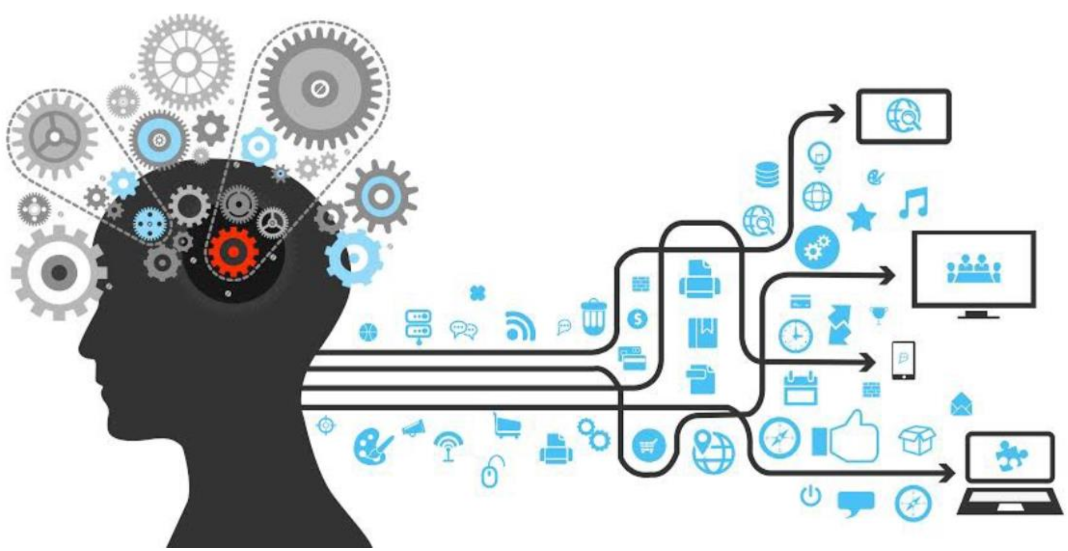
Computer Vision

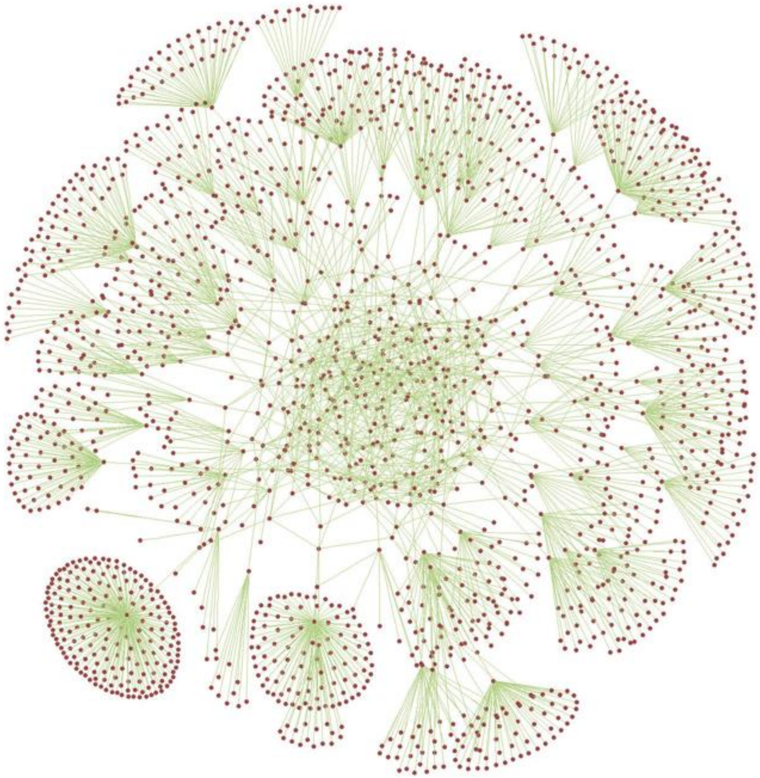
Early vision: Just one image

School of Electronic &
Electrical Engineering

Sungkyunkwan University

Hyunjin Park

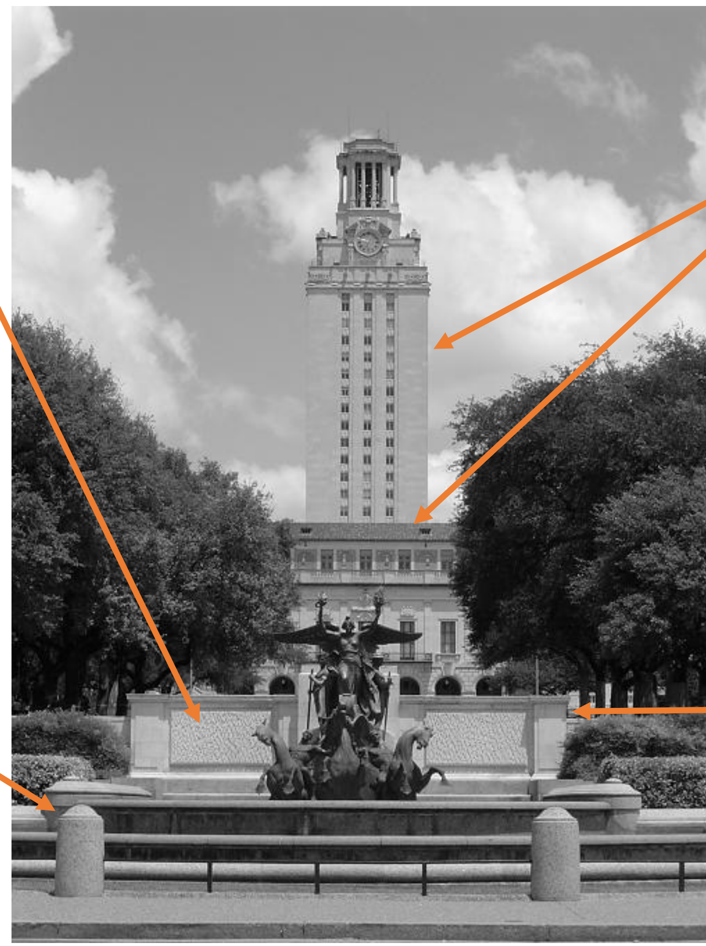




Local Image Features

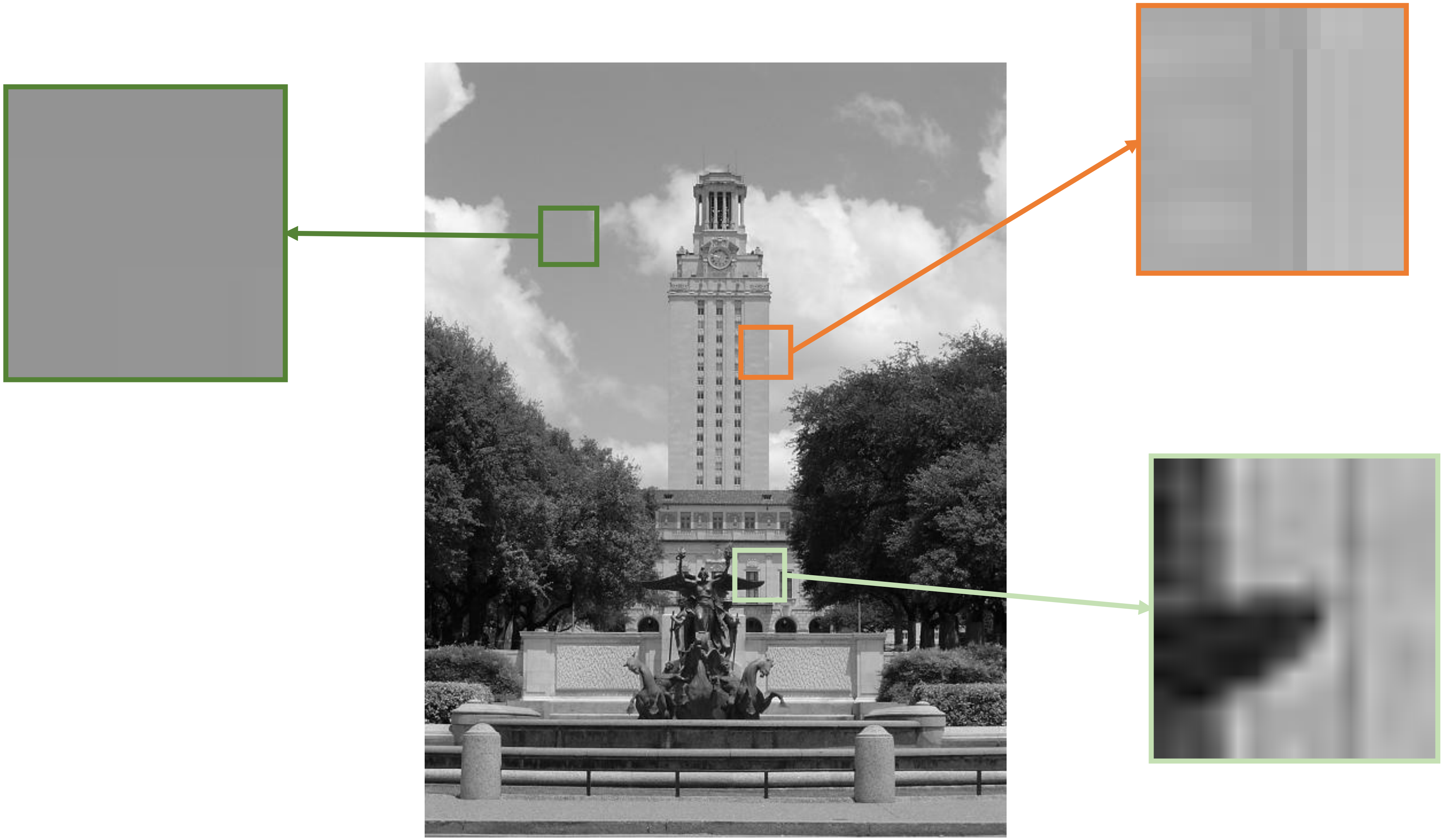
Edges

- Points of discontinuity or sharp change in an image



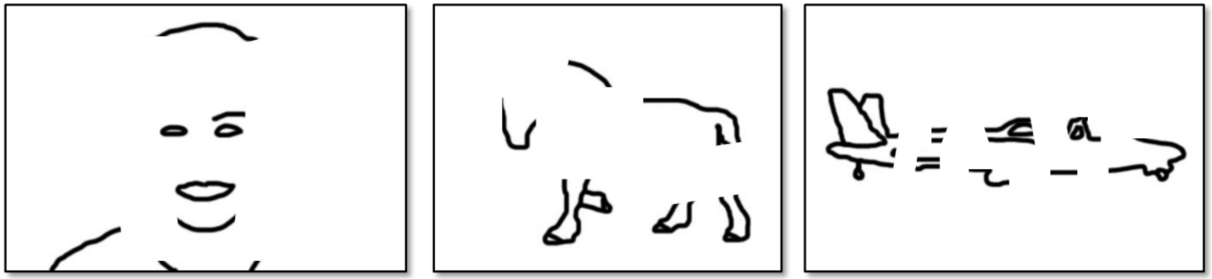
Edges

- Contrast and invariance



Edge detection

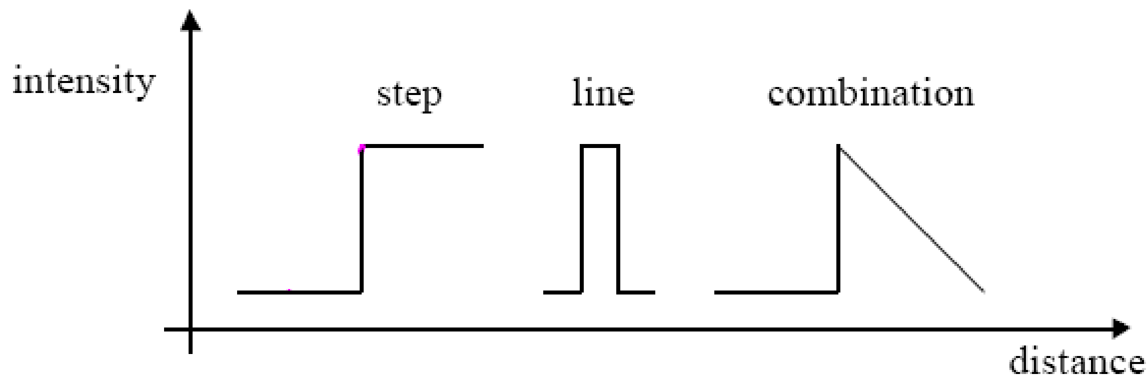
- Edge detection
  - Map image from 2D pixel arrays to a set of curves or line ***segments or contours***



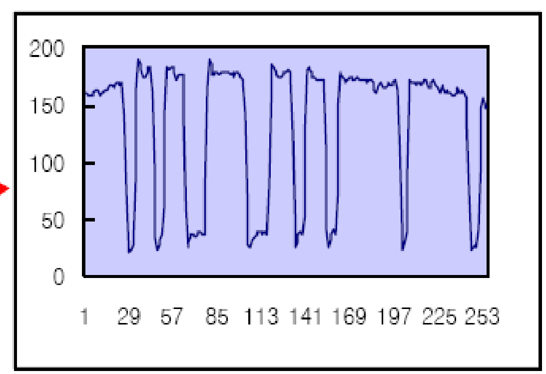
- General strategy
  - Determine image gradient
  - Mark points where gradient magnitude is particularly large with respect to neighbors

Edge detection

- Ideal edge function



The profile of an ideal step change in image intensity



Steps in edge detection

- ***Filtering***
  - To improve the performance of an edge detector with respect to noise
  - Tradeoff between edge strength and noise reduction

- ***Enhancement***
  - Emphasizes pixel where there is a significant change in local intensity value

- ***Detection***
  - Find strong edge content

Steps in edge detection

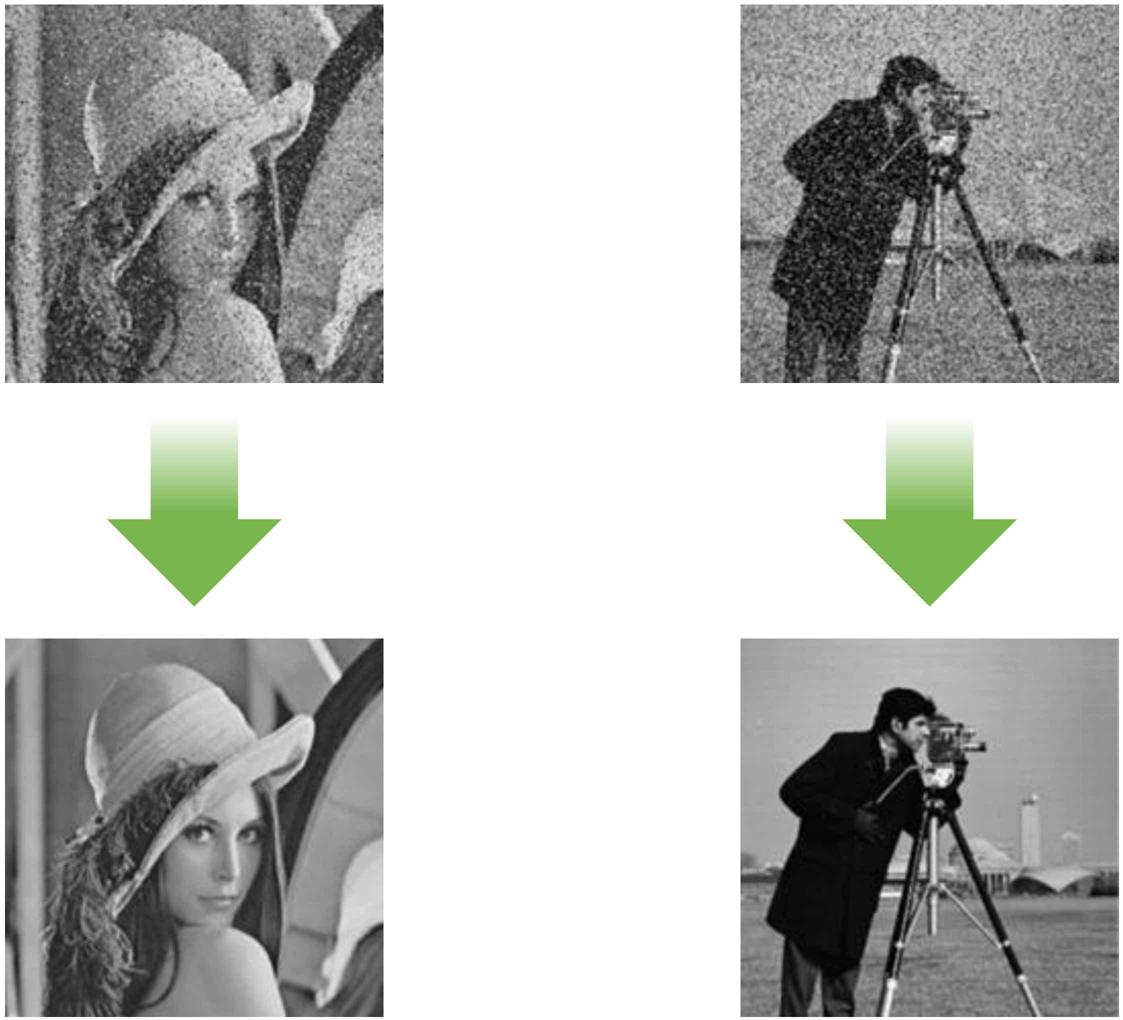
- ***Filtering***
  - To improve the performance of an edge detector with respect to noise
  - Tradeoff between edge strength and noise reduction

- Enhancement
  - Emphasizes pixel where there is a significant change in local intensity value

- Detection
  - Find strong edge content

Filtering

- Noise reduction



Steps in edge detection

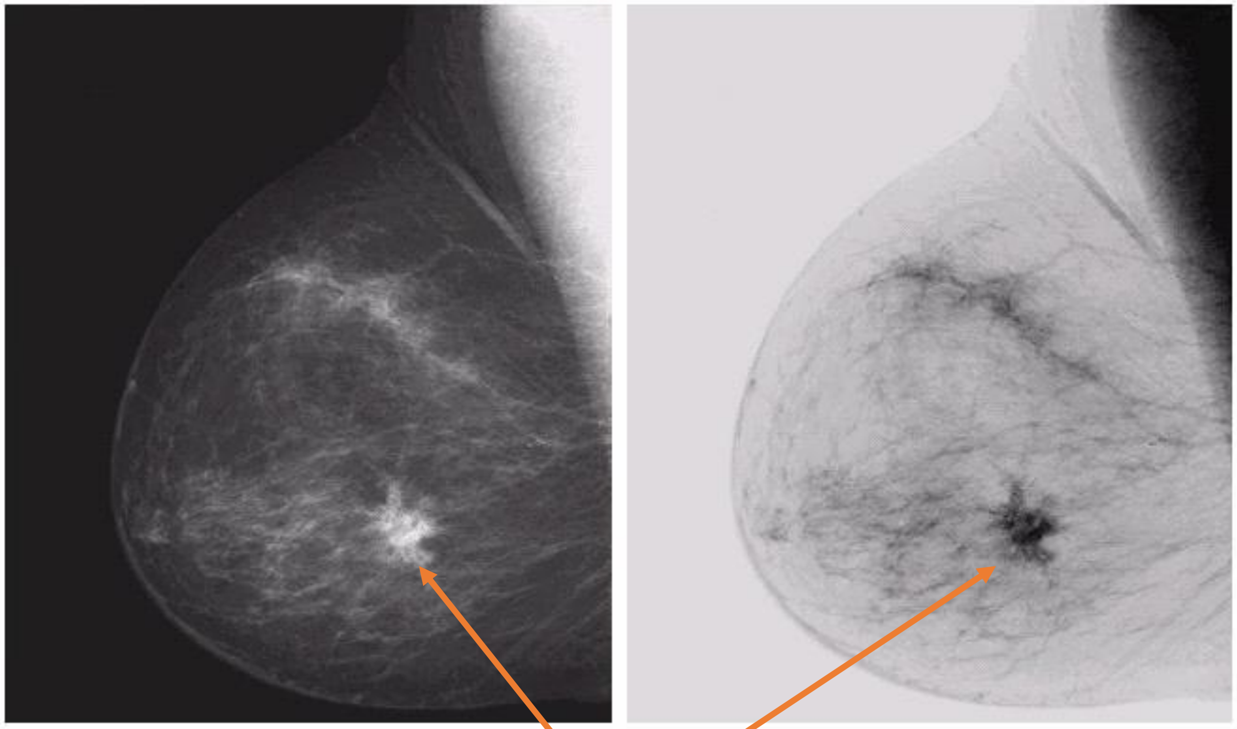
- Filtering
  - To improve the performance of an edge detector with respect to noise
  - Tradeoff between edge strength and noise reduction

- ***Enhancement***
  - Emphasizes pixel where there is a significant change in local intensity value

- Detection
  - Find strong edge content

Enhancement

- Image negatives

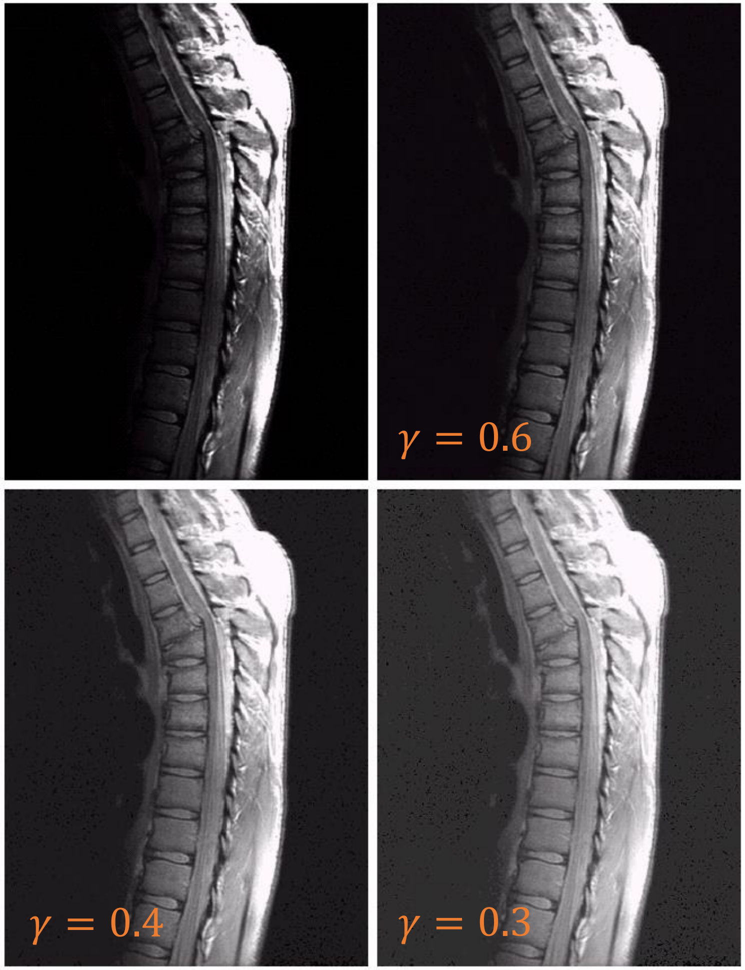


Easier to analyze the specific cancer

Enhancement

- Power-law transformations
  - $s = cr^{\gamma}$, c and $\gamma > 0$

Too dark →

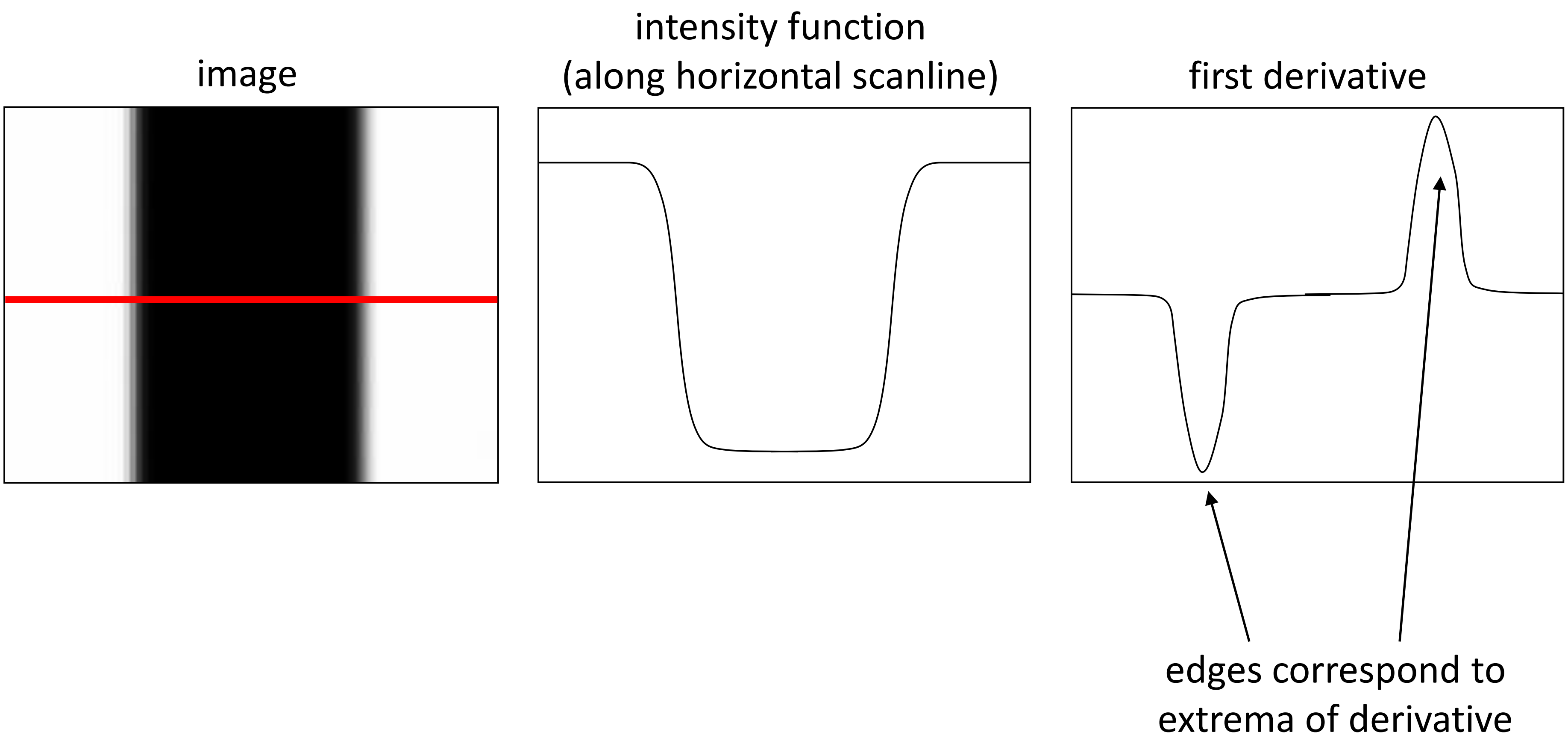


Steps in edge detection

- Filtering
  - To improve the performance of an edge detector with respect to noise
  - Tradeoff between edge strength and noise reduction
- Enhancement
  - Emphasizes pixel where there is a significant change in local intensity value
- ***Detection***
  - Find strong edge content

Derivatives and edges

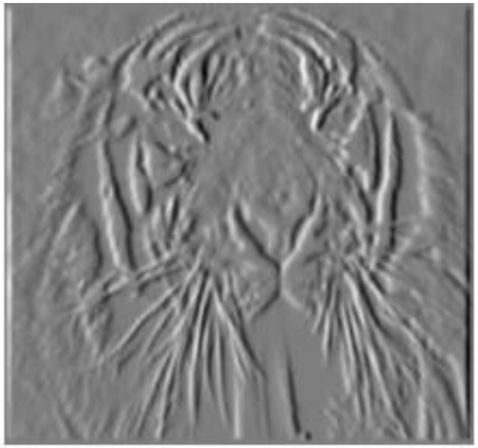
- An edge is a place of rapid change in the image intensity function

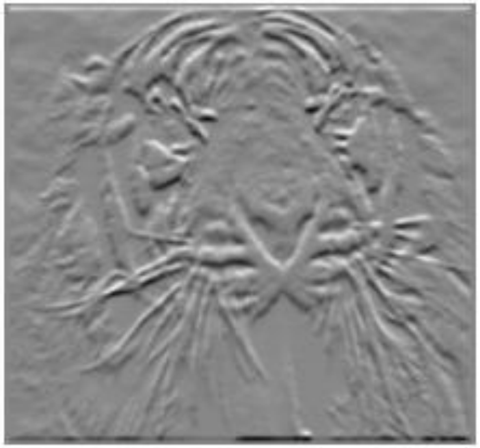


Derivatives and edges

- Partial derivatives of an image

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





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

-1	1

-f(x-1,y) + f(x,y)

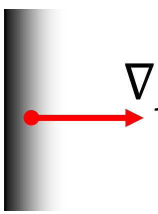
-1
1

Derivatives and edges

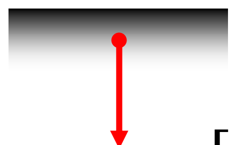
- Image gradient

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

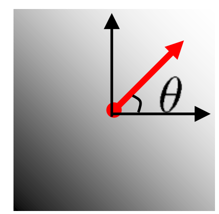
- Gradient points in the direction of most rapid change in intensity



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



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



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

Derivatives and edges

- Gradient direction (orientation of edge normal)

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

- Edge strength is given by the gradient magnitude

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

Gradient image

- The gradient magnitude can be estimated by smoothing an image and then differentiating it

Difference operators

- 2D difference operators

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

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

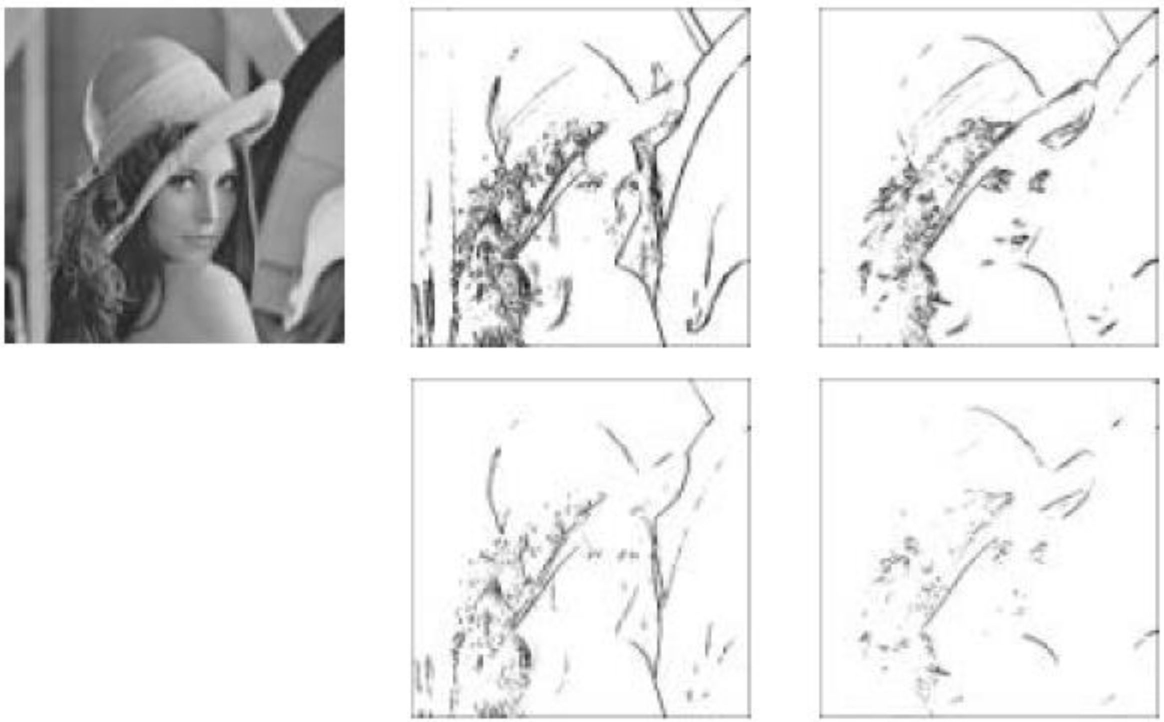
Roberts: $G_x = \begin{array}{|c|c|}\hline 0 & 1 \\ \hline -1 & 0 \\ \hline\end{array}$; $G_y = \begin{array}{|c|c|}\hline 1 & 0 \\ \hline 0 & -1 \\ \hline\end{array}$

Difference operators

- Prewitt operator

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \qquad G_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

Original image

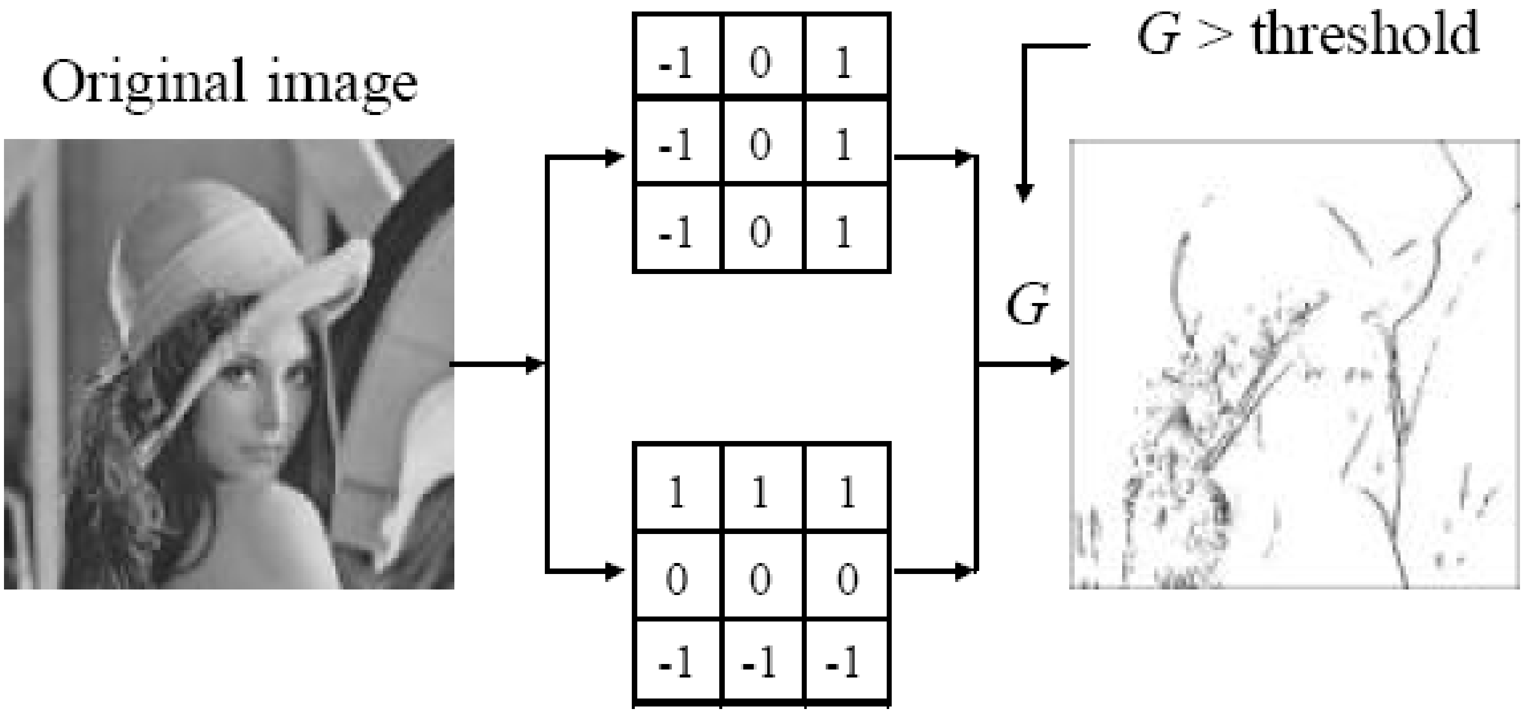


threshold = 70

threshold = 130

Difference operators

- Prewitt operator



Result of Prewitt operator (threshold = 130)

Difference operators

- Sobel operator

-1	0	1
-2	0	2
-1	0	1

G_x

1	2	1
0	0	0
-1	-2	-1

G_y

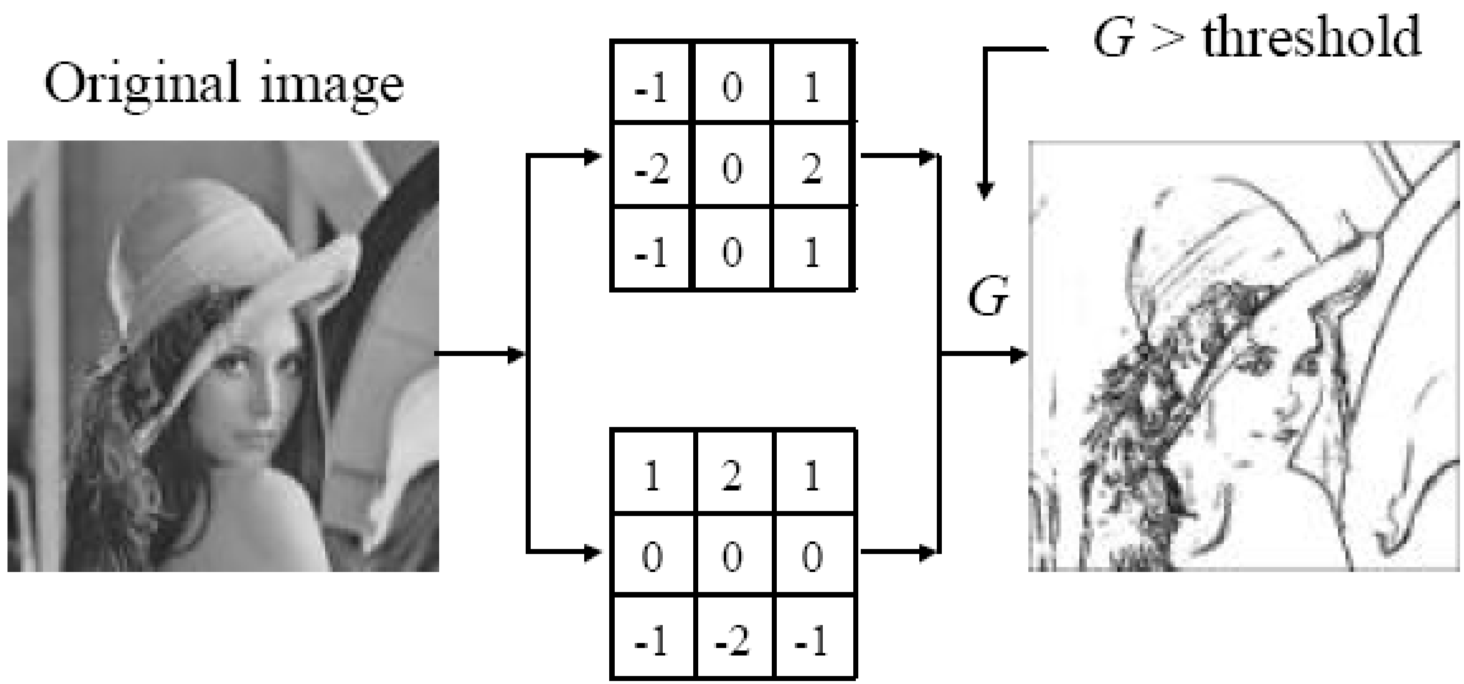
Original image

threshold = 70

threshold = 130

Difference operators

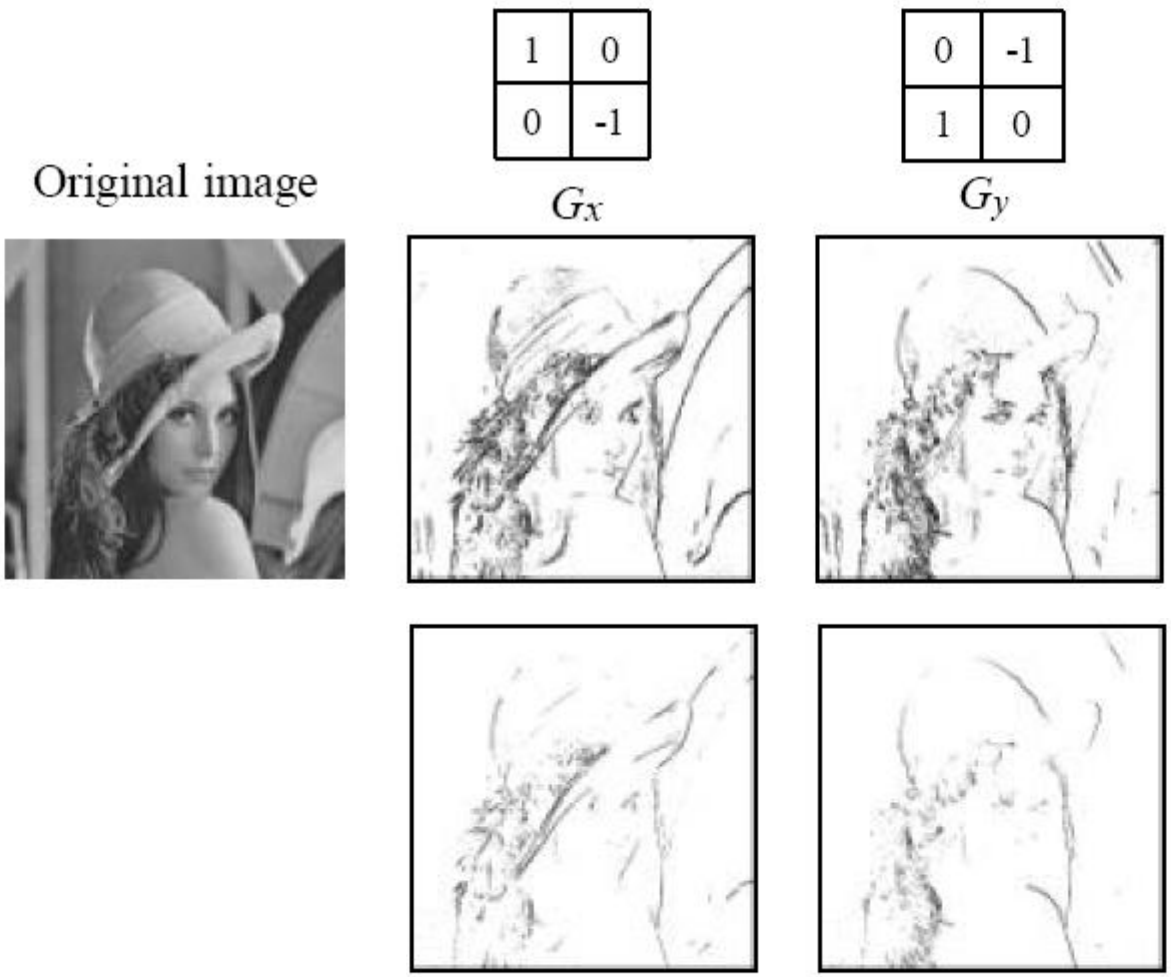
- Sobel operator



Result of Sobel operator (threshold = 130)

Difference operators

- Robert’s cross operator

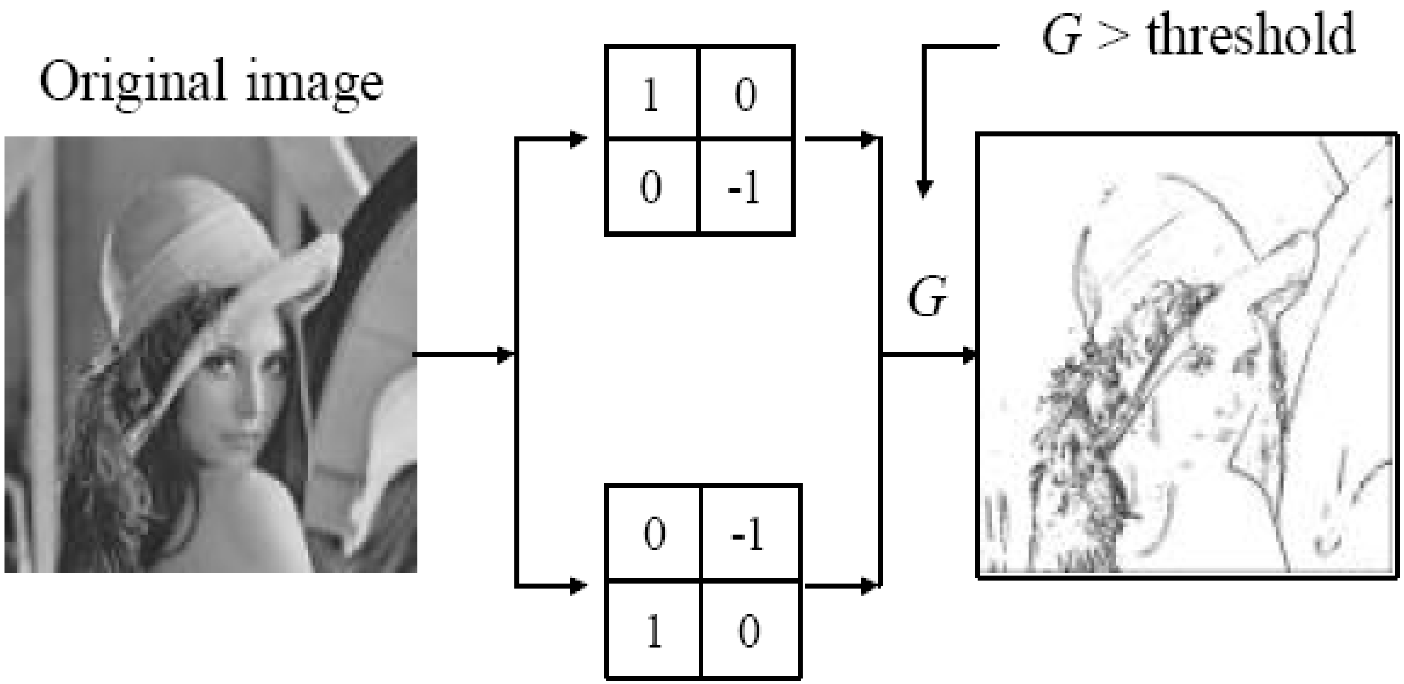


threshold = 20

threshold = 40

Difference operators

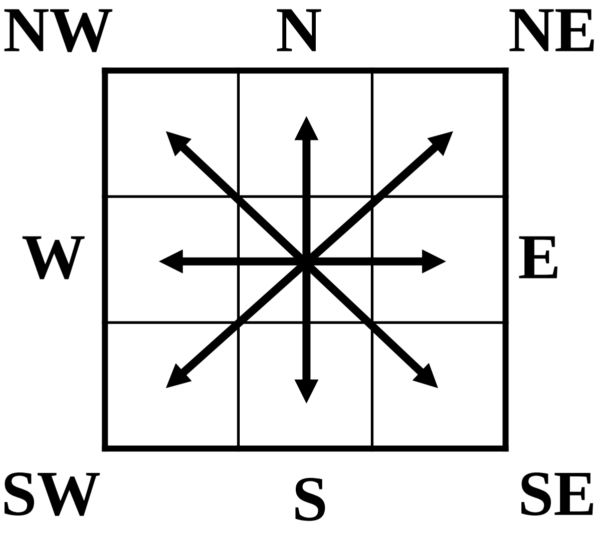
- Robert's cross operator



Result of Roberts cross operator (threshold = 40)

Compass gradient masks

- Basic idea
  - Use 8 masks aligned with the usual compass directions
  - Select largest response (magnitude)
  - Orientation is the direction associated with the largest response
- Gradient magnitude is given by the maximum response
- Gradient direction: Compass direction of the maximum response



Compass gradient masks

N

NW

2	1	0
1	0	-1
0	-1	-2

N

1	2	1
0	0	0
-1	-2	-1

NE

0	1	2
-1	0	1
-2	-1	0

W

1	0	-1
2	0	-2
1	0	-1

E

-1	0	1
-2	0	2
-1	0	1

SW

0	-1	-2
1	0	-1
2	1	0

S

-1	-2	-1
0	0	0
1	2	1

SE

-2	-1	0
-1	0	1
0	1	2

S

Canny edge detector

- Optimal kernel
  - Justified use of Gaussian
- Non-maximum suppression
  - Remove edges orthogonal to a maxima
- Linking and thresholding (hysteresis)
  - Improved recovery of long image contours
  - Use the high threshold to start edge curves and the low threshold to continue them

Canny edge detector

- Original image

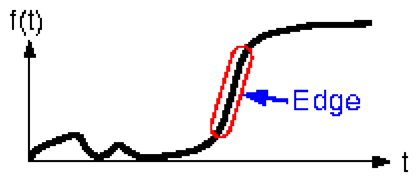
Canny edge detector

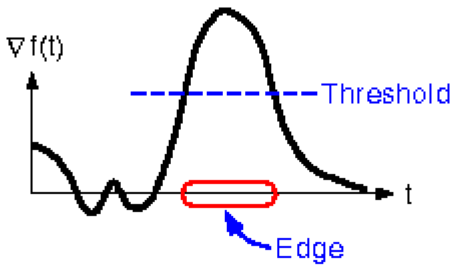
- Norm of the gradient

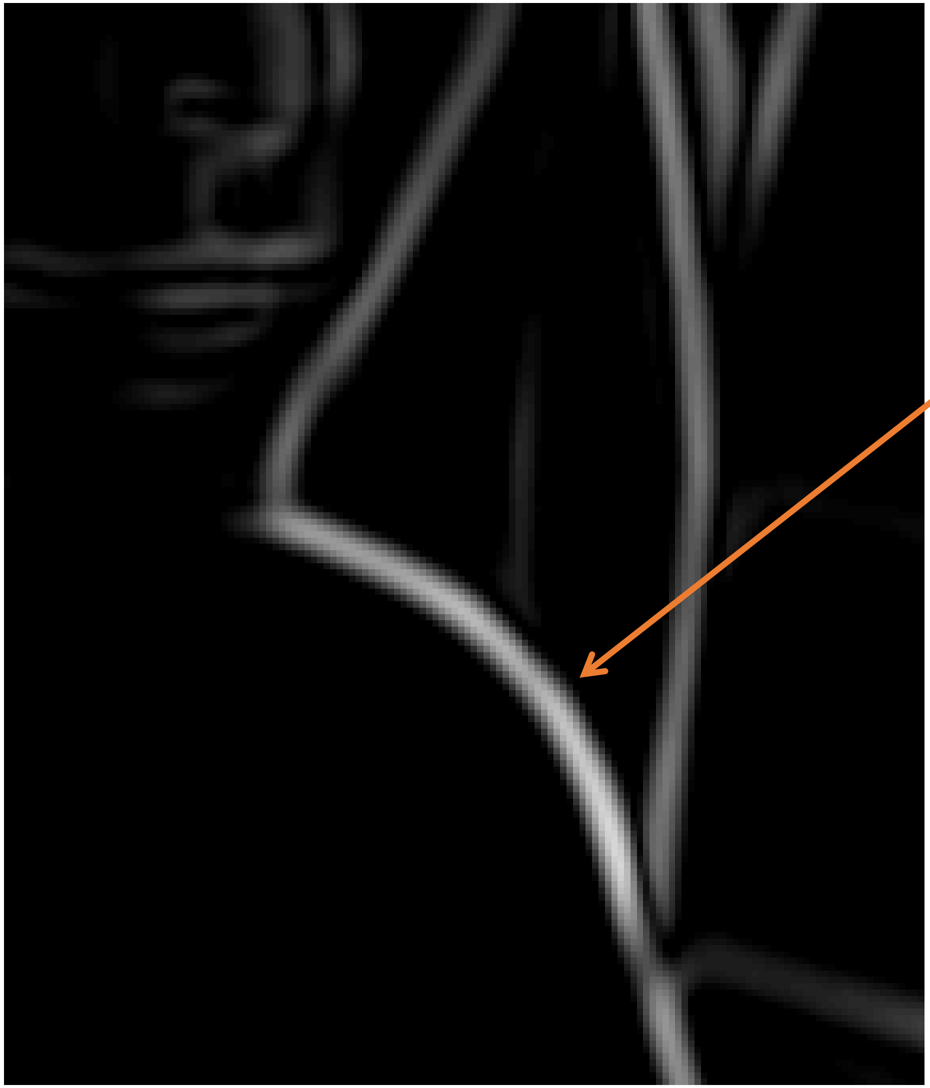
Canny edge detector

- Thresholding

Canny edge detector



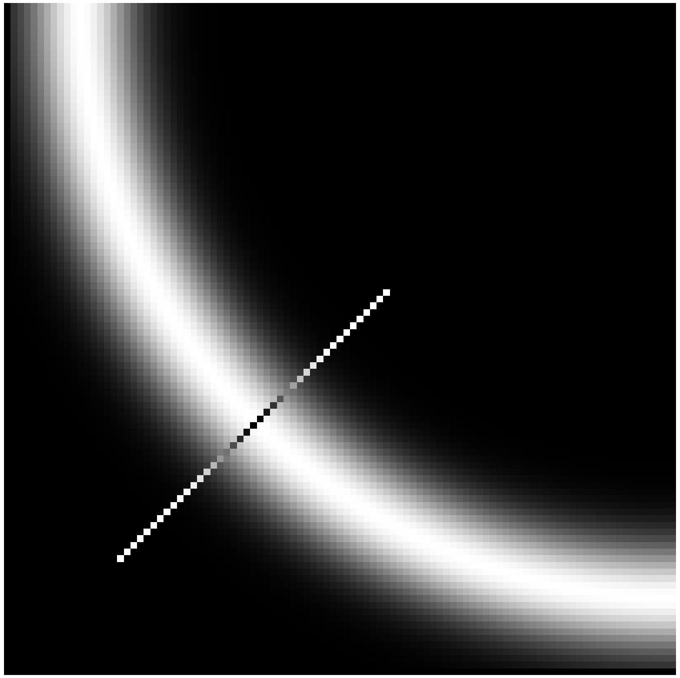


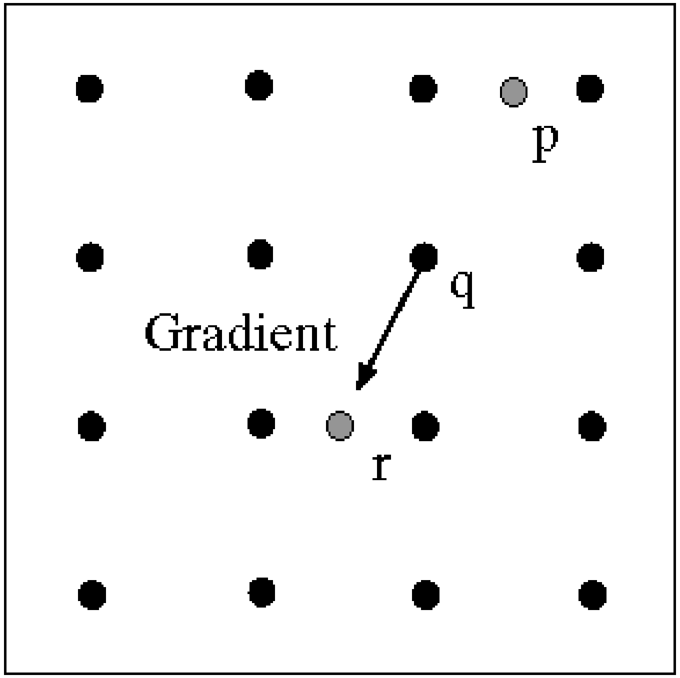


How to turn these thick regions of the gradient into curves?

Canny edge detector

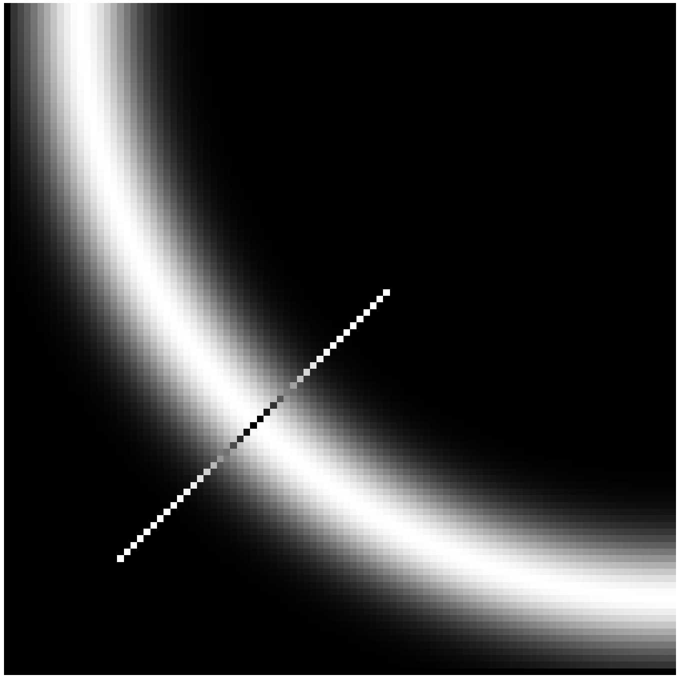
- Non-maximum suppression
  - Check if a pixel is local maximum along gradient direction
  - At q, we have a maximum if the value is larger than those at both p and at r

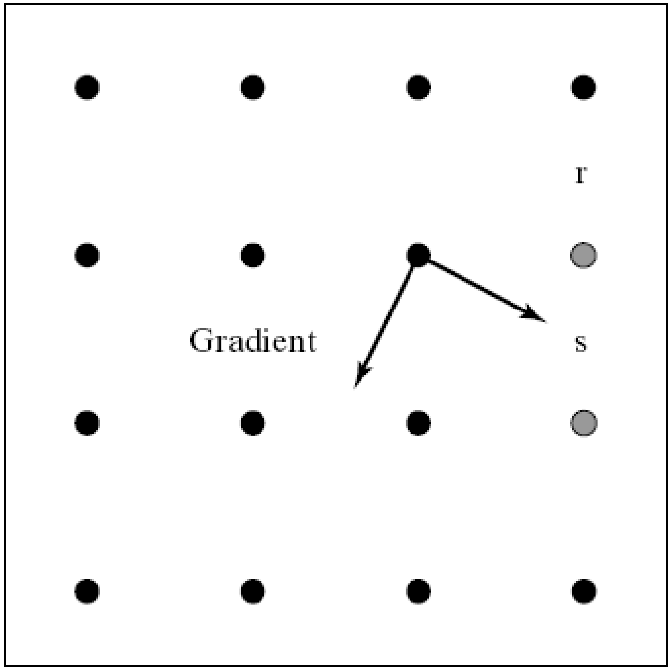




Canny edge detector

- Non-maximum suppression
  - Assume the marked point is an edge point
  - Then, construct the tangent to the edge curve and use this to predict the next points (here, either r or s)



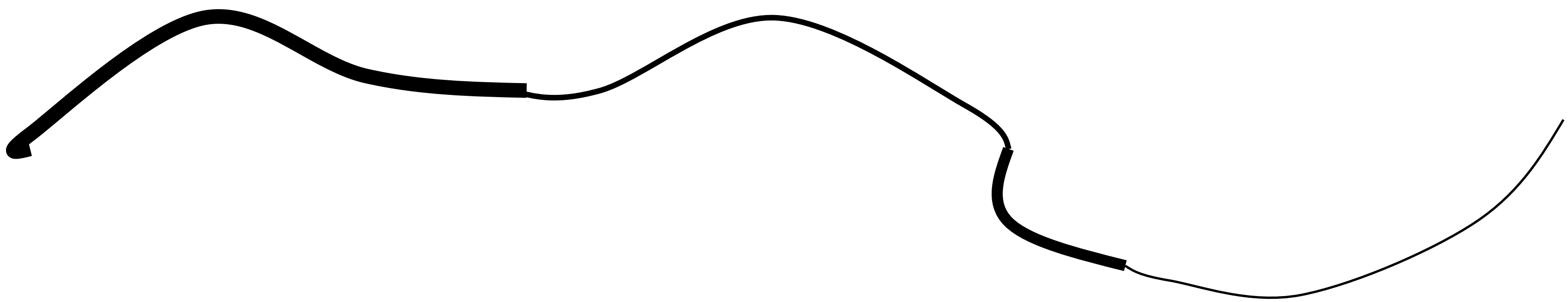


Canny edge detector

- Problem
  - Pixels along some edges did not survive the thresholding

Canny edge detector

- Hysteresis thresholding
  - Check that maximum value of gradient value is sufficiently large
  - Use a high threshold to start edge curves and a low threshold to continue them



Canny edge detector

- Hysteresis thresholding

original image

high threshold
(strong edges)

low threshold
(weak edges)

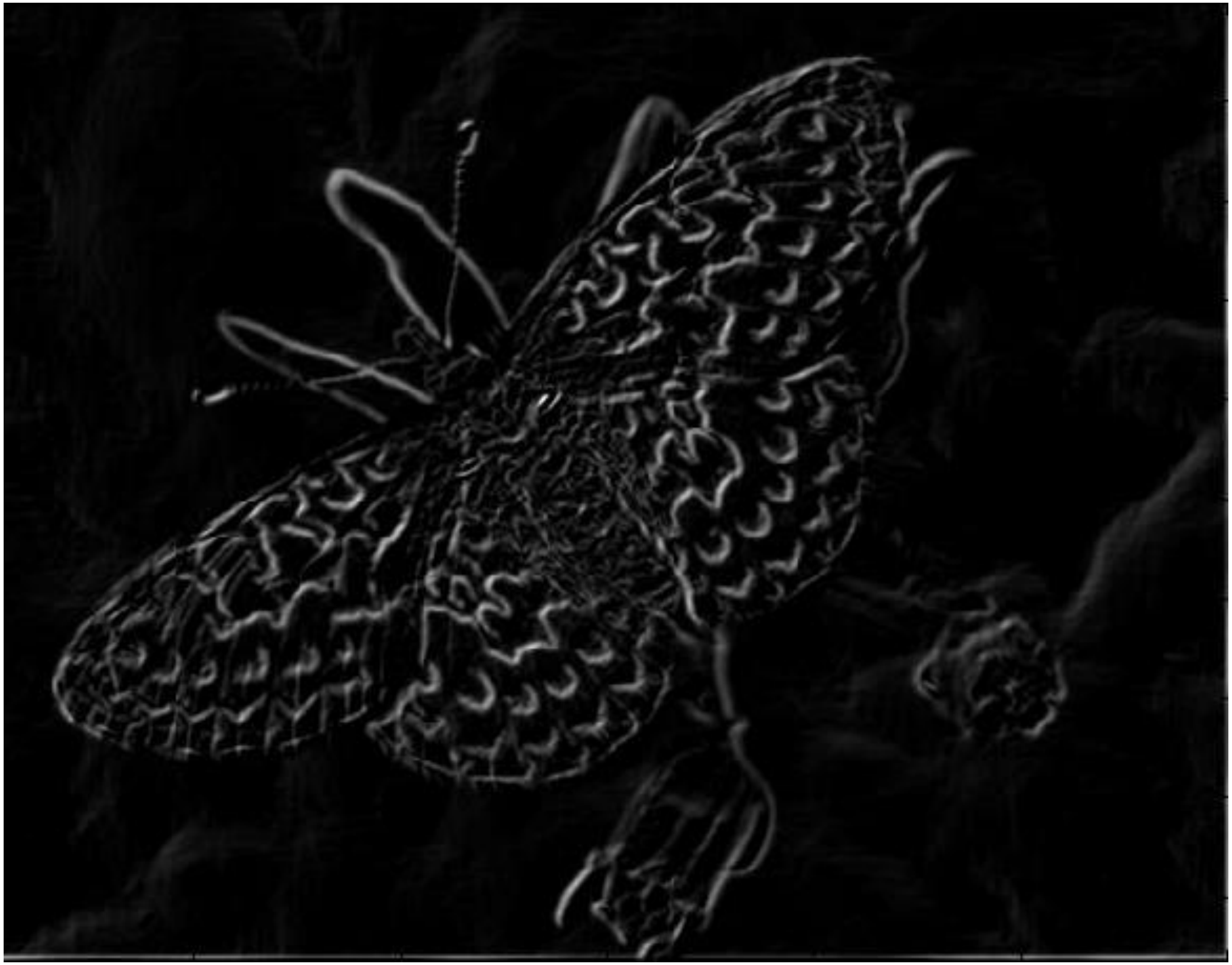
hysteresis threshold

Thresholding

- Original image

Thresholding

- Gradient magnitude image



Thresholding

- Thresholding gradient with a ***lower threshold***

Thresholding

- Thresholding gradient with a ***higher threshold***

LoG

- Thresholding gradient images
  - Too many edge points
- Edges from derivatives
  - Peak in the first derivative
  - Zero-crossing in the second derivative

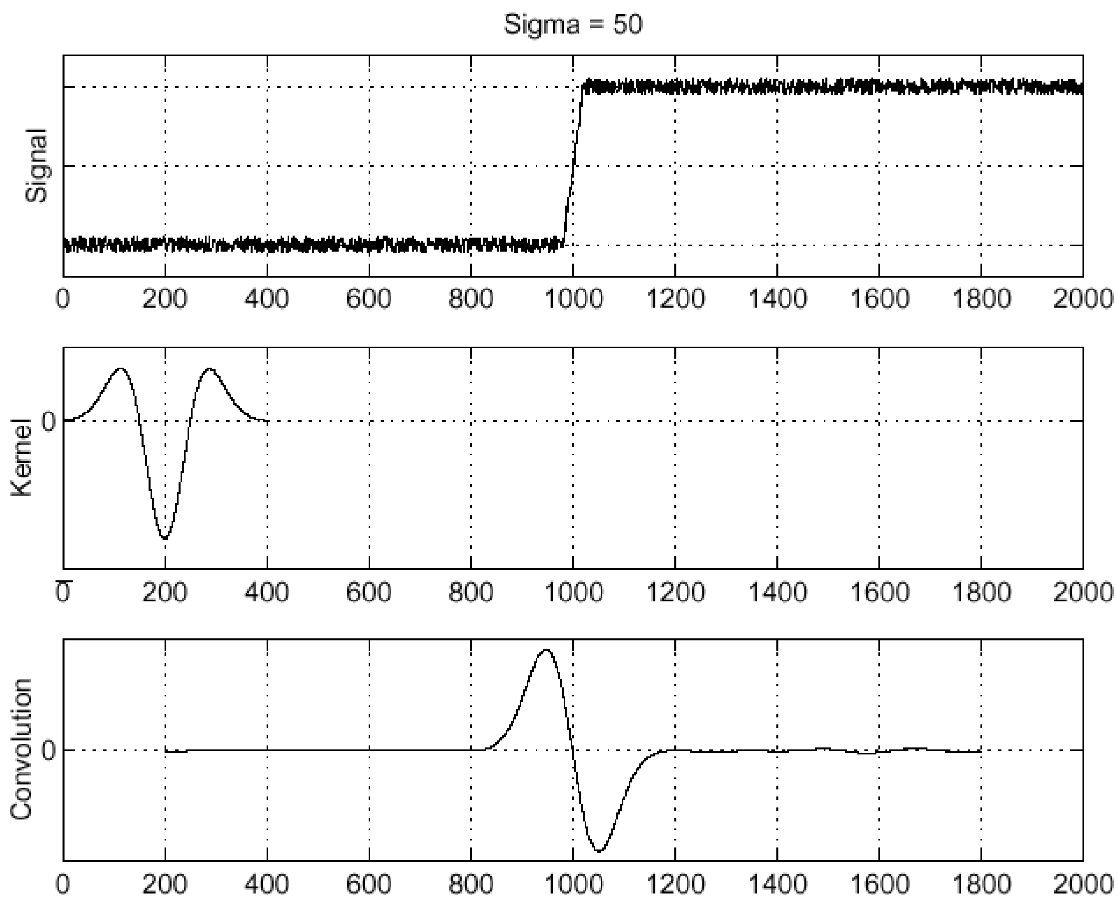
LoG

- Laplacian of Gaussian (LoG): ***Second derivative operator***
  - Consider $\frac{\partial^2}{\partial x^2}(h * f)$

f

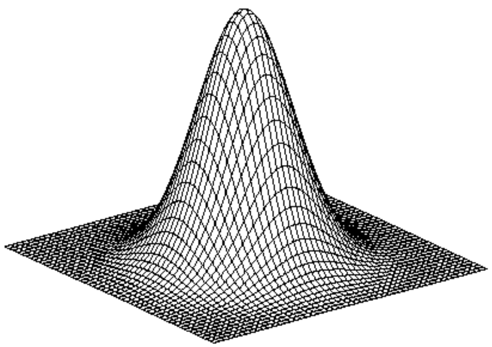
$\frac{\partial^2}{\partial x^2} h$

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



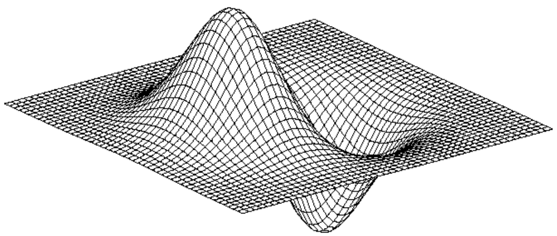
LoG

- Laplacian of Gaussian (LoG): ***Second derivative operator***
  - Presence of a ***zero crossing*** in the ***second derivative*** with a corresponding ***large peak*** in the ***first derivative***



Gaussian

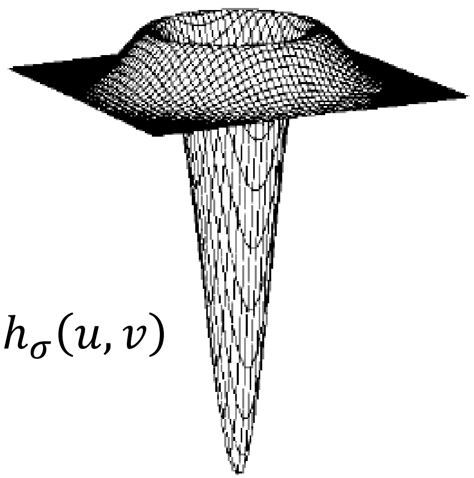
$$h_\sigma(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$



derivative of Gaussian

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

Laplacian of Gaussian



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

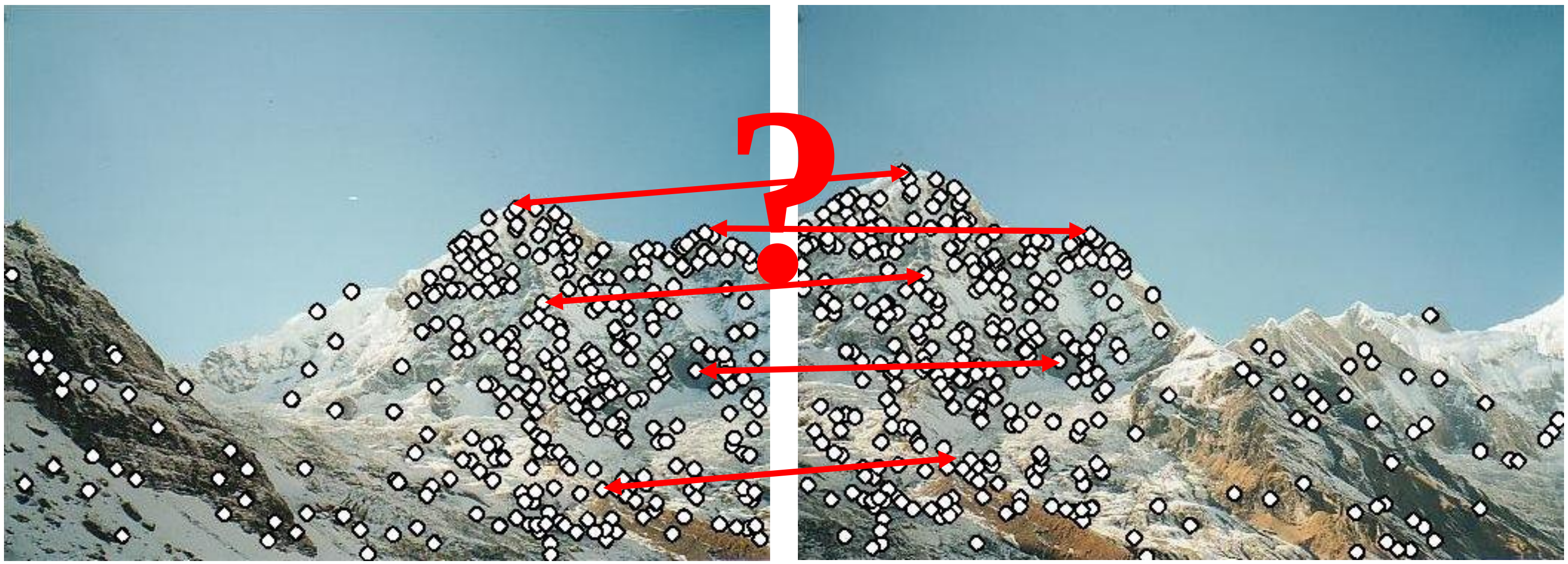
$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

LoG

- Edge detection with LoG
  - Smoothing filter is Gaussian
  - Enhancement step is the second derivative (2D Laplacian)
  - Detection criterion
    - Presence of a zero crossing in the second derivative with a corresponding large peak in the first derivative
  - Edge location can be estimated with subpixel resolution using linear interpolation

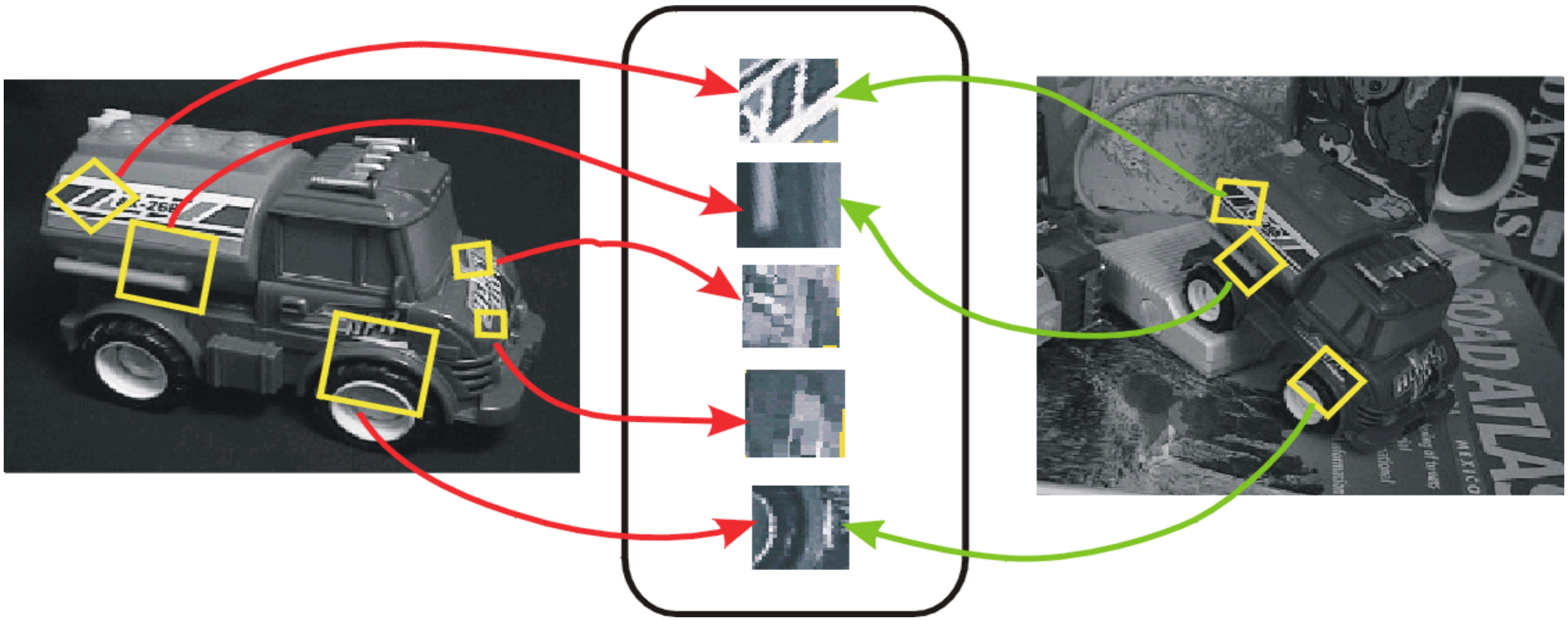
Describe a local region

- We know how to detect points
- Next question
  - How to describe them for matching?
- Point descriptor should be
  - Invariant
  - Distinctive



SIFT features

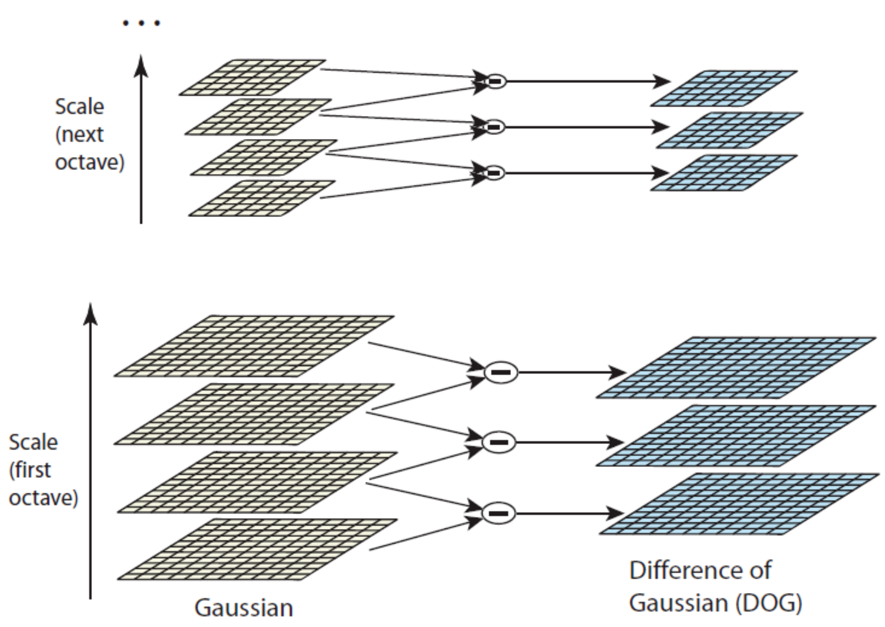
- SIFT (Scale Invariant Feature Transform)
  - 1) Keypoint detection
  - 2) Keypoint description

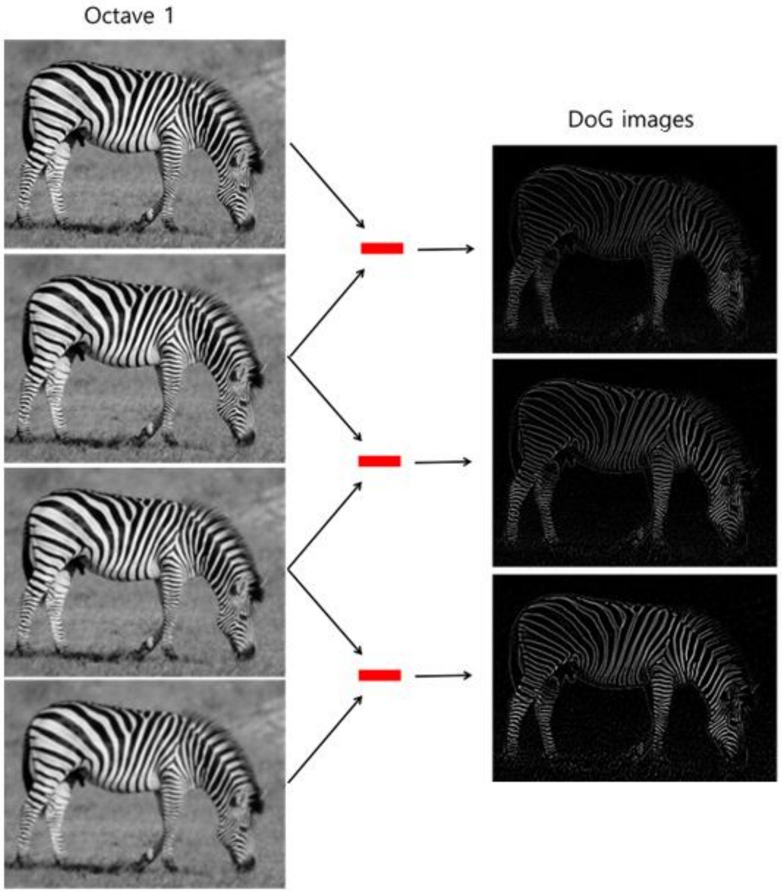


SIFT Features

SIFT features

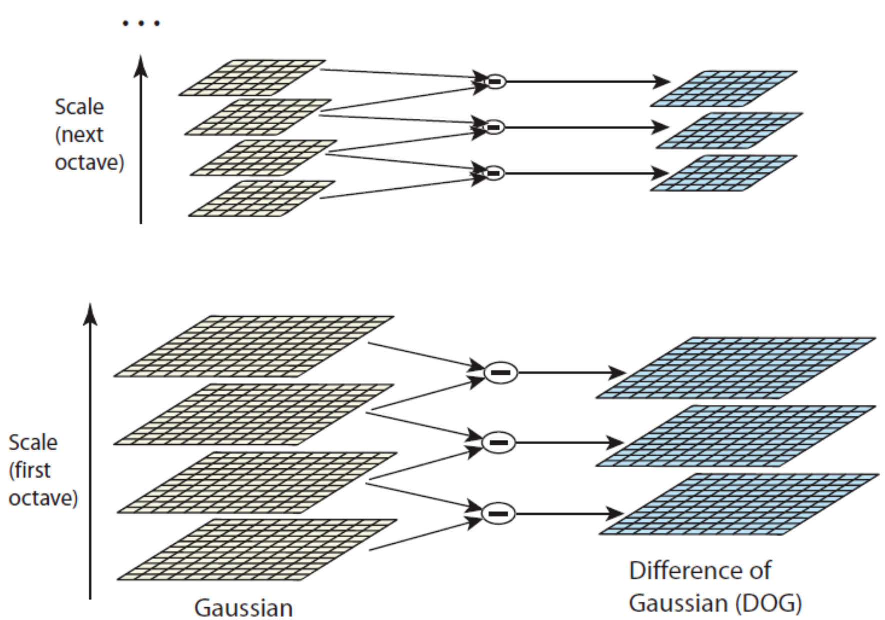
- SIFT (Scale Invariant Feature Transform)
  - 1) Keypoint detection
    - Find keypoints (extrema) using DoG (approximated LoG)

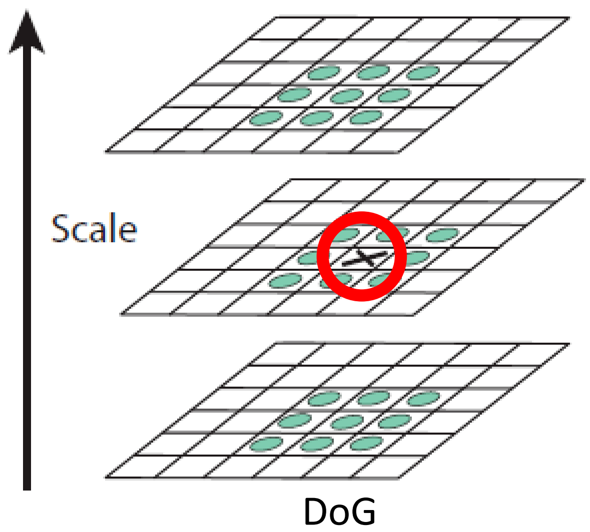




SIFT features

- SIFT (Scale Invariant Feature Transform)
  - 1) Keypoint detection
    - Find keypoints (extrema) using DoG

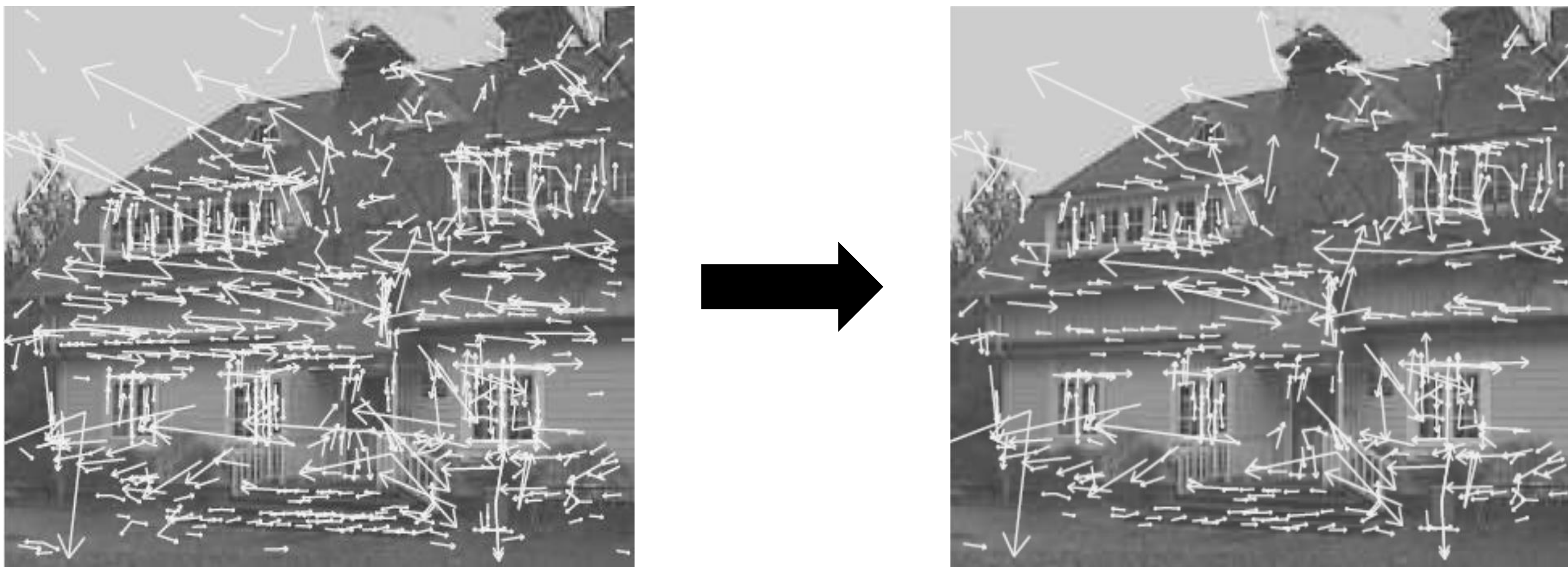




if pixel x's value is higher (or lower) than 26 pixels around it, the pixel x can be named as a keypoint

SIFT features

- SIFT (Scale Invariant Feature Transform)
  - 1) Keypoint detection
    - Remove inappropriate keypoints (threshold)

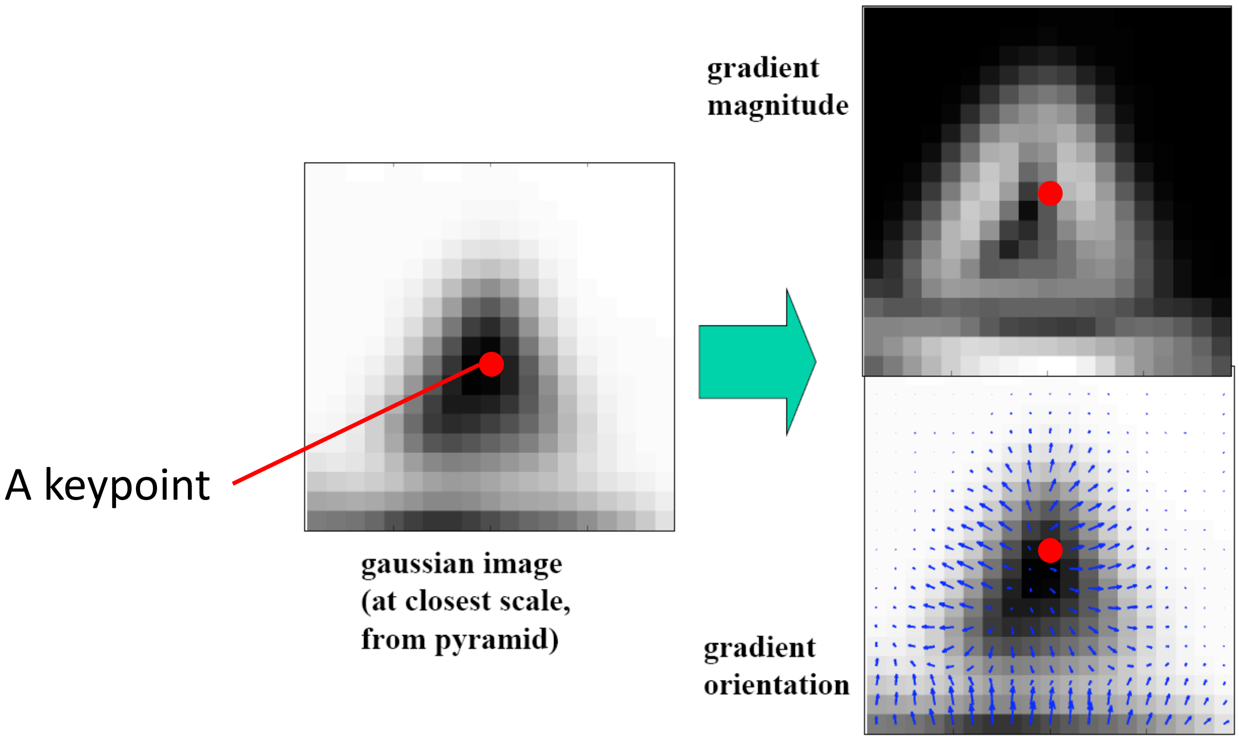


extrema locations extrema locations

(from 832 keypoints to 536 keypoints)

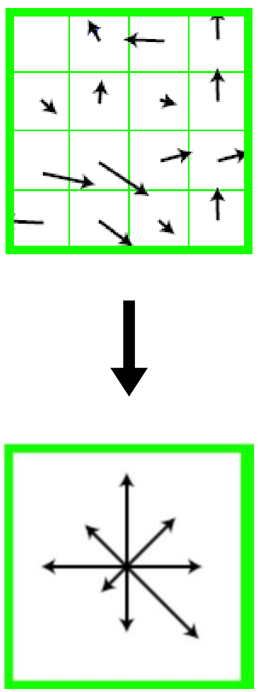
SIFT features

- SIFT (Scale Invariant Feature Transform)
  - 2) Keypoint description
    - Create a histogram of local gradient directions at selected scale
    - Orientation of a keypoint = the largest value of surrounding gradients

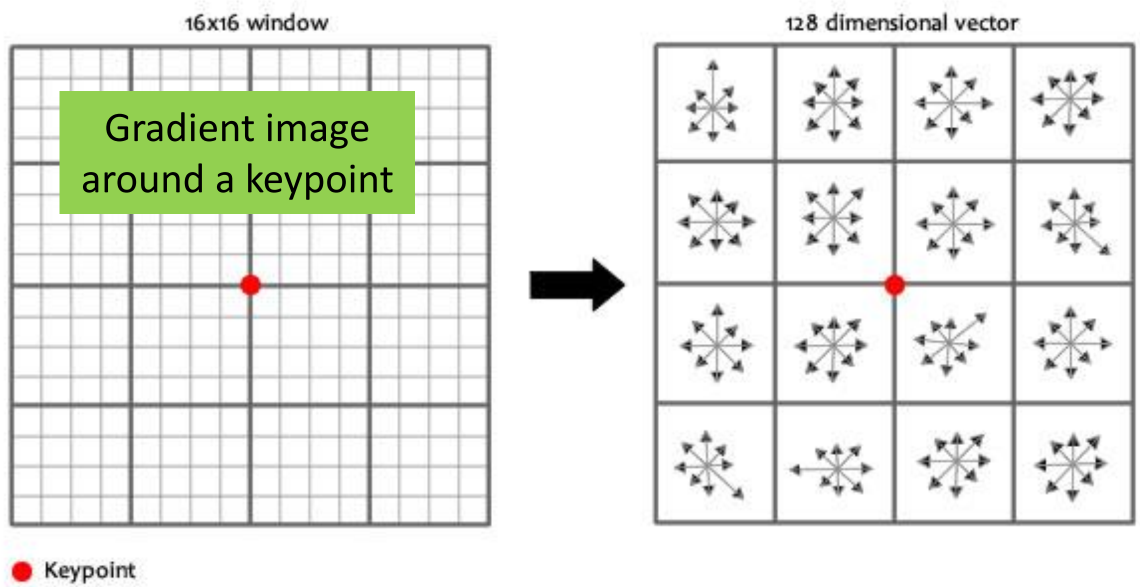


SIFT features

- SIFT (Scale Invariant Feature Transform)
  - 2) Keypoint description
    - Compute a descriptor for the local image region about each keypoint
    - Keypoint (x, y, scale, orientation) = [feature vector]



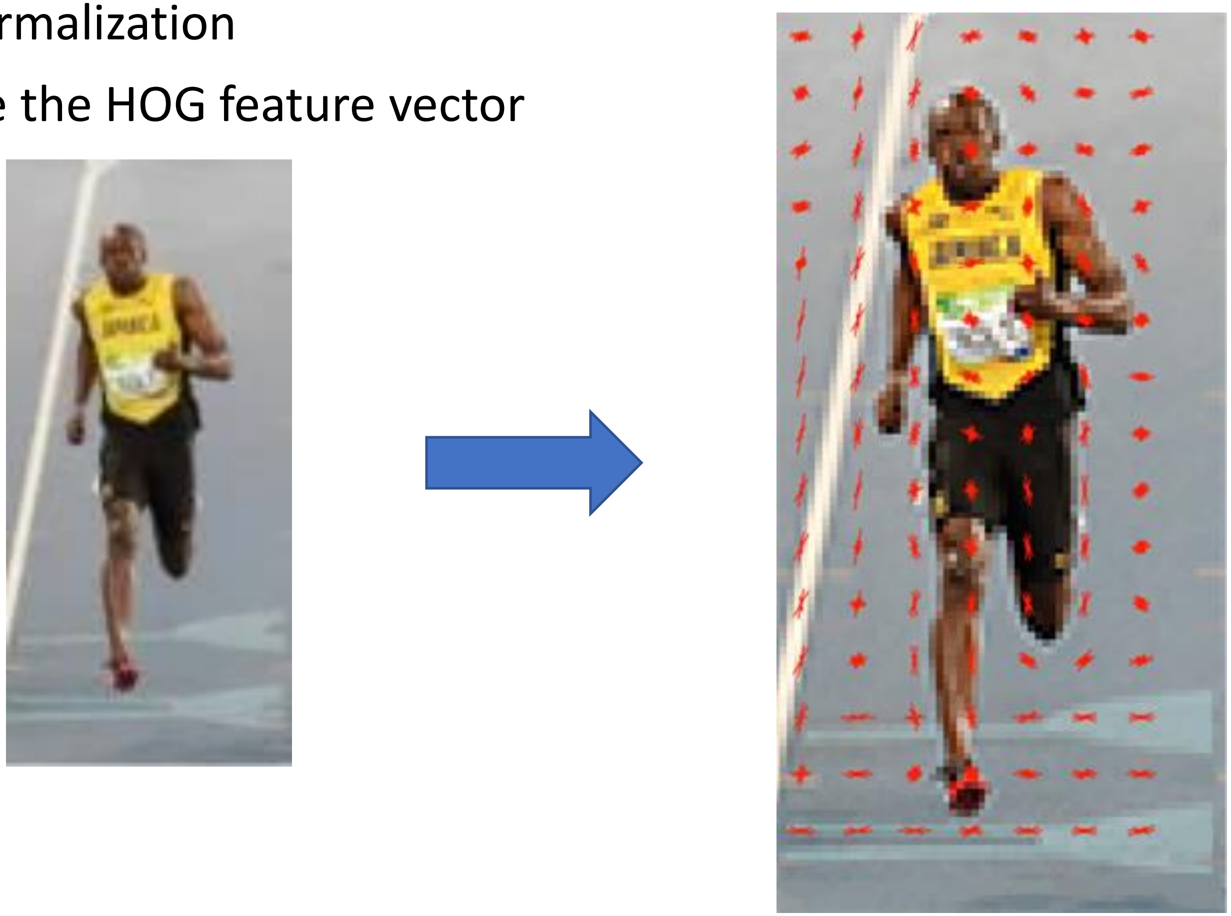
8 bin orientation



4x4 arrays * 8 bin histogram
= 128 features for one keypoint

HOG features

- HOG (Histogram of Oriented Gradient)
  - 1) Gradient computation
  - 2) Calculate Histogram of Gradients in 8×8 cells
  - 3) Block normalization
  - 4) Calculate the HOG feature vector



HOG features

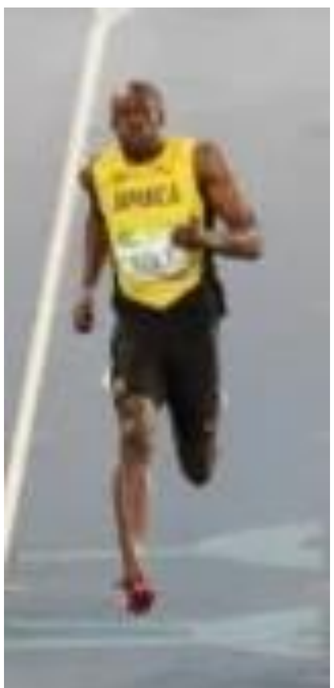
- HOG (Histogram of Oriented Gradient)
  - 1) Gradient computation

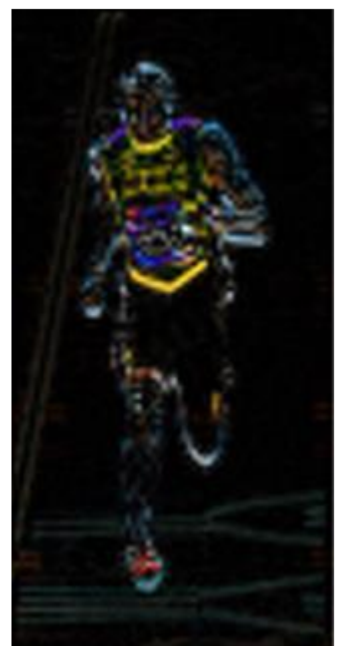
X-gradient (g_x)

-1	0	1

Y-gradient (g_y)

-1
0
1





Gradient direction

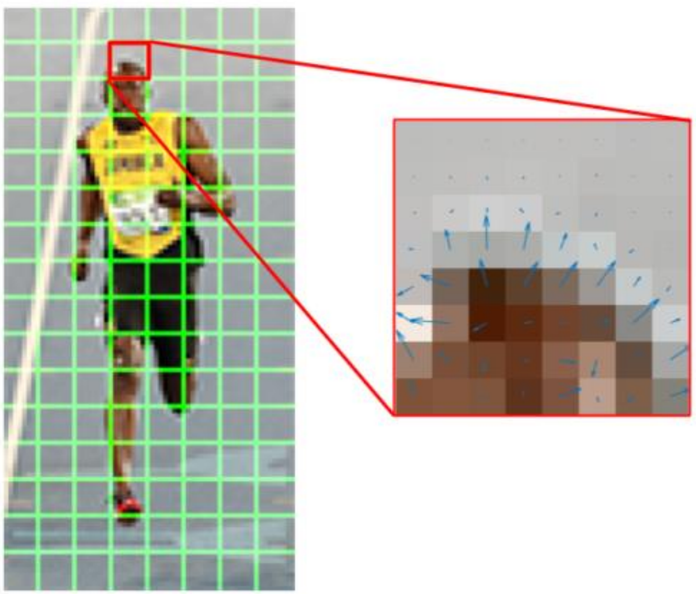
$$g = \arctan \frac{g_y}{g_x}$$

Gradient magnitude

$$g = \sqrt{g_x^2 + g_y^2}$$

HOG features

- HOG (Histogram of Oriented Gradient)
  - 2) Calculate Histogram of Gradients in 8×8 cells



80	36	5	10	0	64	90	73
37	9	9	179	78	27	169	166
87	136	173	39	102	163	152	176
76	13	1	168	159	22	125	143
120	70	14	150	145	144	145	143
58	86	119	98	100	101	133	113
30	65	157	75	78	165	145	124
11	170	91	4	110	17	133	110

Gradient Direction

2	3	4	4	3	4	2	2
5	11	17	13	7	9	3	4
11	21	23	27	22	17	4	6
23	99	165	135	85	32	26	2
91	155	133	136	144	152	57	28
98	196	76	38	26	60	170	51
165	60	60	27	77	85	43	136
71	13	34	23	108	27	48	110

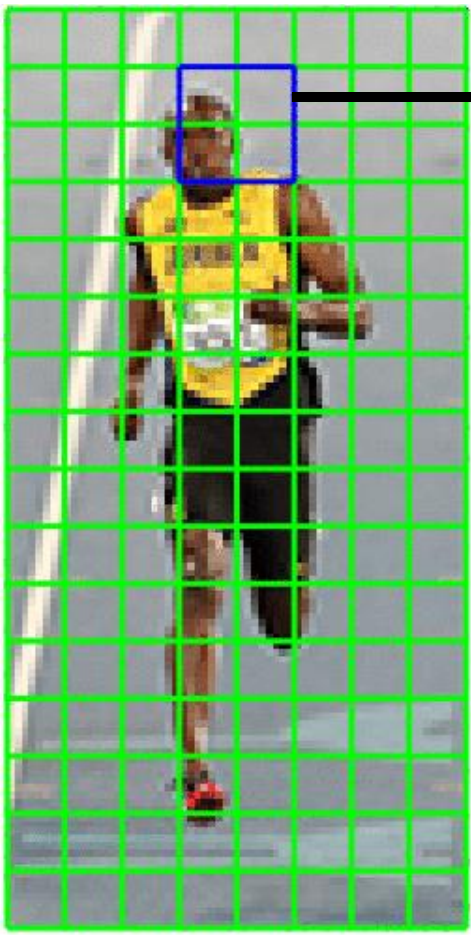
Gradient Magnitude

0	20	40	60	80	100	120	140	160
2	2			2				

Histogram of Gradients

HOG features

- HOG (Histogram of Oriented Gradient)
  - 3) Block normalization (contrast normalization)



Cell1	v00	v01	v02	...	v09
Cell2	v10	v11	v12	...	v19
Cell3	v20	v21	v22	...	v29
Cell4	v30	v31	v32	...	v39

Histogram of gradients

→ Concatenate 4 * 9 and normalize by $\frac{1}{\sum v_{ij}}$

→ 1 block vector = 4 cells * 9 normalized elements

HOG features

- HOG (Histogram of Oriented Gradient)
  - 4) Calculate the HOG feature vector

Total 105 blocks

➞ 105 blocks * 4 cells * 9 normalized elements

➞ 3780-sized feature vector

HOG features

- HOG (Histogram of Oriented Gradient)
  - 4) Calculate the HOG feature vector

← Visualized by averaging the 4 normalized cells