SUPPLEMENTARY

An Introduction to Edge Detection

Outline

- Spatial mask
- Introduction
- Edge detection
  - The Laplacian operator
  - Sobel operators
- Pre-Processing
  - Histogram equalization
  - Noise reduction
- Project assignment

SPATIAL MASK

An Introduction to Edge Detection

Spatial mask

- Gray Scale Image

y = \ x =	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72
41	210	209	204	202	197	247	143	71	64	80	84	54	54	57	58
42	206	196	203	197	195	210	207	56	63	58	53	53	61	62	51
43	201	207	192	201	198	213	156	69	65	57	55	52	53	60	50
44	216	206	211	193	202	207	208	57	69	60	55	77	49	62	61
45	221	206	211	194	196	197	220	56	63	60	55	46	97	58	106
46	209	214	224	199	194	193	204	173	64	60	59	51	62	56	48
47	204	212	213	208	191	190	191	214	60	62	66	76	51	49	55
48	214	215	215	207	208	180	172	188	69	72	55	49	56	52	56
49	209	205	214	205	204	196	187	196	86	62	66	87	57	60	48
[illegible]	[illegible]	[illegible]	[illegible]	203	202	186	174	185	149	71	63	55	55	45	56
[illegible]	[illegible]	[illegible]	[illegible]	199	217	194	183	177	209	90	62	64	52	93	52
[illegible]	[illegible]	[illegible]	[illegible]	209	197	194	183	187	187	239	58	68	61	51	56
[illegible]	[illegible]	[illegible]	[illegible]	209	195	203	188	185	183	221	75	61	58	60	60
[illegible]	[illegible]	[illegible]	[illegible]	236	188	197	183	190	183	196	122	63	58	64	66
[illegible]	[illegible]	[illegible]	[illegible]	203	199	197	196	181	173	186	105	62	57	64	63

Spatial mask

- Spatial domain process will be denoted by
  - $g(x, y) = T[f(x, y)]$
  - where f(x,y) is the input image,
    g(x,y) is the processed image
    T is an operator on f, defined over some neighborhood of (x,y)

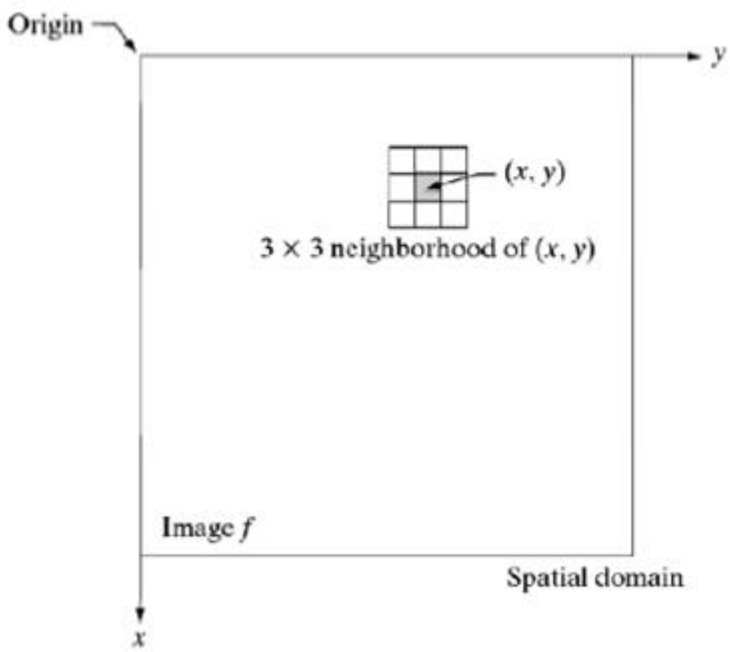


FIGURE 3.1 A 3 × 3 neighborhood about a point (x, y) in an image in the spatial domain. The neighborhood is moved from pixel to pixel in the image to generate an output image.

Spatial mask

- Convolution

$$f(t)\otimes g(t) = \int_{-\infty}^{\infty} f(\tau)h(t-\tau)d\tau$$

- Spatial Mask

$$f(x,y)\otimes g(x,y) = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} f(m,n)h(x-m,y-n)$$

for x=0,1,2,…,M-1, y=0,1,2,…,N-1

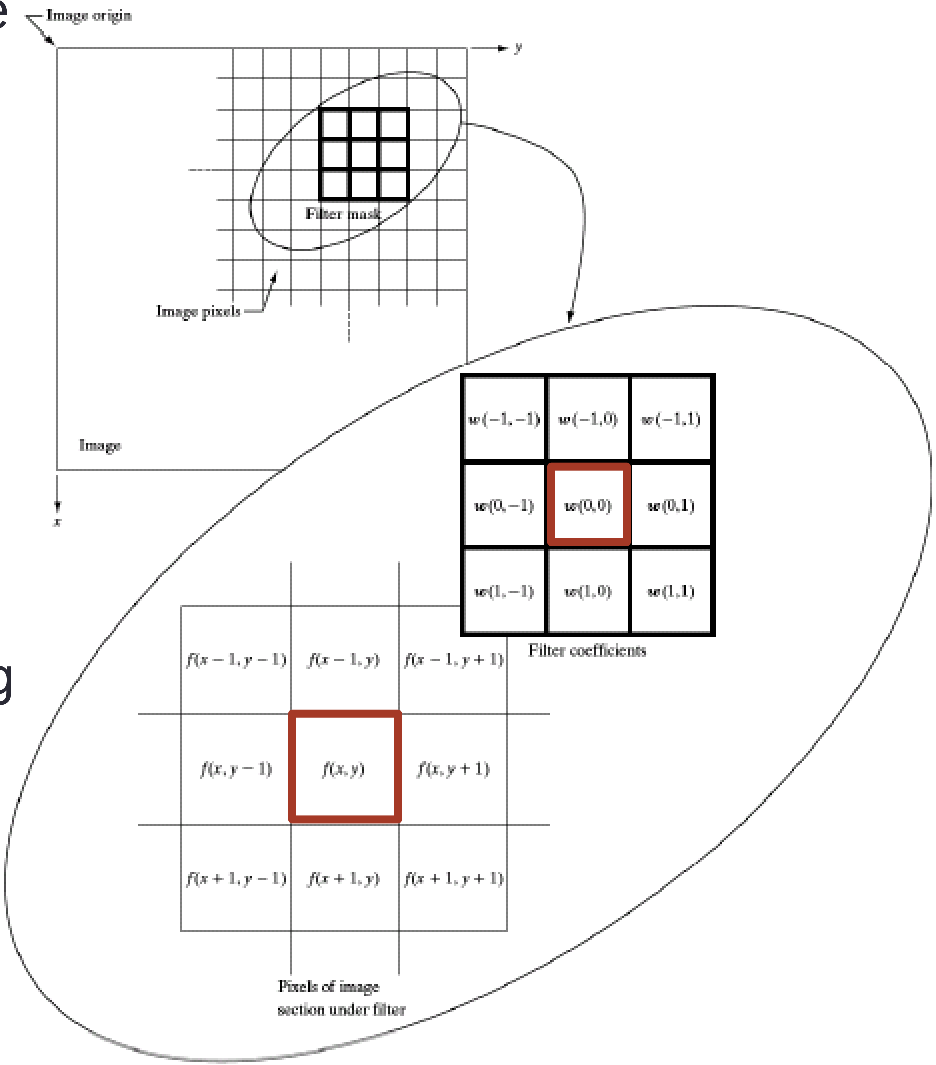
Spatial mask

- Linear filtering of an image f of size M*N with a filter mask of size m*n is given by the expression:

$$g(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x+s, y+t)$$

where $a = \lfloor m/2 \rfloor, b = \lfloor n/2 \rfloor$

- It is often referred to as “convolving a mask with an image”



EDGE DETECTION

An Introduction to Edge Detection

Edge detection

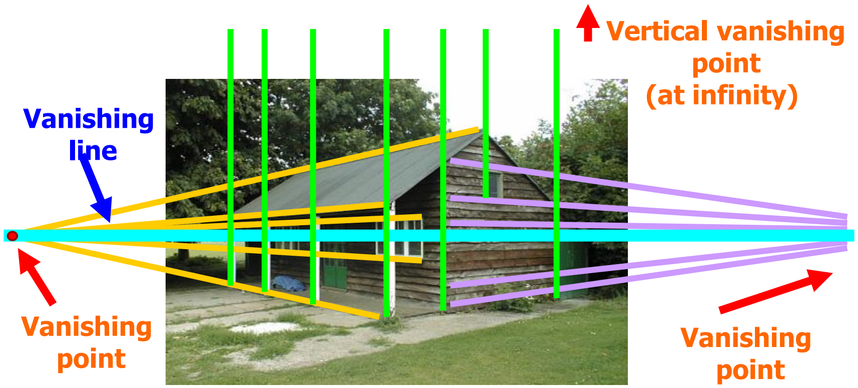
- **Goal:** Identify sudden changes (discontinuities) in an image
  - Intuitively, most semantic and shape information from the image can be encoded in the edges
  - More compact than pixels
- **Ideal:** artist's line drawing (but artist is also using object-level knowledge)

Source: D. Lowe

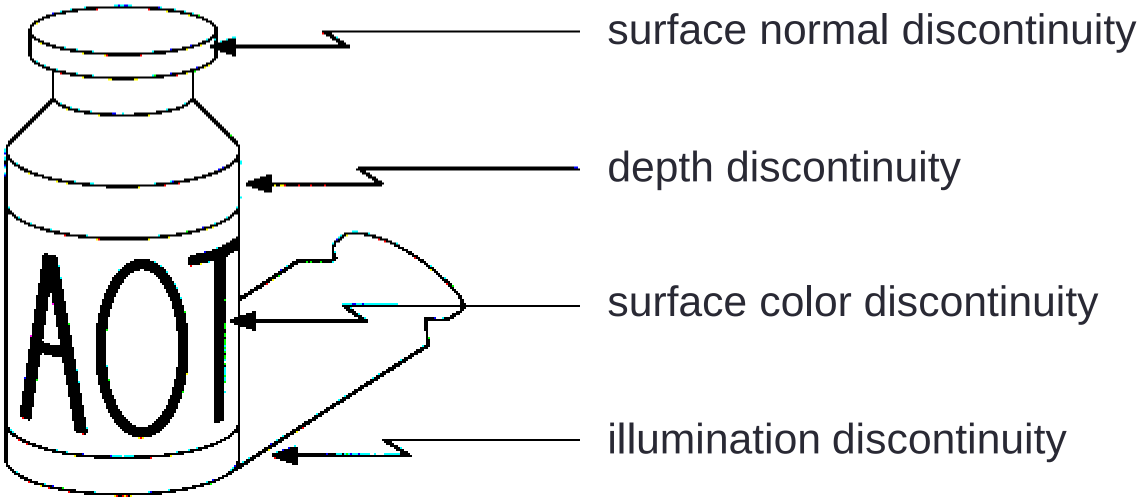
Why do we care about edges?

- Extract information, recognize objects

- Recover geometry and viewpoint



Origin of Edges



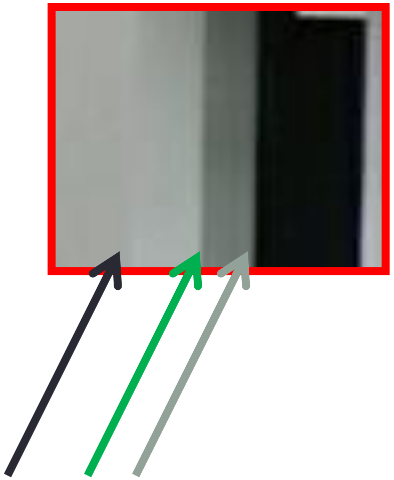
- Edges are caused by a variety of factors

Source: Steve Seitz

Closeup of edges

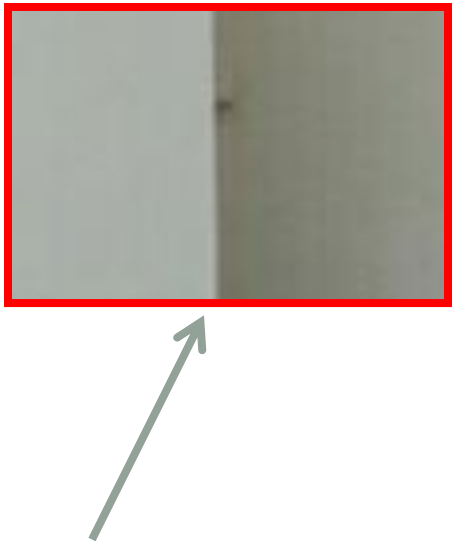
Source: D. Hoiem

Closeup of edges



Source: D. Hoiem

Closeup of edges



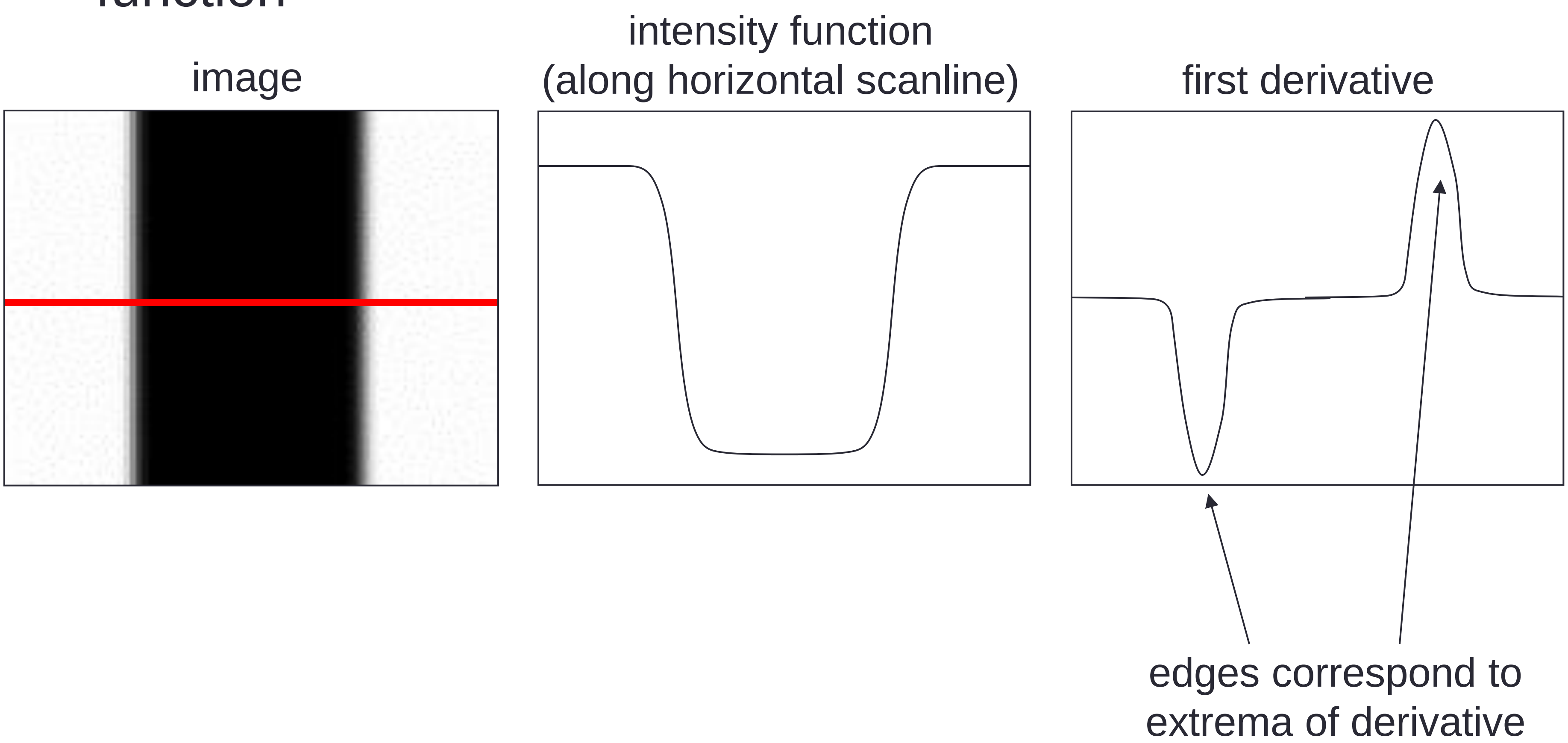
Source: D. Hoiem

Closeup of edges

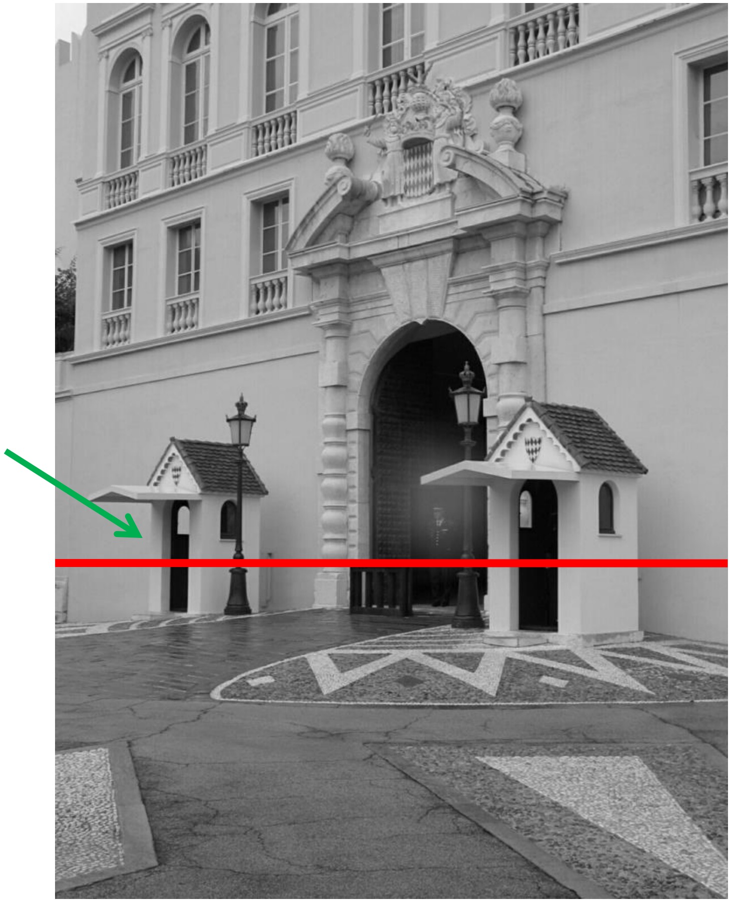
Source: D. Hoiem

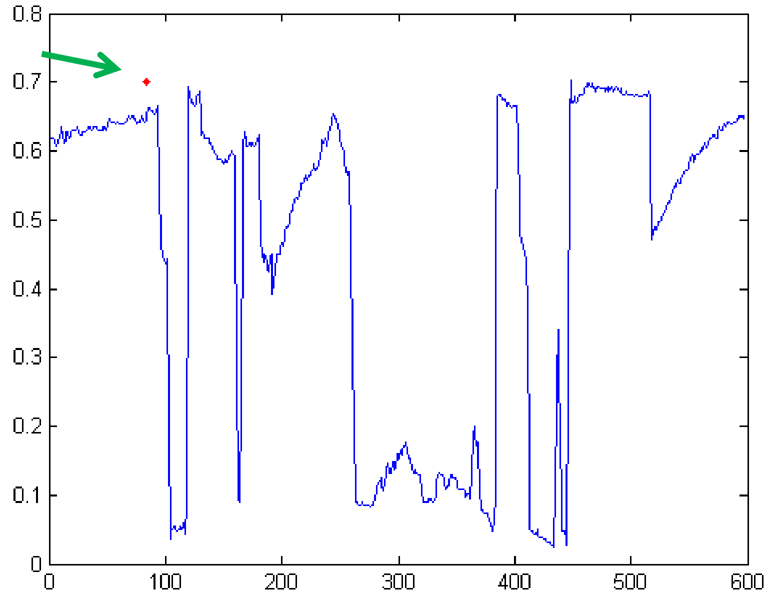
Characterizing edges

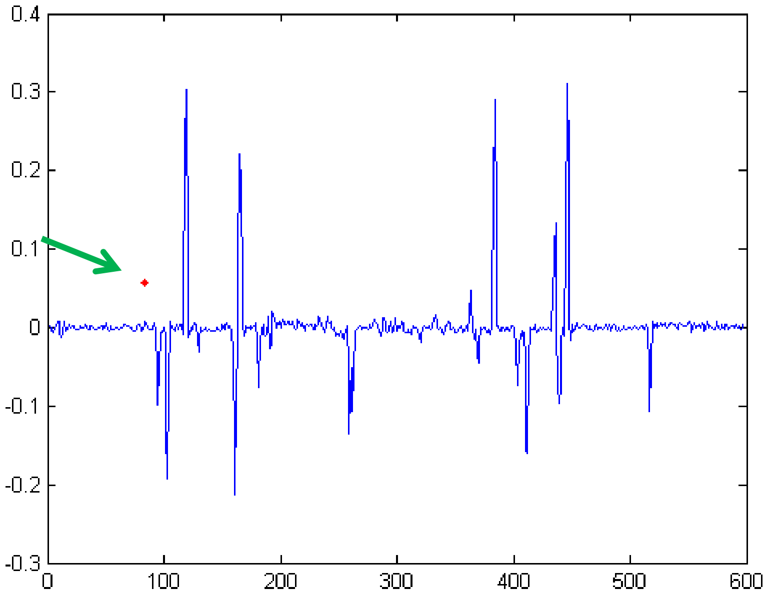
- An edge is a place of rapid change in the image intensity function



Intensity profile

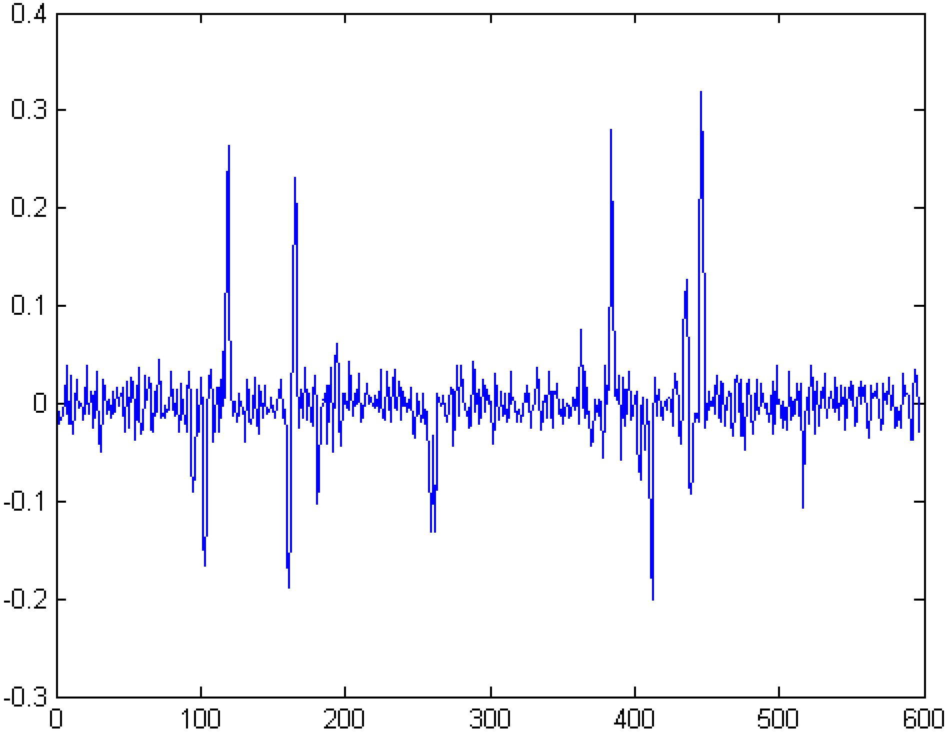






Source: D. Hoiem

With a little Gaussian noise

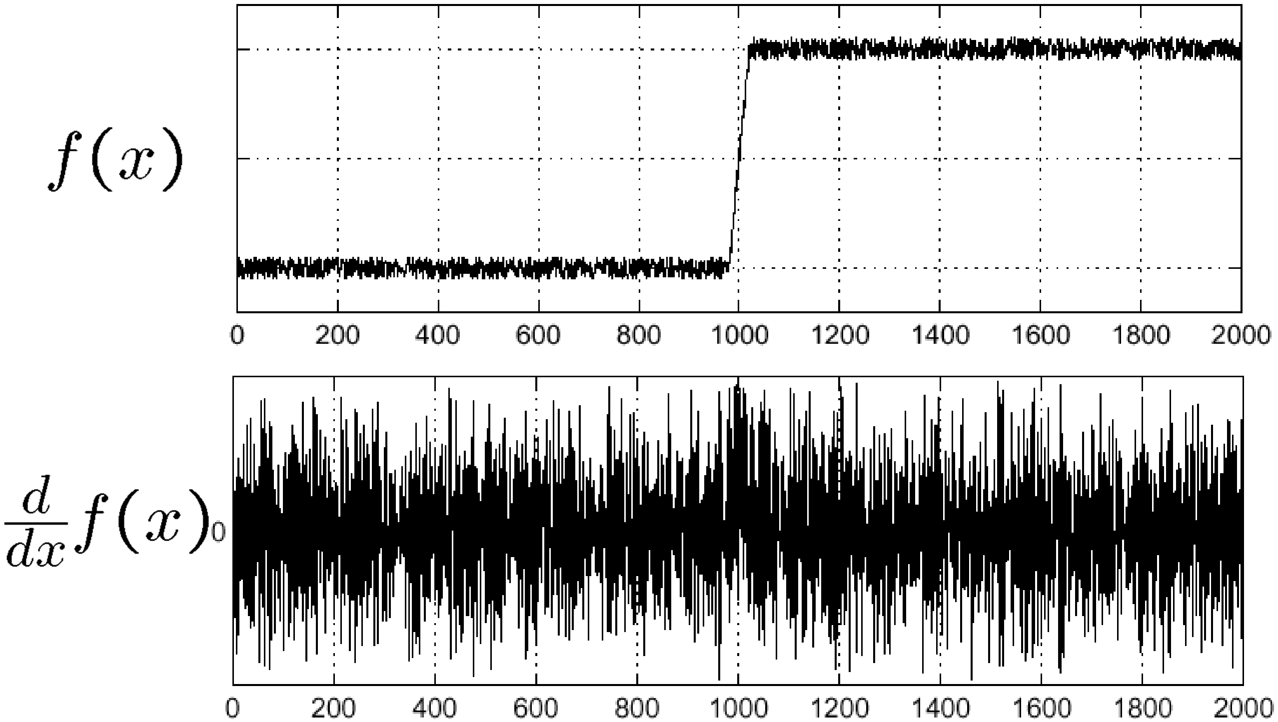


Gradient

Source: D. Hoiem

Effects of noise

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



Where is the edge?

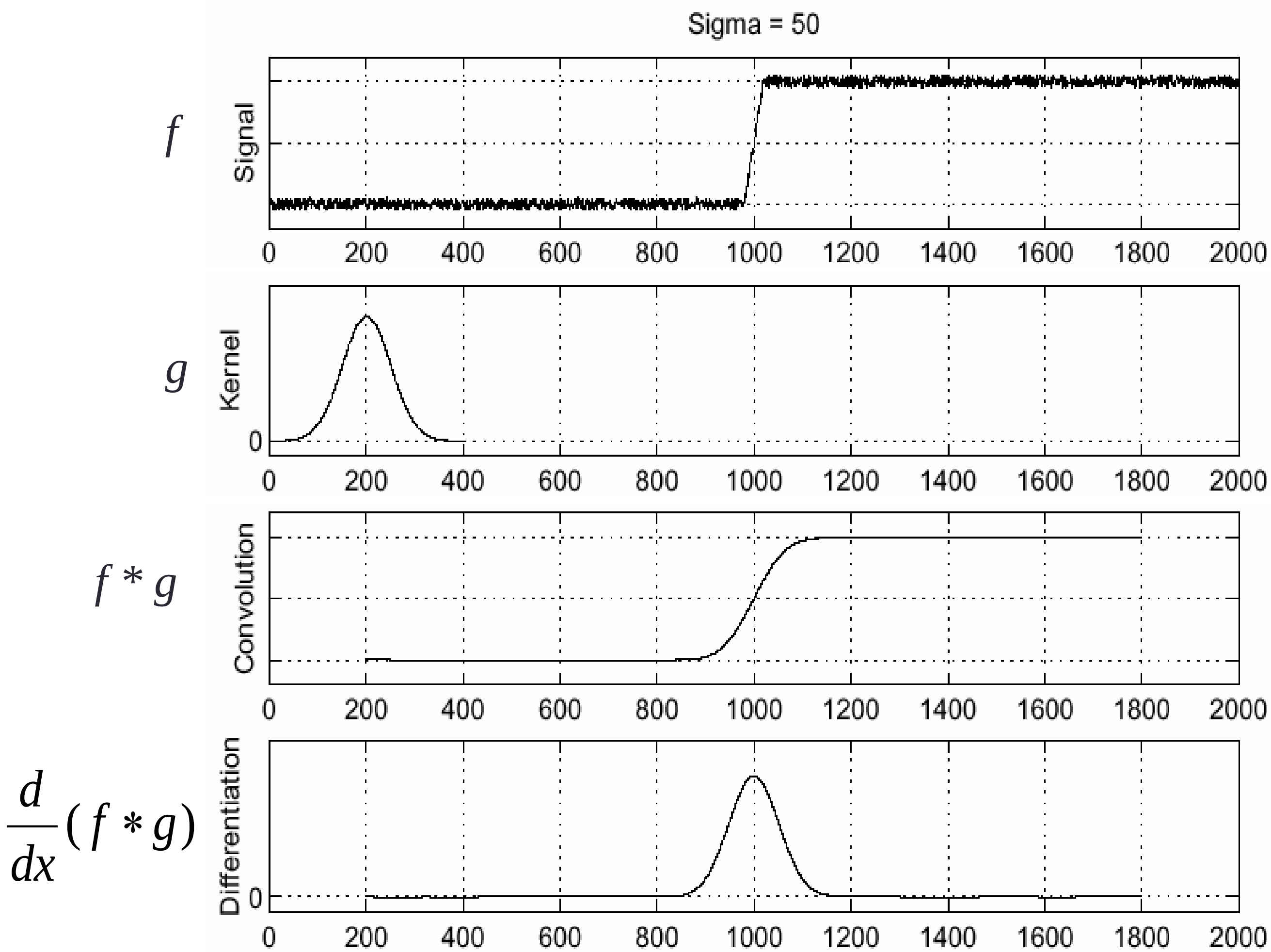
Source: S. Seitz

Effects of noise

- Difference filters respond strongly to noise
  - Image noise results in pixels that look very different from their neighbors
  - Generally, the larger the noise the stronger the response
- What can we do about it?

Source: D. Forsyth

Solution: smooth first



- To find edges, look for peaks in $\frac{d}{dx}(f * g)$

Source: S. Seitz

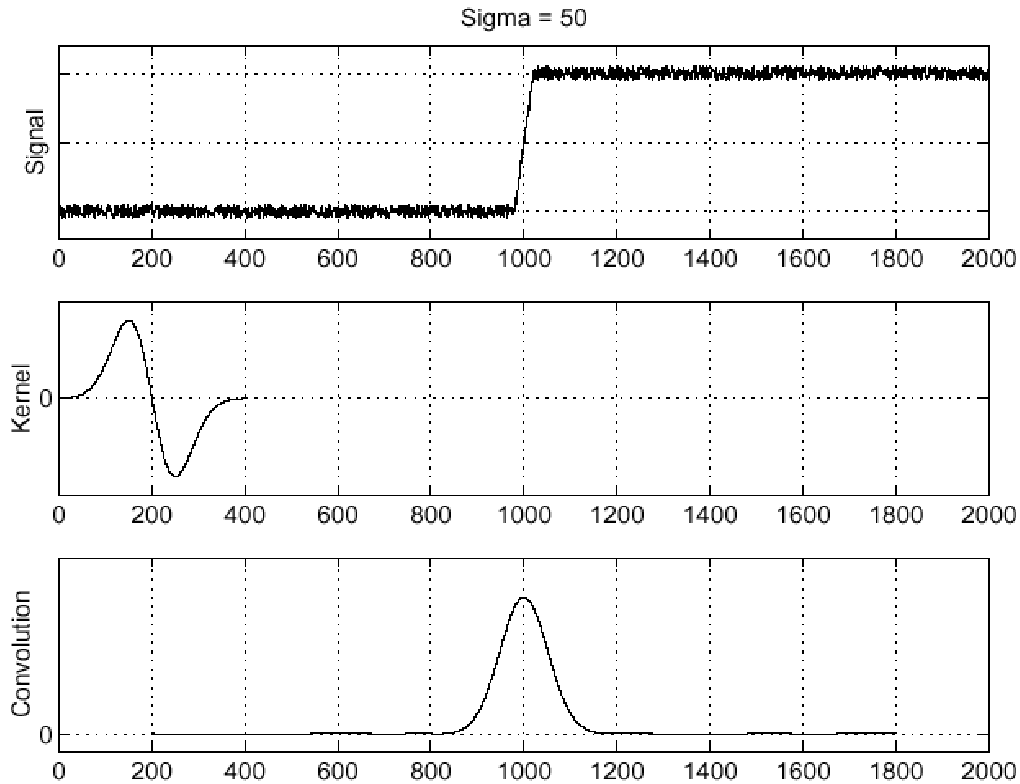
Derivative theorem of convolution

- Differentiation is convolution, and convolution is associative: $$\frac{d}{dx}(f * g) = f * \frac{d}{dx} g$$
- This saves us one operation:

f

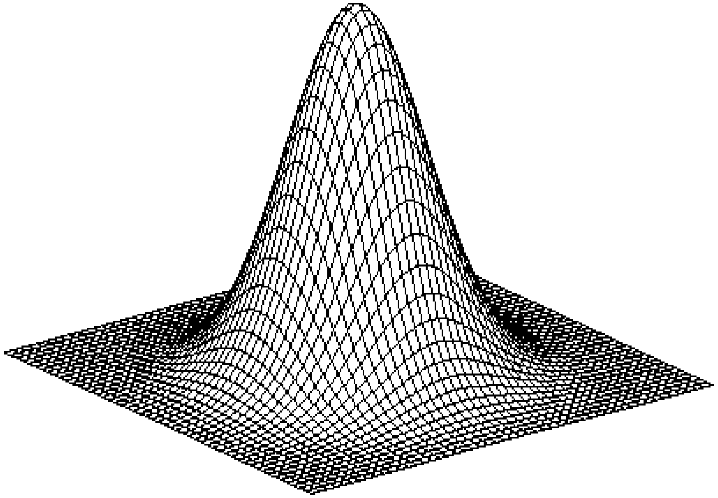
$\frac{d}{dx} g$

$f * \frac{d}{dx} g$

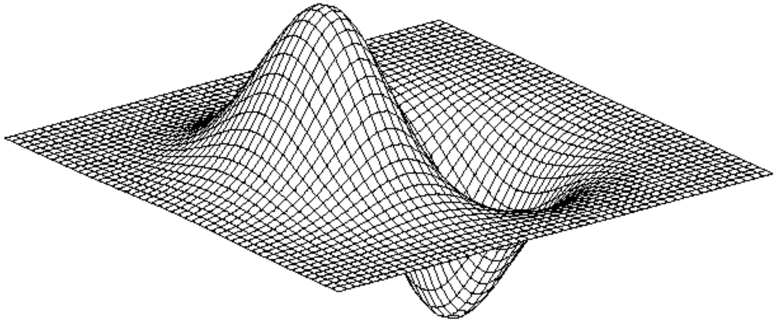


Source: S. Seitz

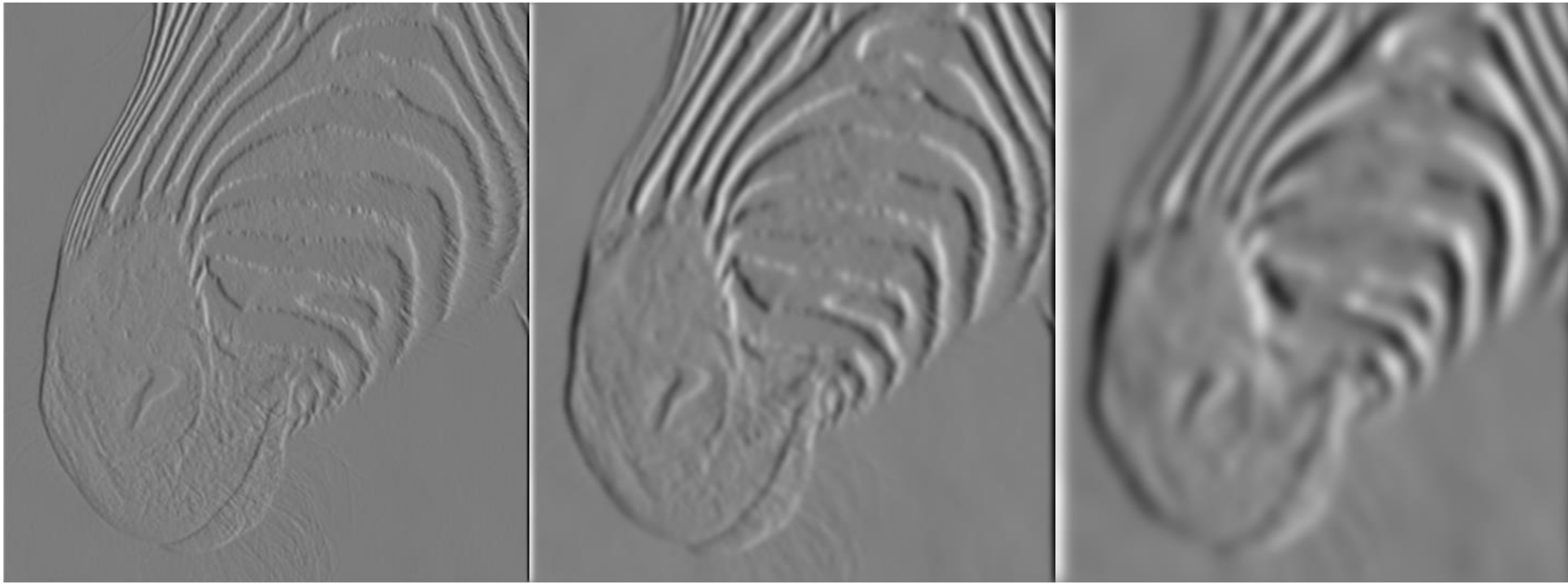
Derivative of Gaussian filter



* [1 -1] =



Tradeoff between smoothing and localization



1 pixel 3 pixels 7 pixels

- Smoothed derivative removes noise, but blurs edge. Also finds edges at different "scales".

Source: D. Forsyth

Designing an edge detector

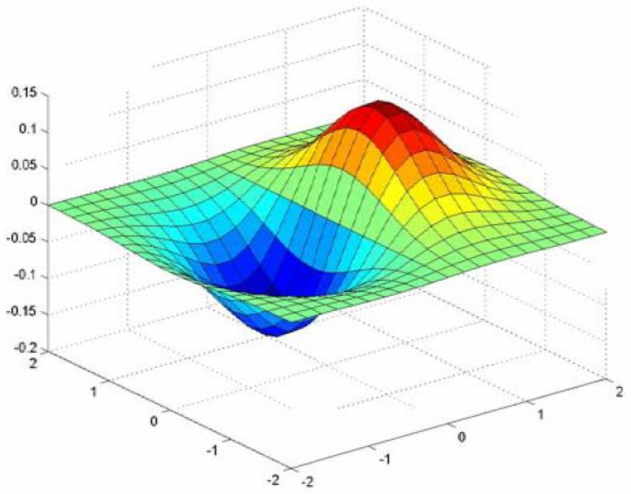
- Criteria for a good edge detector:
  - **Good detection:** the optimal detector should find all real edges, ignoring noise or other artifacts
  - **Good localization**
    - the edges detected must be as close as possible to the true edges
    - the detector must return one point only for each true edge point
- Cues of edge detection
  - Differences in color, intensity, or texture across the boundary
  - Continuity and closure
  - High-level knowledge

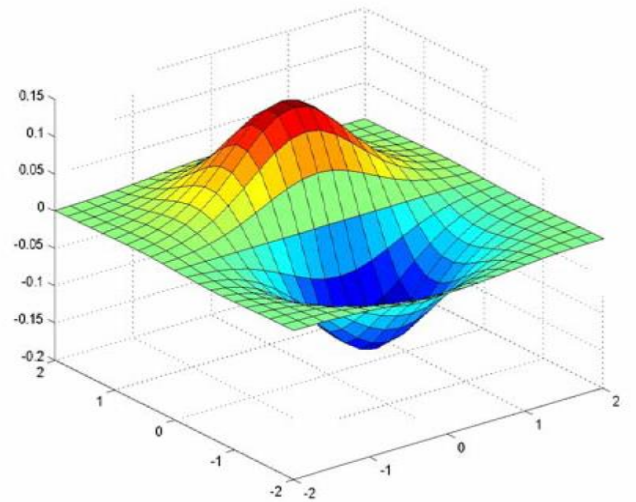
Source: L. Fei-Fei

Example

original image (Lena)

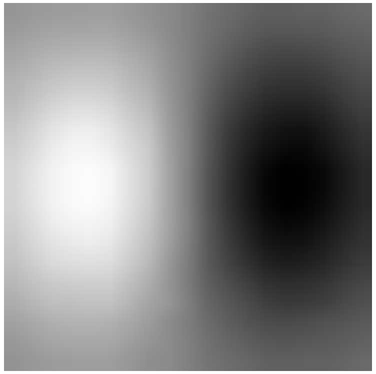
Derivative of Gaussian filter

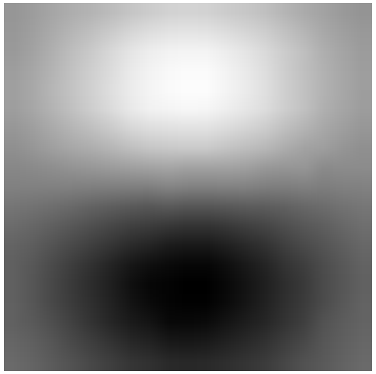




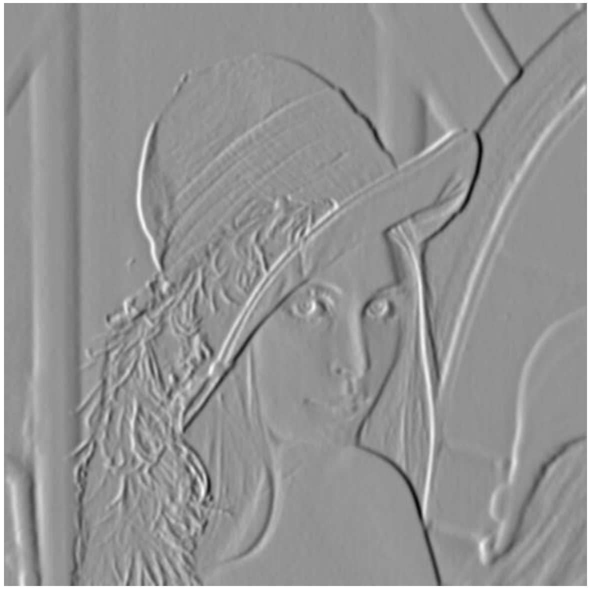
x-direction

y-direction





Compute Gradients (DoG)


X-Derivative of Gaussian


Y-Derivative of Gaussian


Gradient Magnitude

Learning to detect boundaries

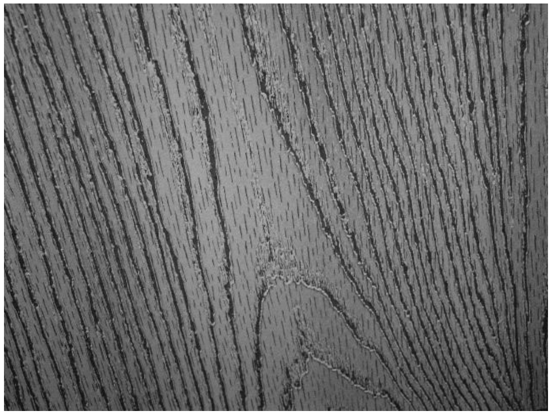
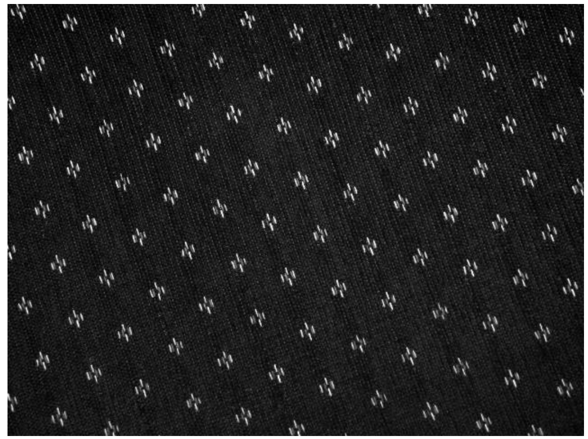
image | human segmentation | gradient magnitude

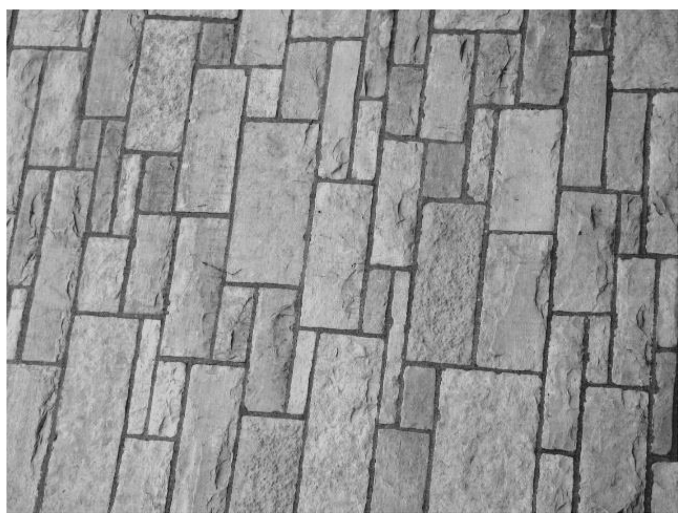
- Berkeley segmentation database: http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/

Representing Texture

Source: Forsyth

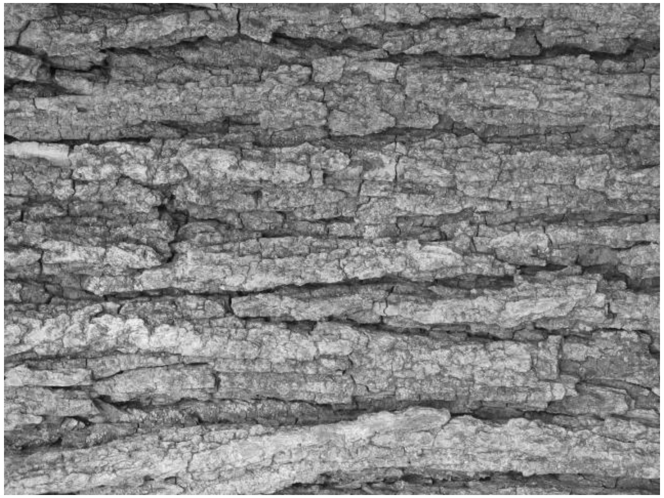
Texture and Material

http://www-cvr.ai.uiuc.edu/ponce_grp/data/texture_database/samples/

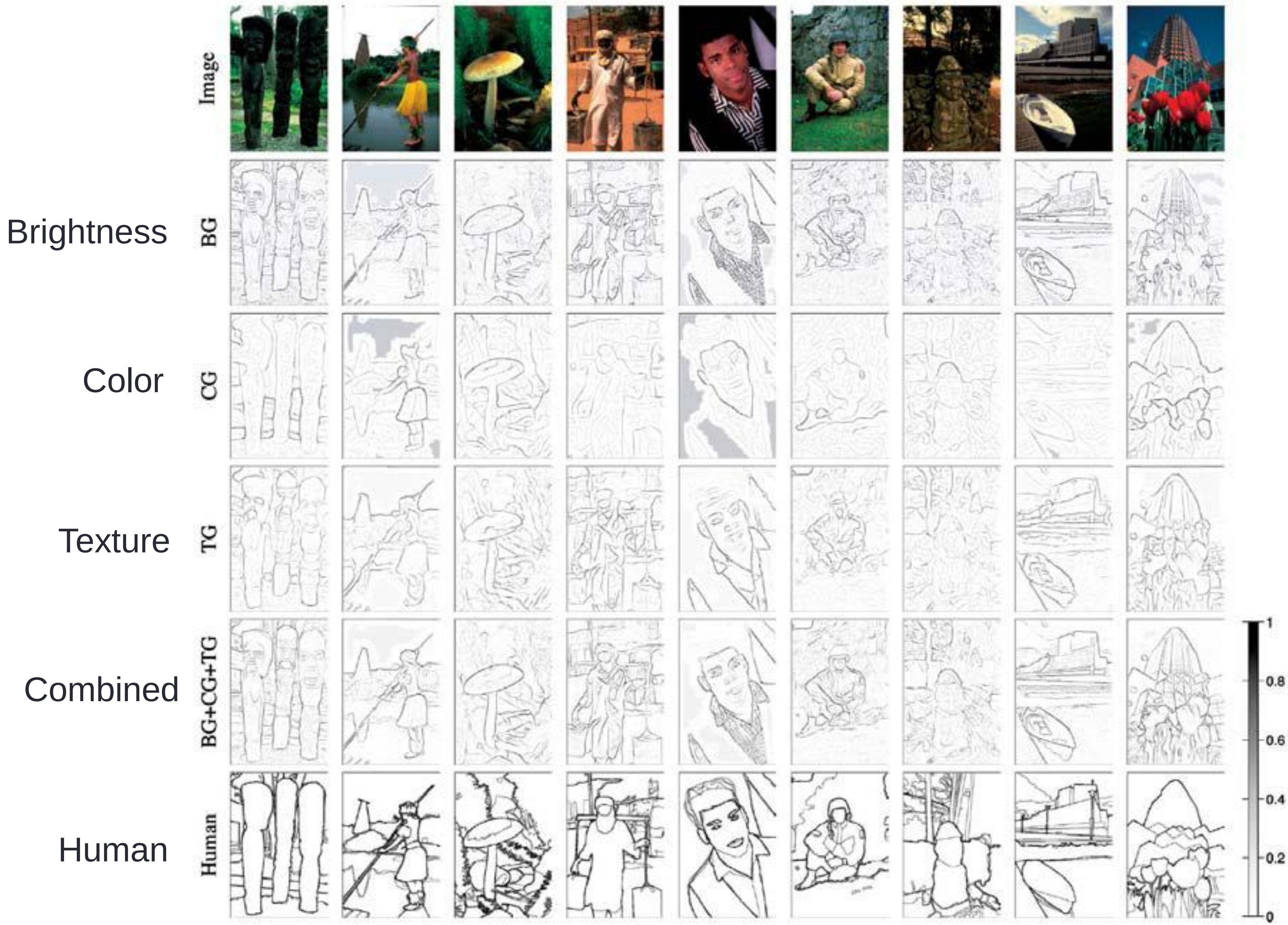
Texture and Orientation



http://www-cvr.ai.uiuc.edu/ponce_grp/data/texture_database/samples/

Texture and Scale

http://www-cvr.ai.uiuc.edu/ponce_grp/data/texture_database/samples/



Brightness
Color
Texture
Combined
Human
Image
BG
CG
TG
BG+CG+TG
Human
1
0.8
0.6
0.4
0.2
0

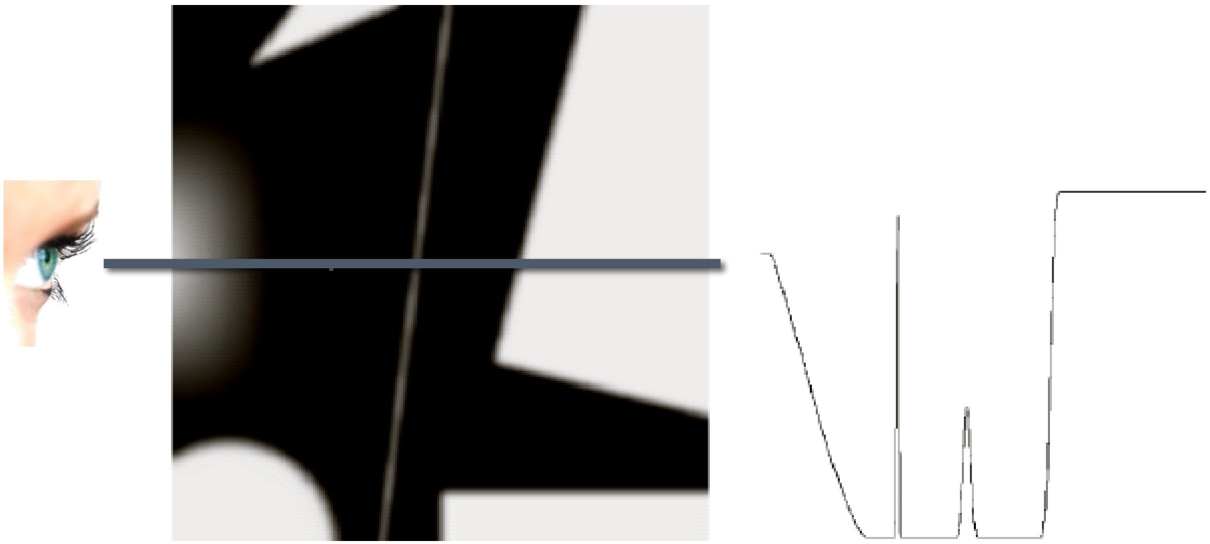
Edge detection
- The Laplacian operator

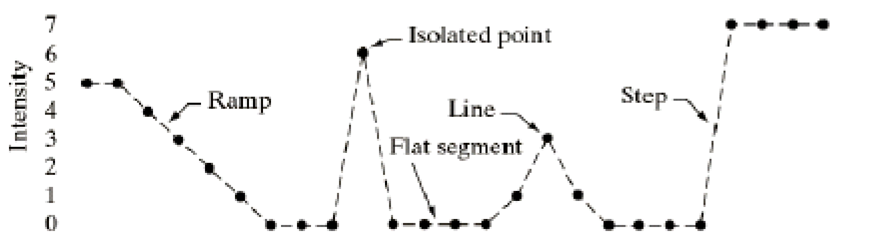
- The Laplacian of an image highlights regions of rapid intensity change

- The Laplacian operator searches for zero crossings in the second derivative of the image to find edges
  - When the first derivative is at an extreme, the second derivative is zero

Edge detection
- The Laplacian operator

- Horizontal intensity profile through the center of the image





$f(x)$ Image strip	5	5	4	3	2	1	0	0	0	6	0	0	0	0	1	3	1	0	0	0	0	7	7	7	7	•	•
$\frac{\partial f(x)}{\partial x}$ First derivative		−1	−1	−1	−1	−1	0	0	6	−6	0	0	0	1	2	−2	−1	0	0	0	7	0	0	0			
$\frac{\partial^2 f(x)}{\partial x^2}$ Second derivative		−1	0	0	0	0	1	0	6	−12	6	0	0	1	1	−4	1	1	0	0	7	−7	0	0			

Edge detection
- The Laplacian operator

- For one-dimensional function f(x) (an image)

$$\frac{\partial f(x)}{\partial x} = f'(x) \cong f(x+1) - f(x)$$

$$\frac{\partial^2 f(x)}{\partial x^2} = \frac{f'(x)}{\partial x} \cong f'(x+1) - f'(x)$$

$$= \big(f(x+2) - f(x+1)\big) - \big(f(x+1) - f(x)\big)$$

$$= f(x+2) + f(x) - 2f(x+1)$$

Edge detection
- The Laplacian operator

- For two-dimensional function f(x) (an image)

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

$$\frac{\partial^2 f(x,y)}{\partial x^2} = f(x+1,y) + f(x-1,y) - 2f(x,y)$$

$$\frac{\partial^2 f(x,y)}{\partial y^2} = f(x,y+1) + f(x,y-1) - 2f(x,y)$$

$$\nabla^2 f(x,y) = f(x+1,y) + f(x-1,y) \\ + f(x,y+1) + f(x,y-1) - 4f(x,y)$$

Edge detection
- The Laplacian operator

- The Laplacian operator

0	1	0
1	−4	1
0	1	0

1	1	1
1	−8	1
1	1	1

0	−1	0
−1	4	−1
0	−1	0

−1	−1	−1
−1	8	−1
−1	−1	−1

a b
c d

FIGURE 3.37 (a) Filter mask used to implement Eq. (3.6-6). (b) Mask used to implement an extension of this equation that includes the diagonal terms. (c) and (d) Two other implementations of the Laplacian found frequently in practice.

Edge detection
- The Laplacian operator

- Advantage
  - Detection of edges and their orientations
  - Having fixed characteristics in al directions

- Drawback
  - Responding to some of the existing edges
  - Very sensitivity to noise

Edge detection
- Sobel operator

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

$$\frac{\partial f(x,y)}{\partial x} = \sum_{s=-1}^{1}\sum_{t=-1}^{1} g_x(s,t) f(x+s, y+t)$$

$$\frac{\partial f(x,y)}{\partial y} = \sum_{s=-1}^{1}\sum_{t=-1}^{1} g_y(s,t) f(x+s, y+t)$$

g_x: *horizontal edge*

−1	−2	−1
0	0	0
1	2	1

g_y: *vertical edge*

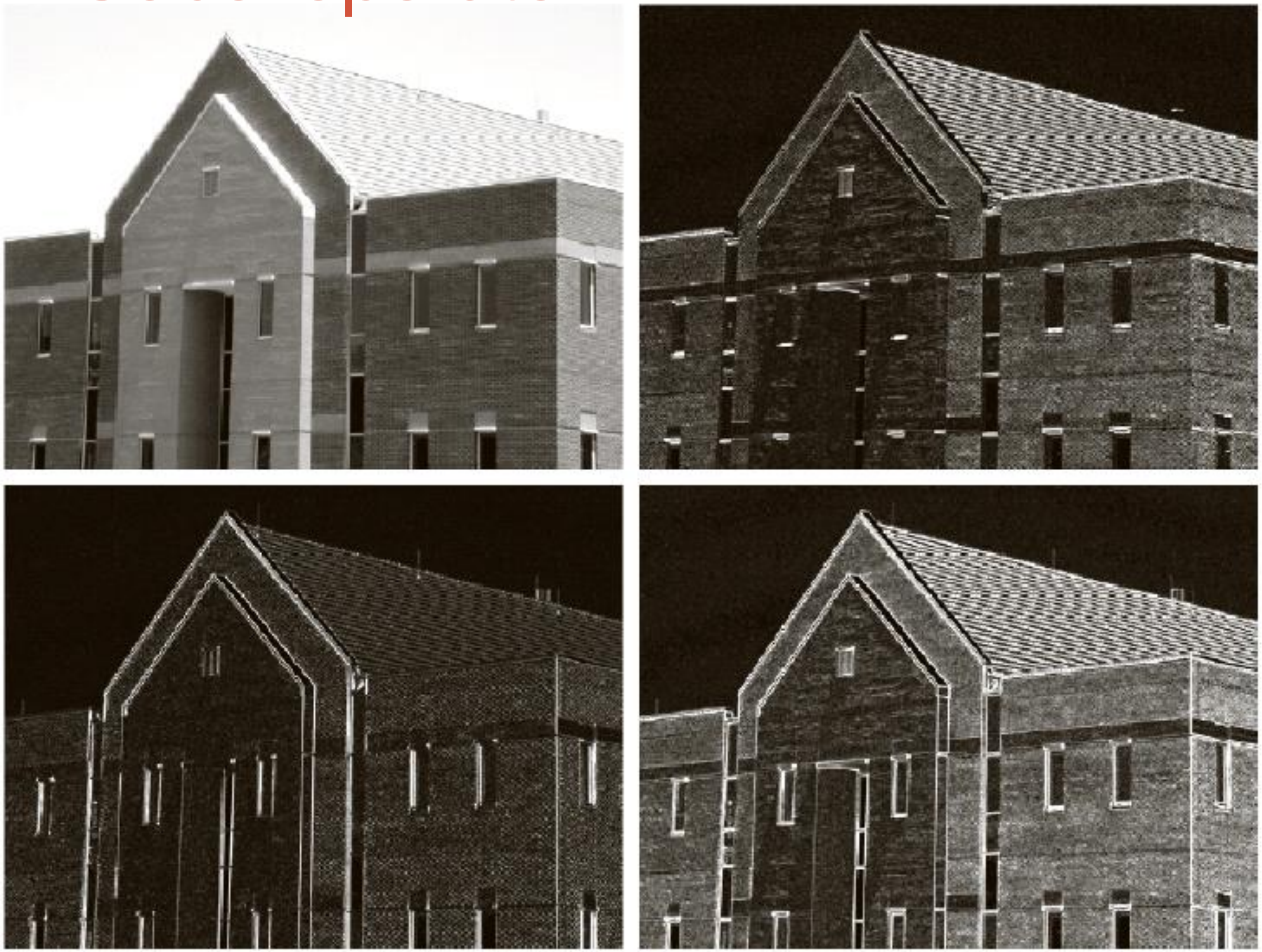
−1	0	1
−2	0	2
−1	0	1

Optional: *diagonal edge*

0	1	2
−1	0	1
−2	−1	0

−2	−1	0
−1	0	1
0	1	2

Edge detection - Sobel operator



a b
c d

FIGURE 10.16 (a) Original image of size 834 × 1114 pixels, with intensity values scaled to the range [0, 1]. (b) $|g_x|$, the component of the gradient in the x-direction, obtained using the Sobel mask in Fig. 10.14(f) to filter the image. (c) $|g_y|$, obtained using the mask in Fig. 10.14(g). (d) The gradient image, $|g_x| + |g_y|$.

Edge detection
- Sobel operator

- Advantage
  - Easy to be implemented in hardware and software
    - Only eight image points around a point are needed to compute the corresponding result
    - Only integer arithmetic is needed to compute the opposite of the gradient vector approximation

- Drawback
  - Sensitivity to noise
  - Inaccurate since the range is limited in 3x3

PRE-PROCESSING

An Introduction to Edge Detection

Pre-Processing

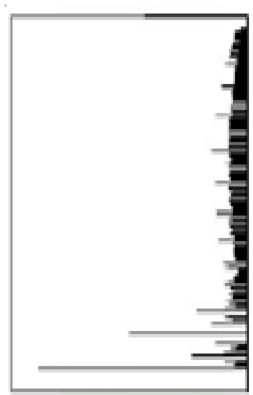
- Histogram equalization
  - The discontinuities between pixels may be continuous if the contrast of the image is low

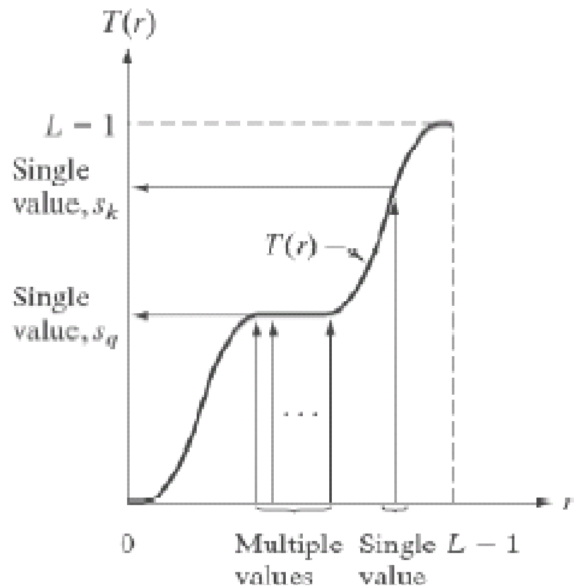
- Noise Reduction
  - The false edges are detected since the operators are sensitive to noise

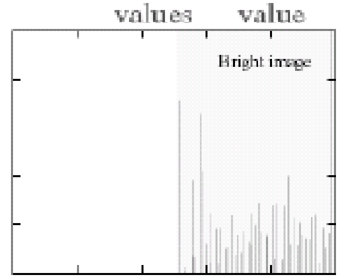
- Morphology
  - A technique for analysis and processing of geometrical structures

Pre-processing
- Histogram equalization

1.2







1.1

Figure 1. 1.1- original image, 1.2- Resulting image after histogram equalization

Pre-processing
- Histogram equalization

- Histogram equalization is a technique where the histogram of resultant image is as flat as possible

- For an image that its gray level is in [0,L], Histogram is defined as:
  - $h(r_k) = n_k$
  - where r_k is k-th gray level, n_k is the number of pixels that have r_k gray level.

- A normalized histogram is defined as:
  - $p(r_k) = n_k/MN$

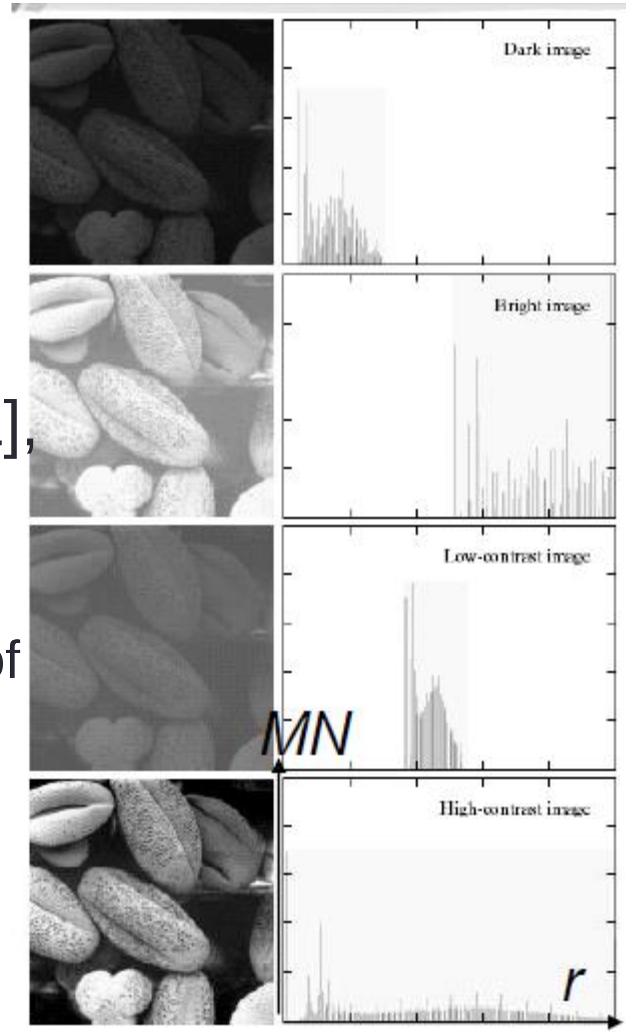
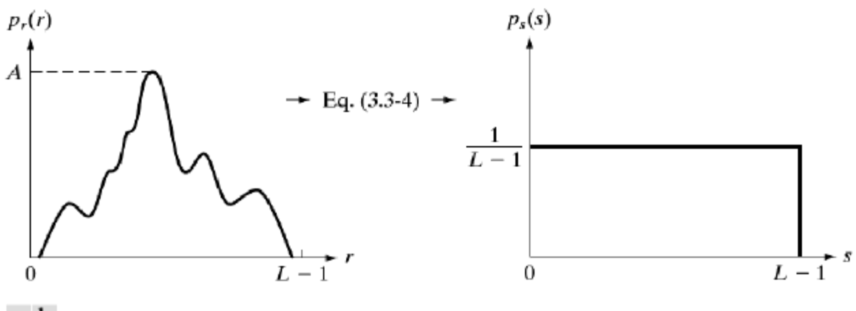


FIGURE 3.16 Four basic image types: dark, light, low contrast, high contrast, and their corresponding histograms.

Pre-processing
- Histogram equalization

- For any *r* satisfying the aforementioned conditions, we focus attention on transformations of the form
- $$s = T(r) \quad 0 \leq r \leq L-1$$

- The transformation function $T(r)$ satisfies the following conditions:
  - (a) $T(r)$ is single-valued and monotonically increasing in the interval $0 \leq r \leq L-1$
  - (b) $0 \leq T(r) \leq L-1$ for $0 \leq r \leq L-1$



Pre-processing
- Histogram equalization

- $s = T(r) = (L-1)\int_0^r p_r(w)dw$

- $\frac{ds}{dr} = \frac{dT(r)}{dr} = (L-1)\frac{d}{dr}\{\int_0^r p_r(w)dw\} = (L-1)p_r(r)$

- $p_s(s) = p_r(r)\left|\frac{dr}{ds}\right| = p_r(r)\left|\frac{1}{(L-1)p_r(r)}\right| = \frac{1}{L-1}$
- The form of ps(s) is a uniform probability density function

- Since $p(r_k) = n_k/MN$,
- $s = T(r_k) = (L-1)\sum_{j=0}^{k} p_r(r_j) = \frac{L-1}{MN}\sum_{j=0}^{k} n_j \qquad , k = 0,1,\ldots,L-1$

Pre-processing
- Histogram equalization

- Suppose that a 3-bit image (L=8) of size 64x64 pixels (MN=4096) has the intensity distribution shown in Table 3.1

TABLE 3.1 Intensity distribution and histogram values for a 3-bit, 64 × 64 digital image.

r_k	n_k	$p_r(r_k) = n_k/MN$
$r_0 = 0$	790	0.19
$r_1 = 1$	1023	0.25
$r_2 = 2$	850	0.21
$r_3 = 3$	656	0.16
$r_4 = 4$	329	0.08
$r_5 = 5$	245	0.06
$r_6 = 6$	122	0.03
$r_7 = 7$	81	0.02

- Histogram equalization
  - $S_0 = T(r_0) = 7\sum_{j=0}^{0} p_r(r_j) = 7p_r(r_0) = 1$
  - $S_1 = T(r_1) = 7\sum_{j=0}^{1} p_r(r_j) = 7p_r(r_1) = 3.08 \cong 3$
  - $S_2 \cong 5,\ S_3 \cong 6,\ S_4 \cong 6,\ S_5 \cong 7,\ S_6 \cong 7,\ S_7 \cong 7.$

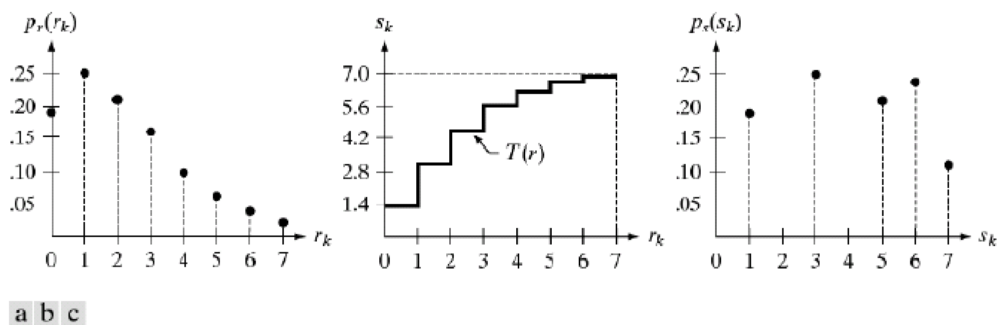


FIGURE 3.19 Illustration of histogram equalization of a 3-bit (8 intensity levels) image. (a) Original histogram. (b) Transformation function. (c) Equalized histogram.

PROJECT ASSIGNMENT

An Introduction to Edge Detection

Project – Edge detection

- Write a program using the language (Matlab, C++,…) you are familiar with to detect the edges from an image
- The image can be downloaded from the website

- Basic issue
  - Histogram equalization
  - Sobel operator
  - Add your comments in source code
    - The goal of the function, the main concept of the code section, the meaning of each variable, …
- Bonus
  - Other ideas for improving the detection result

- Submission due: Two weeks after the project is assigned

Project – Edge detection

Image_0314

Image_0323

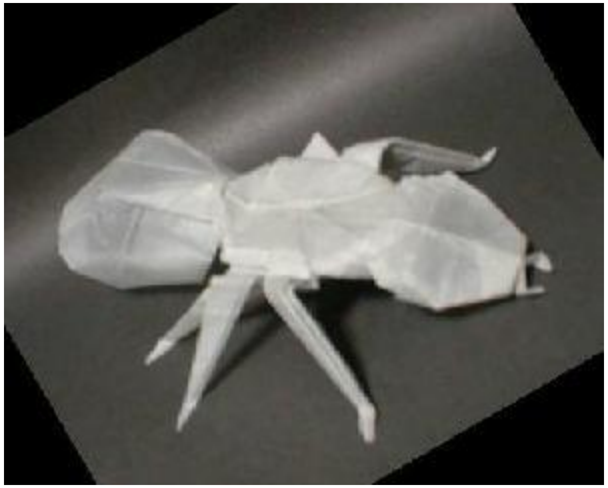


Image_0008

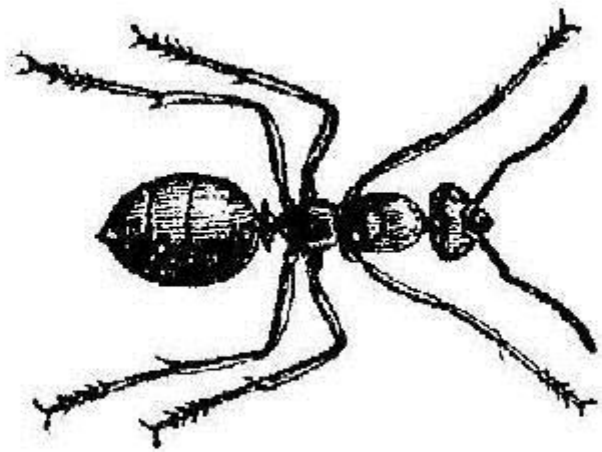


Image_0023

Project – Edge detection

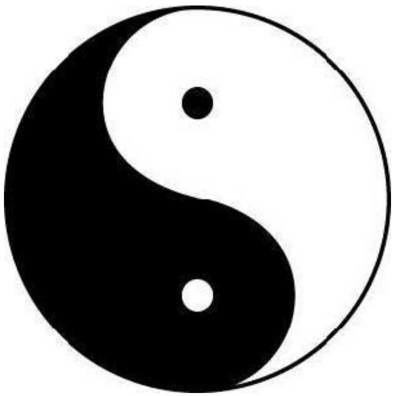


image_0001.jpg

image_0069.jpg

image_0013.jpg

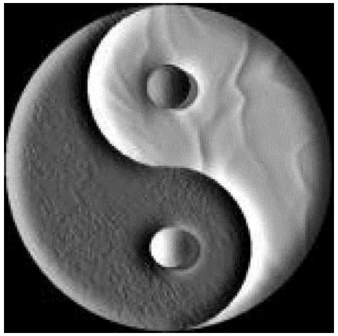


image_0021.jpg

image_0034.jpg

image_0002.jpg