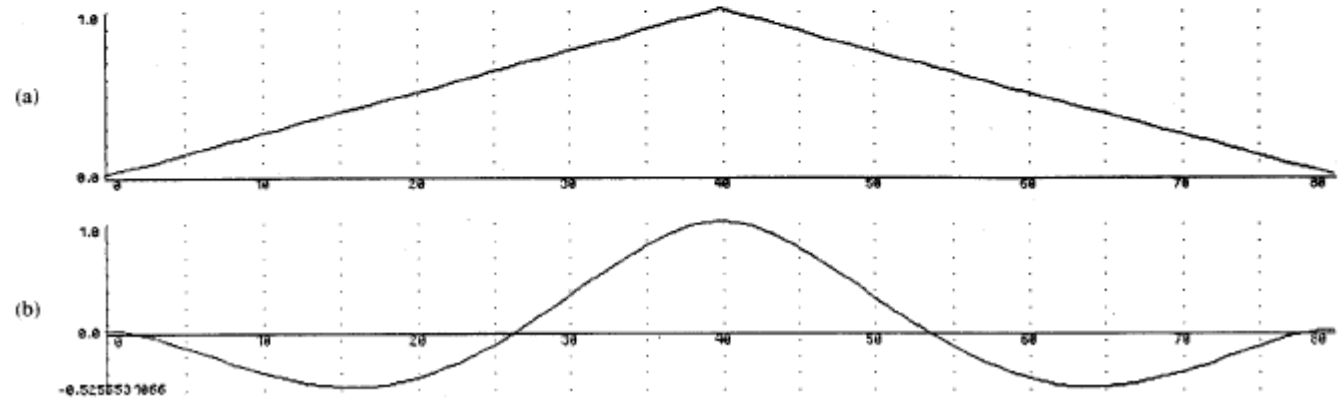
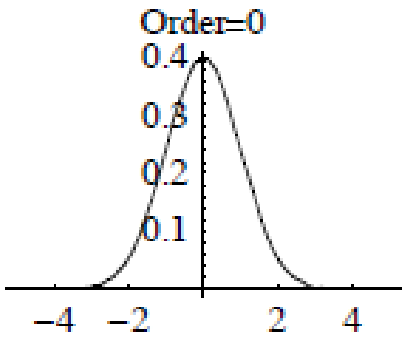
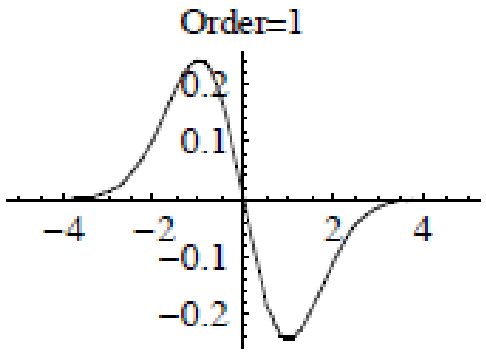


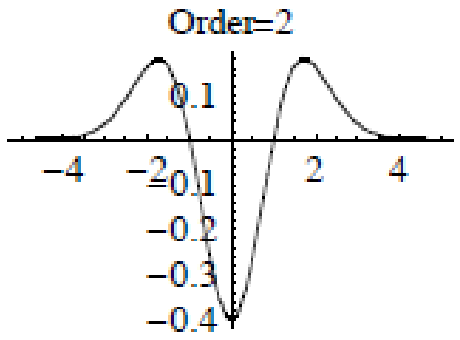
Fig. 3. A roof profile and an optimal operator for roofs.

Resembles 2nd derivative of Gaussian

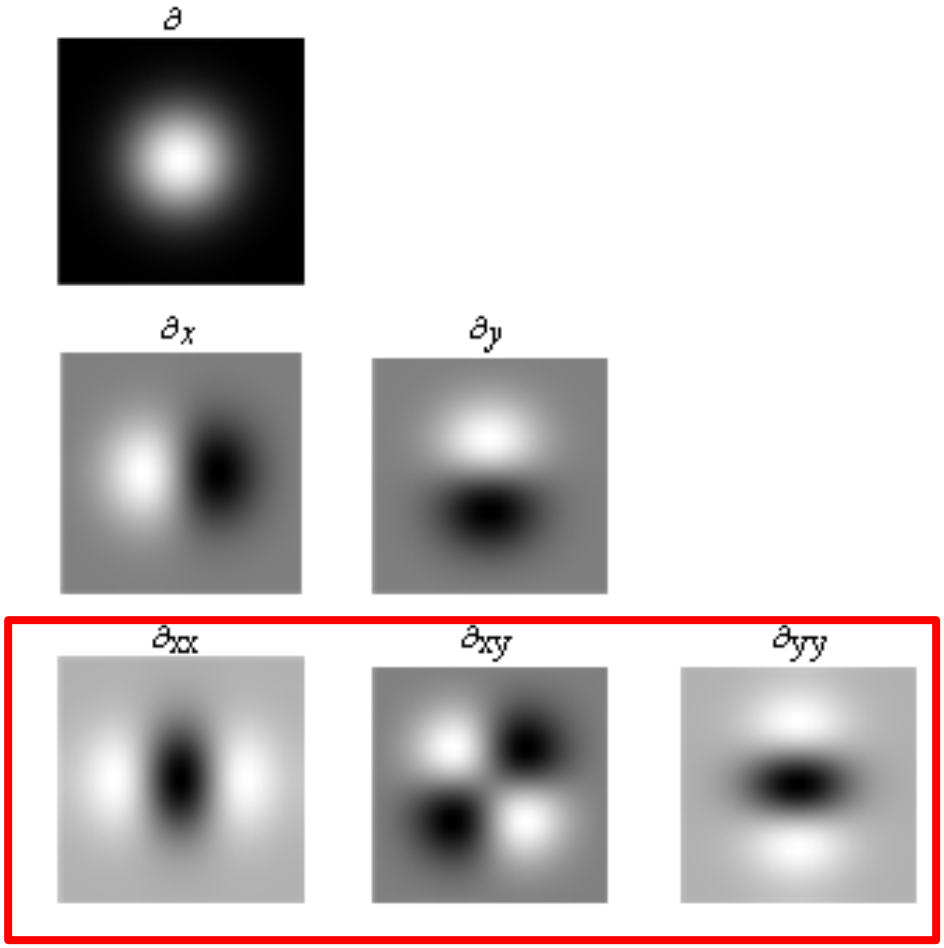
Gaussian Derivatives

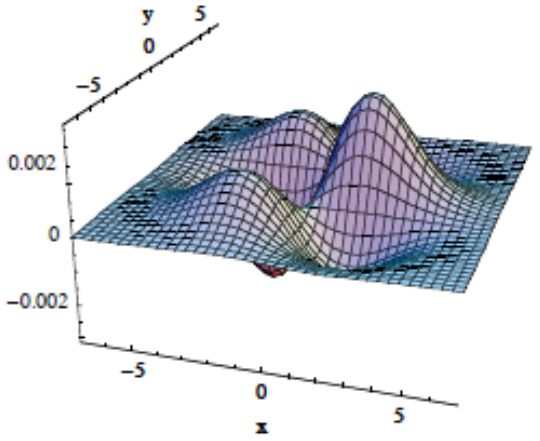






2nd Derivative Operator to detect lines and curves



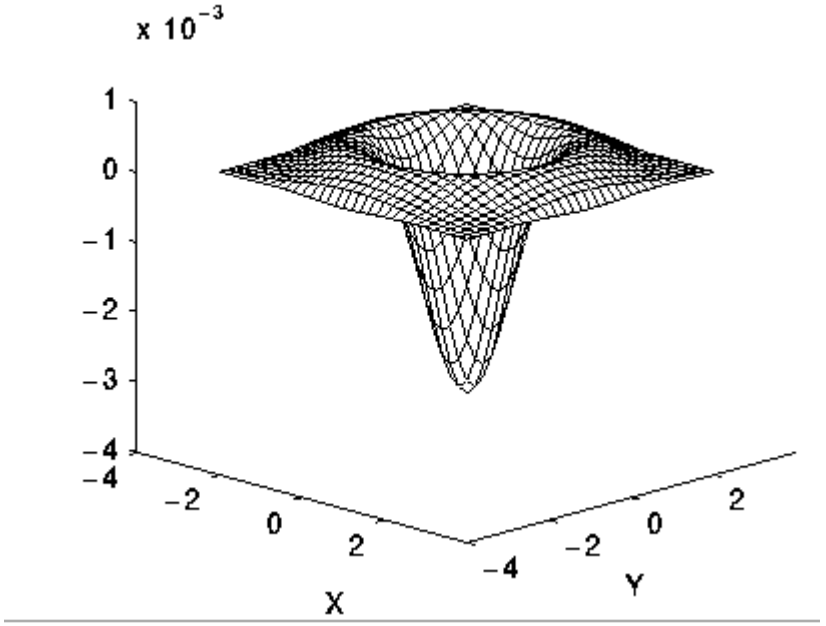


Laplacian

0	−1	0
−1	4	−1
0	−1	0

−1	−1	−1
−1	8	−1
−1	−1	−1

Local kernels



$$LoG(x,y) = -\frac{1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\sigma^2}\right]e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Laplacian of 2D Gaussian kernel

Laplacian of Gaussian (LoG) (Marr-Hildreth operator)

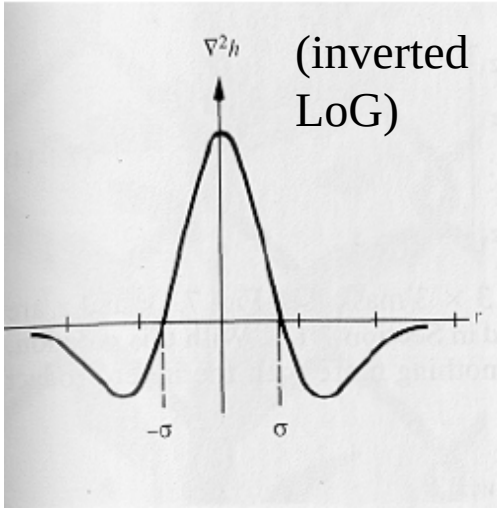
- To reduce the noise effect, the image is first smoothed.
- When the filter chosen is a Gaussian, we call it the LoG edge detector.

$$G(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}}$$

- It can be shown that:

$$\nabla^2[f(x,y) * G(x,y)] = \nabla^2 G(x,y) * f(x,y)$$

$$\nabla^2 G(x,y) = \left(\frac{r^2 - [redacted]}{\sigma^4}\right) e^{-r^2/2\sigma^2}, \; (r^2 = x^2 + y^2)$$



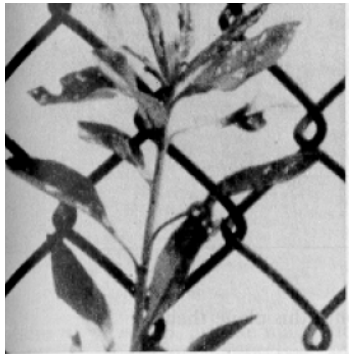
Laplacian of Gaussian (LoG) - Example

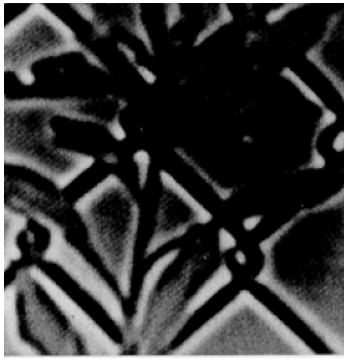
5×5 Laplacian of Gaussian mask

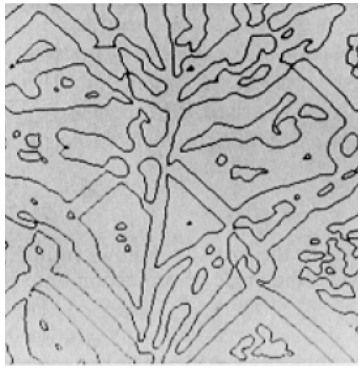
0	0	−1	0	0
0	−1	−2	−1	0
−1	−2	16	−2	−1
0	−1	−2	−1	0
0	0	−1	0	0

17×17 Laplacian of Gaussian mask

0	0	0	0	0	0	−1	−1	−1	−1	−1	0	0	0	0	0	0
0	0	0	0	−1	−1	−1	−1	−1	−1	−1	−1	−1	0	0	0	0
0	0	−1	−1	−1	−2	−3	−3	−3	−3	−3	−2	−1	−1	−1	0	0
0	0	−1	−1	−2	−3	−3	−3	−3	−3	−3	−3	−2	−1	−1	0	0
0	−1	−1	−2	−3	−3	−3	−2	−3	−2	−3	−3	−3	−2	−1	−1	0
0	−1	−2	−3	−3	−3	0	2	4	2	0	−3	−3	−3	−2	−1	0
−1	−1	−3	−3	−3	0	4	10	12	10	4	0	−3	−3	−3	−1	−1
−1	−1	−3	−3	−2	2	10	18	21	18	10	2	−2	−3	−3	−1	−1
−1	−1	−3	−3	−3	4	12	21	24	21	12	4	−3	−3	−3	−1	−1
−1	−1	−3	−3	−2	2	10	18	21	18	10	2	−2	−3	−3	−1	−1
−1	−1	−3	−3	−3	0	4	10	12	10	4	0	−3	−3	−3	−1	−1
0	−1	−2	−3	−3	−3	0	2	4	2	0	−3	−3	−3	−2	−1	0
0	−1	−1	−2	−3	−3	−3	−2	−3	−2	−3	−3	−3	−2	−1	−1	0
0	0	−1	−1	−2	−3	−3	−3	−3	−3	−3	−3	−2	−1	−1	0	0
0	0	−1	−1	−1	−2	−3	−3	−3	−3	−3	−2	−1	−1	−1	0	0
0	0	0	0	−1	−1	−1	−1	−1	−1	−1	−1	−1	0	0	0	0
0	0	0	0	0	0	−1	−1	−1	−1	−1	0	0	0	0	0	0







Edge detector – 1st and 2nd Derivatives

original image

1st deriv. Gaussian

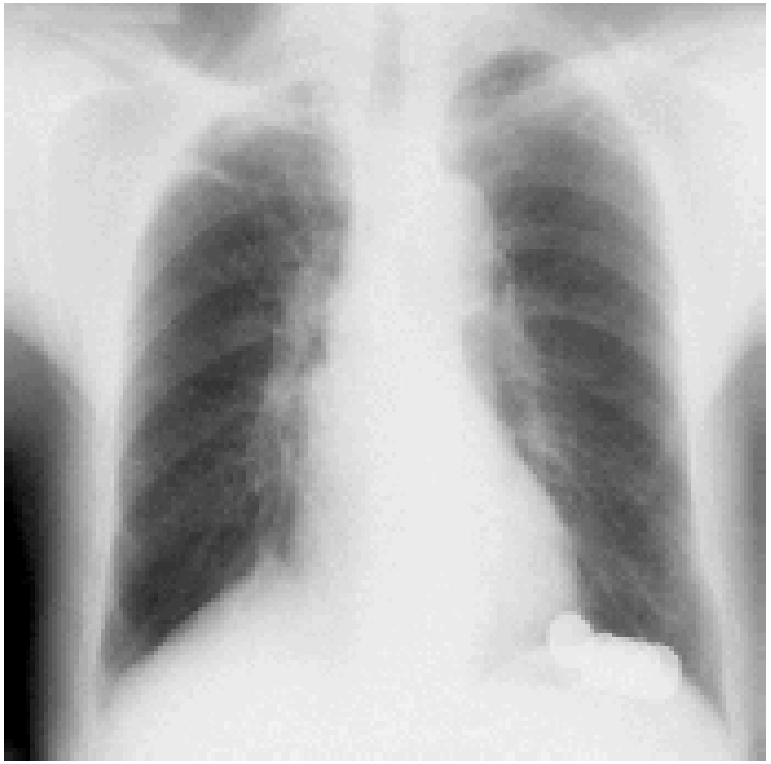
LoG

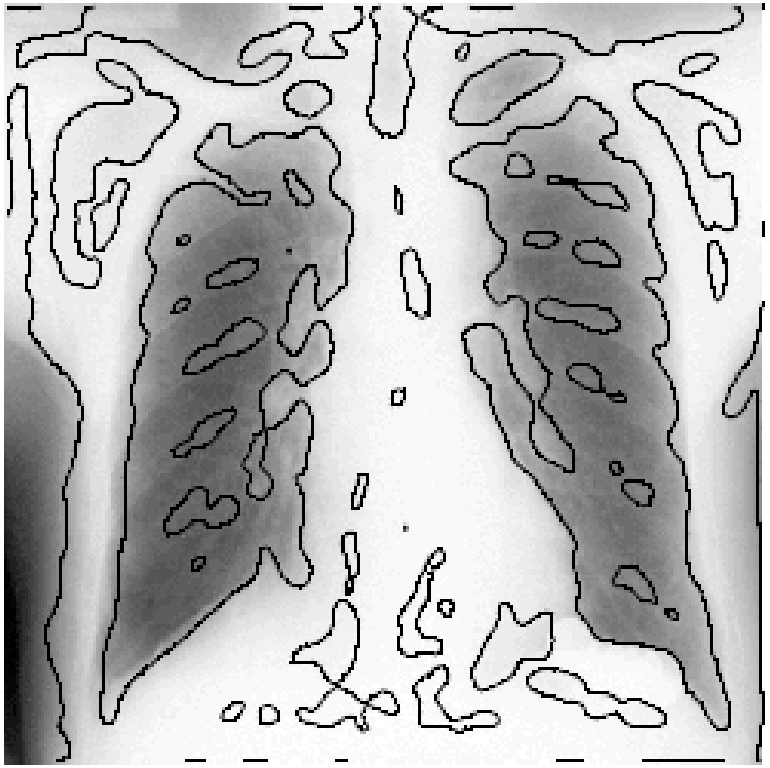
Laplacian of Gaussian (LoG)

Enhances line-like structures (glasses), creates zero-crossing at edges (positive and negative response at both sides of edges)

Source: http://homepages.inf.ed.ac.uk/rbf/HIPR2/log.htm

Often used: Zero-Crossings of LoG for Edge Detection





$\Delta L = 0 \qquad \sigma = 4$ pixels

Hint: Remember that edge positions are extrema of first derivative → zero-crossings of 2nd derivatives. Be careful: Extrema or maxima & minima!

Line/Ridge Detection