ECE 172A: Introduction to Image Processing
Image Processing Tasks: Part I

Rahul Parhi
Assistant Professor, ECE, UCSD

Winter 2025

Outline

- Preprocessing
  - Histogram
  - Normalization
  - Combining Images
  - Spatial Averaging
- Matching and Detection
  - Correlation
  - Matched Filtering
- Feature Extraction
  - Contour/Edge Detection
- Segmentation
  - Variational Thresholding
  - Connected-Component Labeling

Preprocessing

- Histogram
- Normalization
- Combining Images
- Spatial Averaging
  - Linear Smoothing
  - Median Filtering

Graylevel Histogram

Input image: $r[\boldsymbol{k}]$ with $\boldsymbol{k} \in \Omega = \{0, \dots, K-1\} \times \{0, \dots, L-1\}$

Total number of pixels: $\#\Omega = KL$

- Graylevel distribution

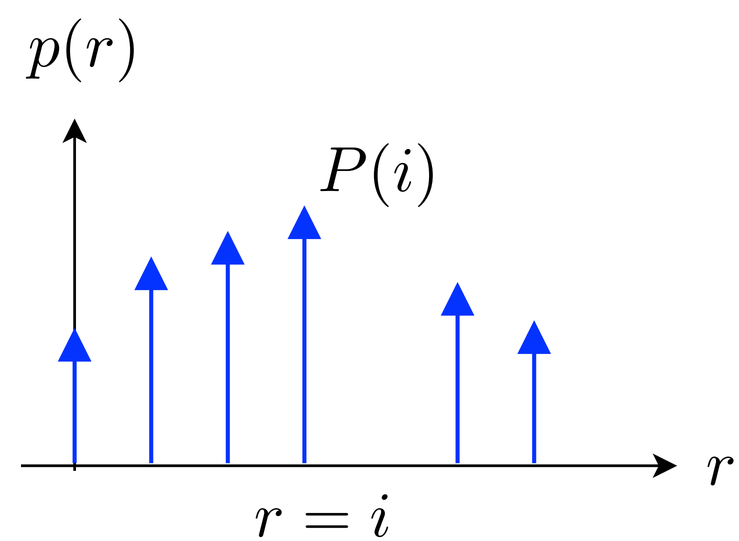
  p.d.f. $p(r)$ with $\int_{-\infty}^{\infty} p(r)\,\mathrm{d}r = 1$

- Histogram

  Quantized graylevels: $\{0, 1, 2, \dots, N-1\}$

  n_i: number of pixels with graylevel i

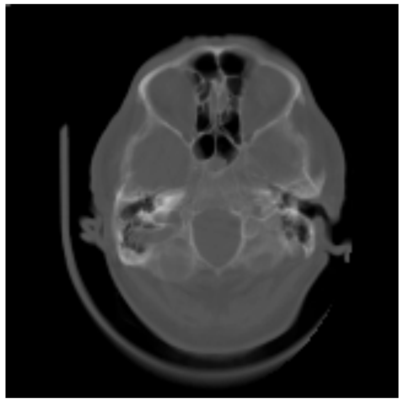
  $P(i) = \dfrac{n_i}{\#\Omega}$: probability of graylevel i

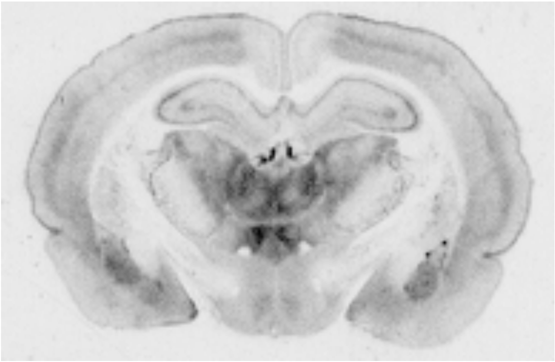


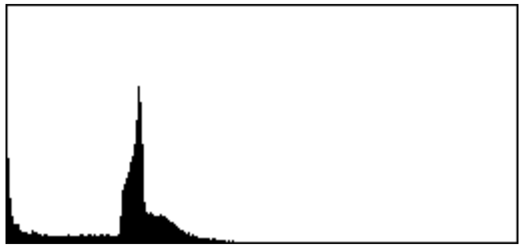
$$p(r) = \sum_{i=0}^{N-1} P(i)\delta(r-i)$$

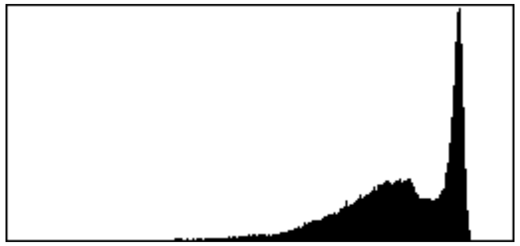
- Probability mass function (p.m.f.)

Examples of Histograms









What can we do with these histograms?

- Reading the histogram can tell us about:
  - Dynamic range
  - Potential saturation problems
  - Average intensities of background and objects

Normalization: Linear Contrast Adjustment

Linear transformation/system: $\mathrm{T}\{f\}[\boldsymbol{k}] = \alpha(f[\boldsymbol{k}]-\beta)$ with parameters $\alpha, \beta \in \mathbb{R}$

How to we implement full dynamic-range contrast stretching?

$$\beta = \min\{f[\boldsymbol{k}] \; : \; \boldsymbol{k} \in \Omega \subset \mathbb{Z}^2\} \qquad \alpha = \frac{255}{\max_{\boldsymbol{k}}\{f[\boldsymbol{k}]\} - \min_{\boldsymbol{k}}\{f[\boldsymbol{k}]\}}$$

- Image normalization

Average graylevel

$$\mu = \frac{1}{\#\Omega} \sum_{\boldsymbol{k}\in\Omega} f[\boldsymbol{k}]$$

Variance

$$\sigma^2 = \frac{1}{\#\Omega} \sum_{\boldsymbol{k}\in\Omega} (f[\boldsymbol{k}] - \mu)^2$$

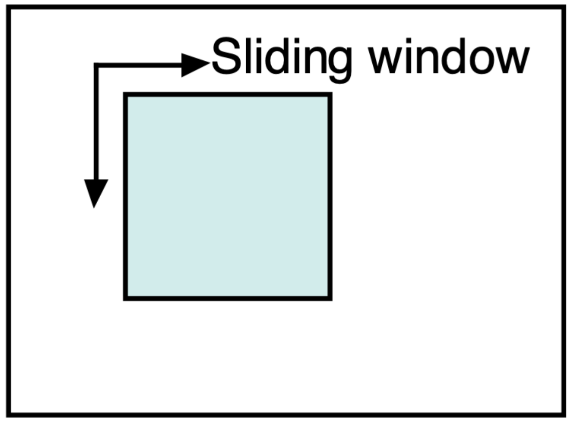
Normalized image statistics: $\mathrm{T}\{f\}[\boldsymbol{k}] = a\left(\frac{f[\boldsymbol{k}] - \mu}{\sigma}\right) + b$

"zero mean and unit variance"

Localized Normalization

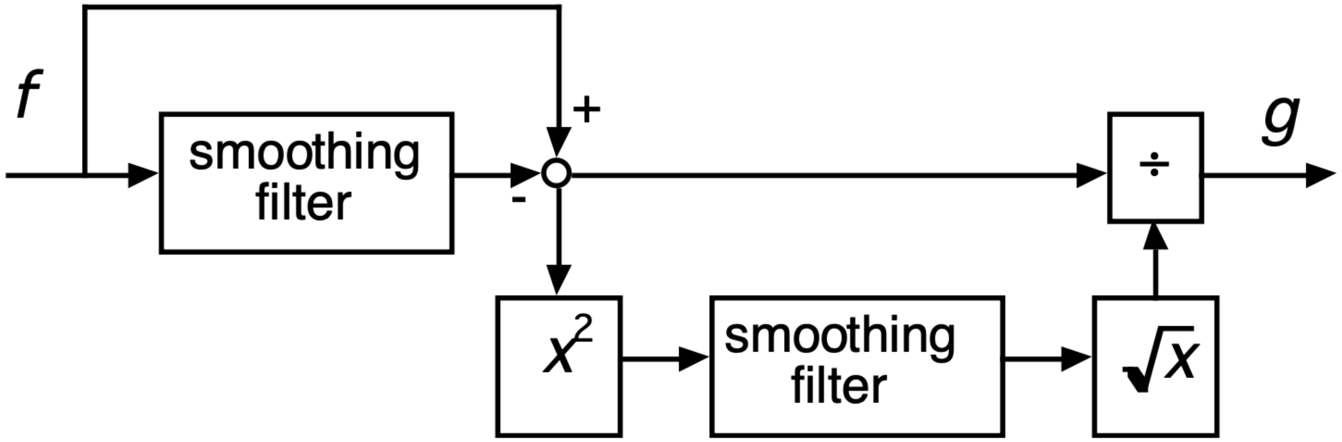
Compensation of non-uniformities across the image;
e.g., shading, nonuniform background, changes in illumination

- Normalization over a sliding window:



$$g[\boldsymbol{k}] = a\left(\frac{f[\boldsymbol{k}] - \tilde{\mu}[\boldsymbol{k}]}{\tilde{\sigma}[\boldsymbol{k}]}\right) + b$$

$$\tilde{\mu}[\boldsymbol{k}] = \sum_{\boldsymbol{n}} w[\boldsymbol{n}] f[\boldsymbol{n} - \boldsymbol{k}] \qquad \sum_{\boldsymbol{k}} w[\boldsymbol{k}] = 1$$



https://bigwww.epfl.ch/demo/ip/demos/local-normalization/

Smoothing filter implements local averaging ⇒ Estimation of local statistics

Combining Images

- Averaging for noise reduction:
  - Independent noisy observations: $f_i[\boldsymbol{k}] = s[\boldsymbol{k}] + n_i[\boldsymbol{k}], \quad i = 1, \ldots, N$
  - Hypotheses:

    (i) $\mathbf{E}[f_i[\boldsymbol{k}]] = s[\boldsymbol{k}] \quad \Rightarrow \quad \mathbf{E}[n_i[\boldsymbol{k}]] = 0$

    (ii) i.i.d. noise at each location $\boldsymbol{k}$ $\quad \Rightarrow \quad \mathrm{var}(f_i[\boldsymbol{k}]) = \mathrm{var}(n_i[\boldsymbol{k}]) = \sigma^2$

  - Noise reduction procedure: $\bar{f}[\boldsymbol{k}] = \dfrac{1}{N} \displaystyle\sum_{i=1}^{N} f_i[\boldsymbol{k}]$

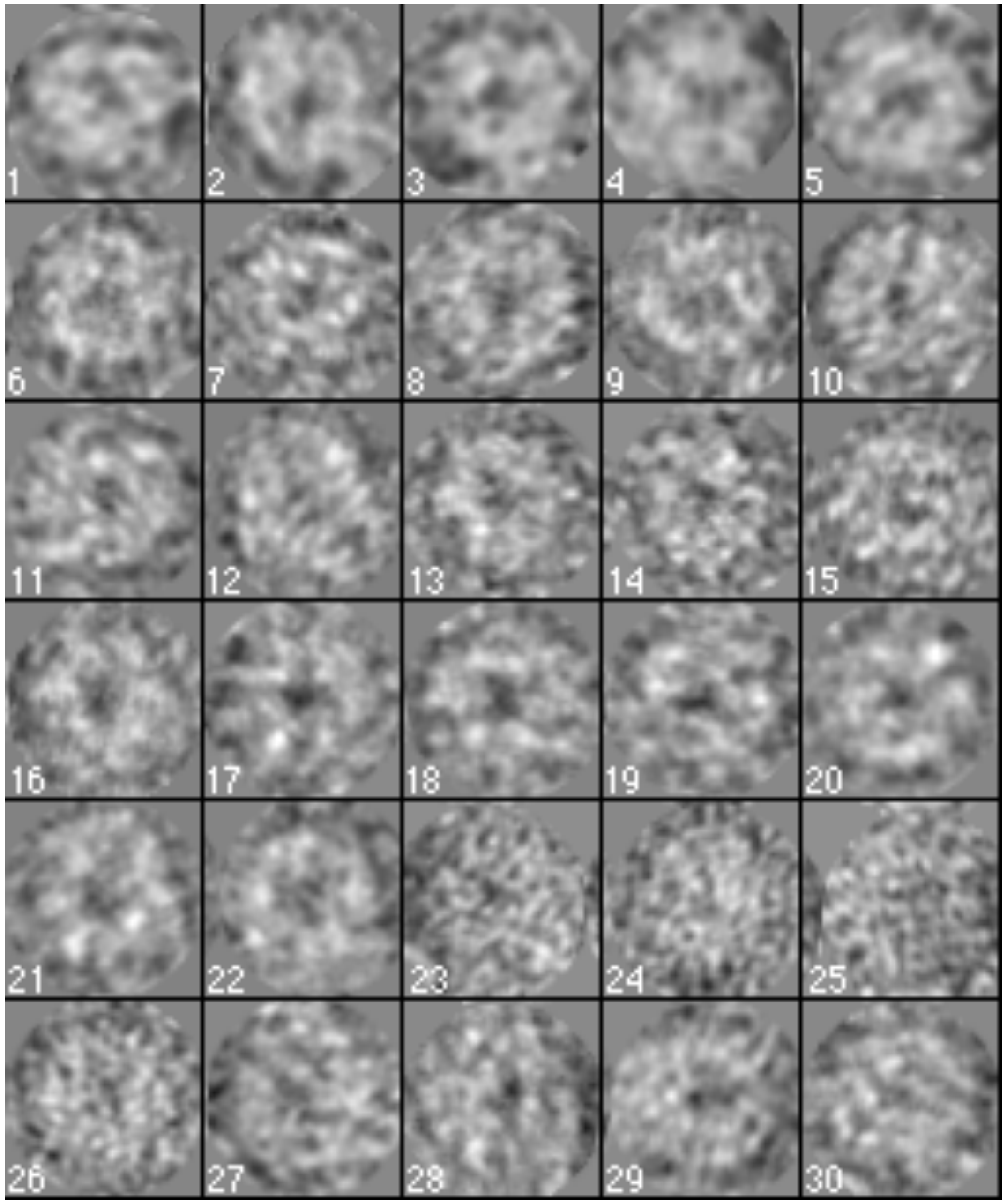
    Exercise: Determine the mean and variance of $\bar{f}[\boldsymbol{k}]$

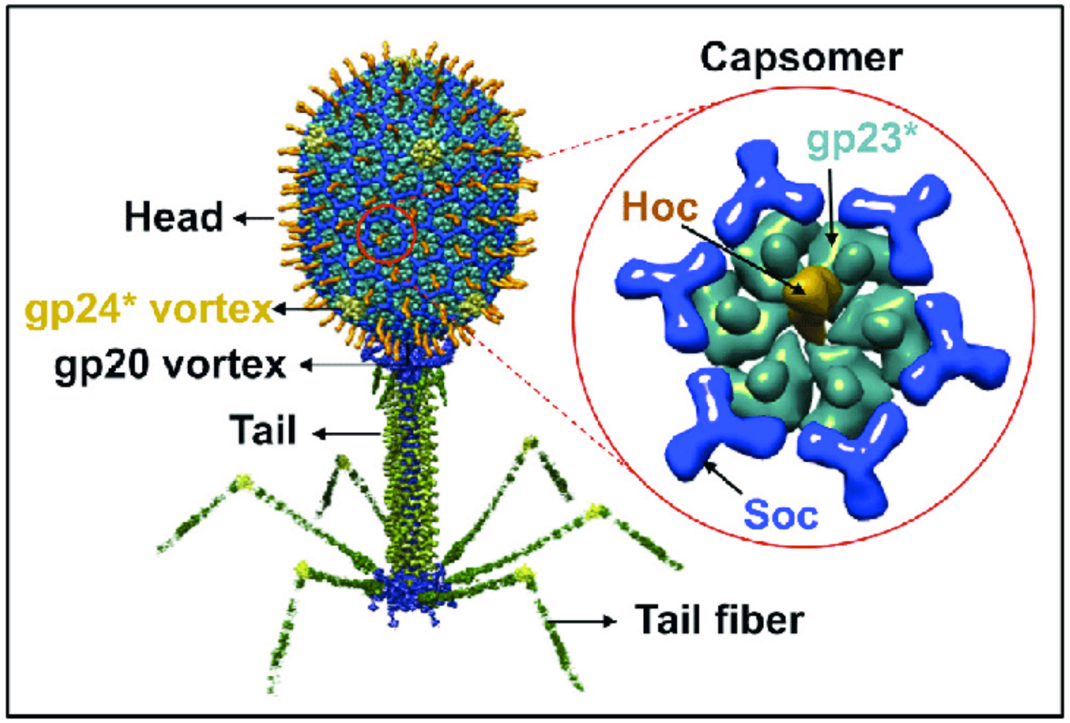
    Mean: $\mathbf{E}[\bar{f}[\boldsymbol{k}]] = s[\boldsymbol{k}]$ Variance: $\mathrm{var}(\bar{f}[\boldsymbol{k}]) = \sigma^2/N$

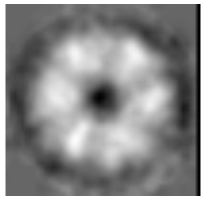
    Central limit theorem: for large N, $\bar{f}[\boldsymbol{k}] \sim \mathcal{N}(s[\boldsymbol{k}], \sigma^2/N)$

Example: Noise Reduction

20 electron micrographs of a virus capsomere





Result of averaging: 

Practical problems

- Image registration
- Detection of outliers

Spatial Averaging: Linear Smoothing

Linear smoothers = Low-pass filters $\qquad g = h * f$ with $\sum_{\boldsymbol{k}} h[\boldsymbol{k}] = 1$

- Finite-impulse response (FIR)

  Moving average

$$\begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & \boxed{1/9} & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix} \qquad \begin{bmatrix} 0 & 1/8 & 0 \\ 1/8 & \boxed{1/2} & 1/8 \\ 0 & 1/8 & 0 \end{bmatrix}$$

- Infinite-impulse response (IIR)
  - Symmetric exponential
  - Gaussian filter
- Main uses
  - noise reductions (high frequencies)
  - estimation of local statistics (mean, variance)

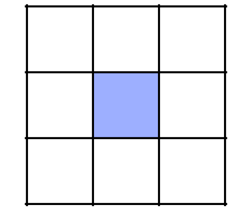
- Limitations
  - Blurring of edges and image details

How do we get around this?

Nonlinear operations

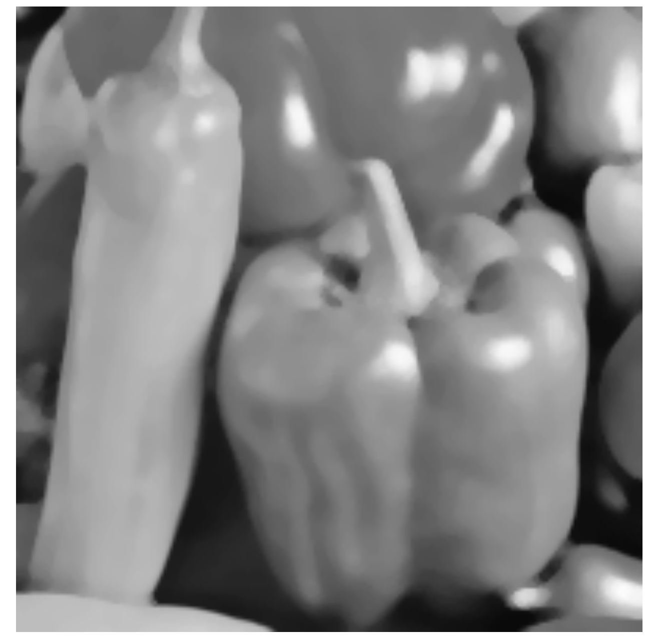
Spatial Averaging: Median Filter

$$g[\boldsymbol{k}] = \mathrm{median}\{f[\boldsymbol{k}-\boldsymbol{n}] \;:\; \boldsymbol{n} \in W\}$$



W

Input (200×200)



5×5 median filtered

- Advantages
  - Tend to preserve contours better than linear smoothers
  - Good for impulsive or heavy-tailed (non-Gaussian) noise
- Limitations
  - Computationally costly for large sizes of neighborhoods
  - Breaks down when there is a majority of noisy pixels

Impulsive-Noise Reduction Experiment

Input (200×200)

Input $+$ impulsive noise

3×3 median

3×3 moving average

5×5 moving average

5×5 median

Matching and Detection

- Template Matching
  - Problem Definition
  - Correlation
- Matched-Filter Detection
- Application Areas
  - Object Detection
  - Automated Inspection
  - Data Fusion
  - Registration
  - Motion Compensation

Template Matching

- Problem definition
  - Reference pattern, target, or template: $f_r[\boldsymbol{k}]$, $\boldsymbol{k} \in \Omega_r$
  - Test image: $f[\boldsymbol{k}]$, $\boldsymbol{k} \in \Omega_f$
  - Common support $\Omega = \Omega_f \cap \Omega_r \neq \varnothing$
  - How do we decide whether or not f and f_r are similar?
  - Given a collection of templates f_i, $i = 1, \ldots, N$ (e.g., shifted versions of some reference template), how do we select the best match?

Exercise: Come up with a concrete instantiation of this sort of problem

Correlation Measures

- Basic correlation

$$\sum_{\boldsymbol{k}\in\Omega} f[\boldsymbol{k}]f_r[\boldsymbol{k}] = \langle f, f_r\rangle$$

$\ell^2(\Omega)$-inner product

How is maximizing the correlation related to the similarity between f and f_r?

Similarity $=$ distance $= \|f - f_r\|_{\ell^2(\Omega)}$

$$\begin{aligned}\|f - f_r\|^2_{\ell^2(\Omega)} &= \langle f - f_r, f - f_r\rangle \\ &= \|f\|^2_{\ell^2(\Omega)} + \|f_r\|^2_{\ell^2(\Omega)} - 2\langle f, f_r\rangle \\ &= \text{constant} - 2\langle f, f_r\rangle\end{aligned}$$

increasing correlation decreases distance

$\|f - f\|^2_{\ell^2(\Omega)}$ is minimum $\Leftrightarrow$ $\langle f, f_r\rangle$ is maximum

Correlation Measures (cont'd)

What if our template and test image have different intensity ranges?

- Centered correlation

$$\langle f - \bar{f}, f_r - \bar{f}_r \rangle = \sum_{\boldsymbol{k} \in \Omega} (f[\boldsymbol{k}] - \bar{f})(f_r[\boldsymbol{k}] - \bar{f}_r)$$

average value

$$\bar{g} = \frac{1}{\#\Omega} \sum_{\boldsymbol{k} \in \Omega} g[\boldsymbol{k}]$$

- Normalized correlation coefficient

$$-1 \leq \rho(f, f_r) = \frac{\langle f - \bar{f}, f_r - \bar{f}_r \rangle}{\|f - \bar{f}\|_{\ell^2(\Omega)} \|f_r - \bar{f}_r\|_{\ell^2(\Omega)}} \leq 1$$

Invariant to linear amplitude scalings: $af + b$

Matched-Filter Detection

- Measurement model (signal + noise): $f[\boldsymbol{k}] = s[\boldsymbol{k} - \boldsymbol{k}_0] + n[\boldsymbol{k}]$

  s: known deterministic template or pattern

  n: additive **white** noise with zero mean and variance σ^2

  $\boldsymbol{k}_0$: unknown template location $\quad\quad \mathbf{E}[f[\boldsymbol{k}]] = s[\boldsymbol{k} - \boldsymbol{k}_0]$

- Goal: Design a correlation-like detector

$$g[\boldsymbol{k}] = (h * f)[\boldsymbol{k}]$$

$$= \underbrace{\sum_{\boldsymbol{n} \in \mathbb{Z}^2} h[\boldsymbol{n}] f[\boldsymbol{k} - \boldsymbol{n}]}_{\text{“convolution”}} = \underbrace{\sum_{\boldsymbol{n} \in \mathbb{Z}^2} w[\boldsymbol{n}] f[\boldsymbol{k} + \boldsymbol{n}]}_{\text{“correlation”}}$$

where $w[\boldsymbol{k}] = h[-\boldsymbol{k}]$

Matched-Filter Detection

- Optimal detector: Maximizes SNR at $\boldsymbol{k} = \boldsymbol{k}_0$

  Solution: $w[\boldsymbol{k}] = s[\boldsymbol{k}]$ (matched filter)

  (technically, $w[\boldsymbol{k}] = \alpha\, s[\boldsymbol{k}]$ is fine, for any $\alpha \in \mathbb{R}$)

  Proof:

"signal" Expected output at $\boldsymbol{k} = \boldsymbol{k}_0$

$$\mathbf{E}[g[\boldsymbol{k}_0]] = \sum_{\boldsymbol{n} \in \mathbb{Z}^2} w[\boldsymbol{n}] s[\boldsymbol{k}_0 - \boldsymbol{k}_0 + \boldsymbol{n}] = \langle w, s \rangle$$

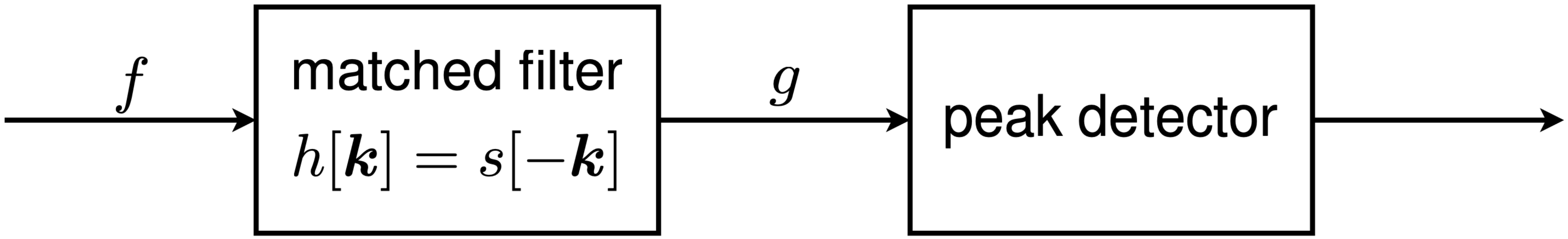
(squared) "noise" Variance output

$$\operatorname{var}(g[\boldsymbol{k}]) = \sum_{\boldsymbol{n} \in \mathbb{Z}^2} w[\boldsymbol{n}]^2 \operatorname{var}(n[\boldsymbol{k} + \boldsymbol{n}]) = \sigma^2 \|w\|_{\ell^2(\mathbb{Z}^2)}^2$$

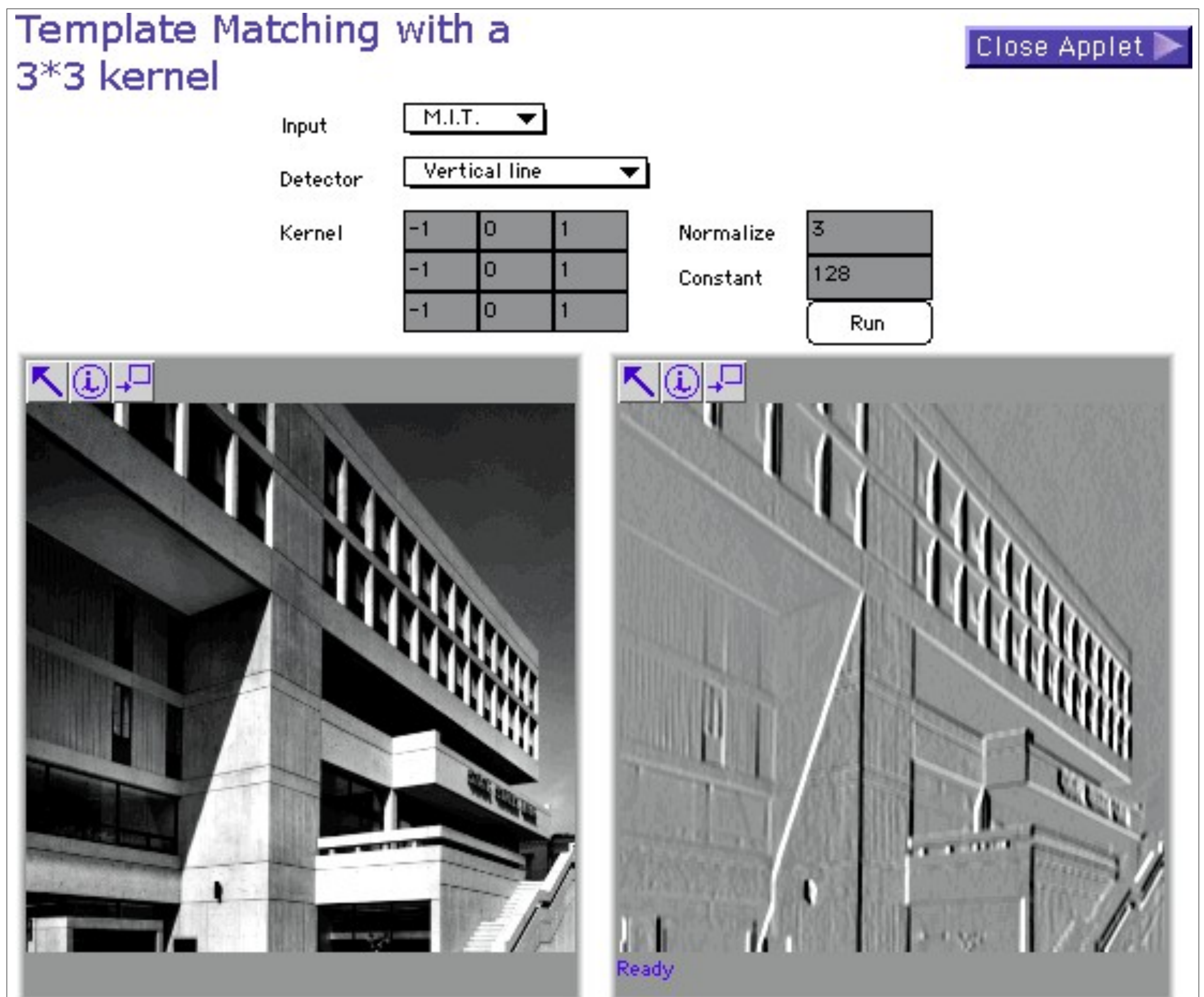
SNR at $\boldsymbol{k} = \boldsymbol{k}_0$: $\text{SNR} = \dfrac{\langle w, s \rangle}{\sigma \|w\|_{\ell^2(\mathbb{Z}^2)}}$

Maximized when $w[\boldsymbol{k}] = \alpha\, s[\boldsymbol{k}]$

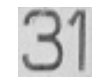
Pattern Detection by Template Matching

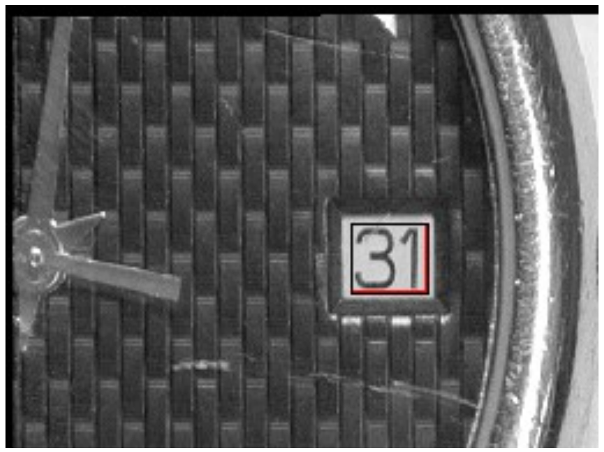


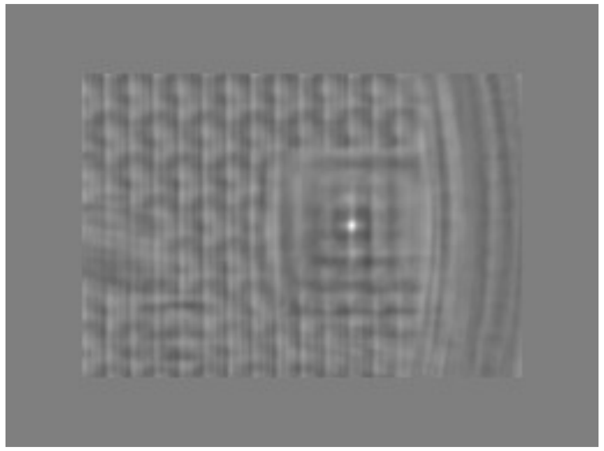
- Application: Line detector



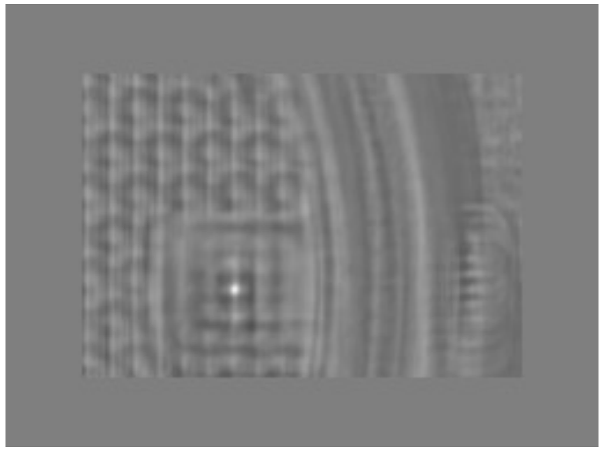
Pattern Detection by Template Matching

Reference template (33×31 pixels) 

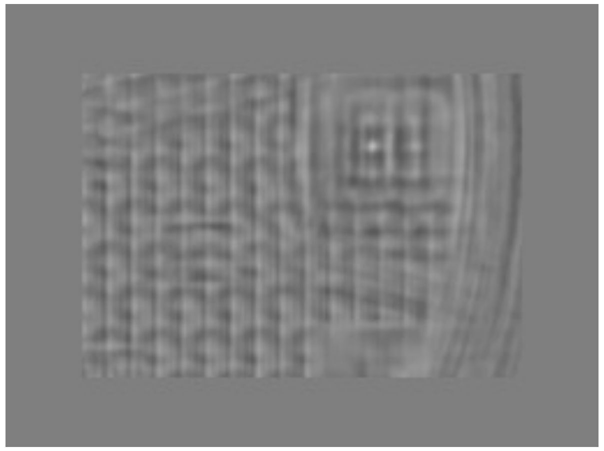




$$(x, y) = (149, 95)$$
$$\rho = 100\%$$



$$(x, y) = (98, 123)$$
$$\rho = 88\%$$



$$(x, y) = (58, 61)$$
$$\rho = 33\%$$

Feature Extraction

- Edge detection

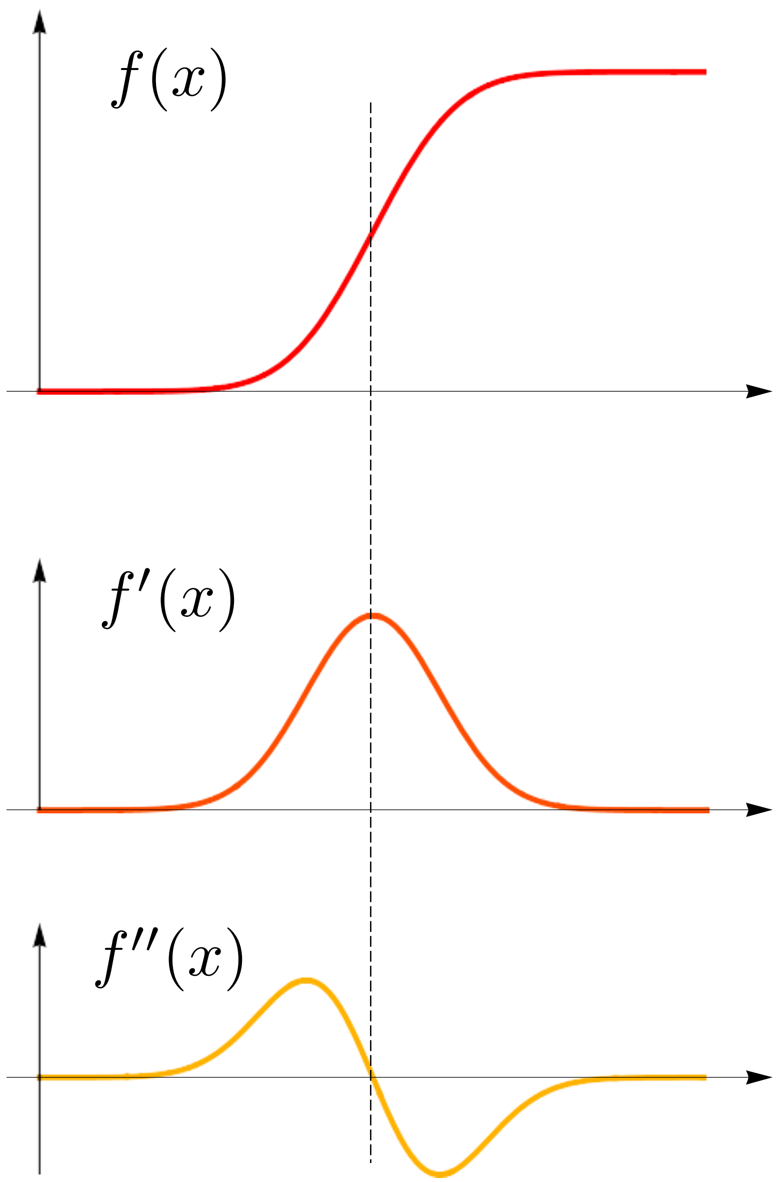
  Edges are important clues for the interpretation of images; they are essential to object recognition

  - Edges: Analog formulation
  - Gradient-based edge detection

Edges: Analog Formulation

What is an edge?

Definition: An edge point is a location of abrupt change in an image



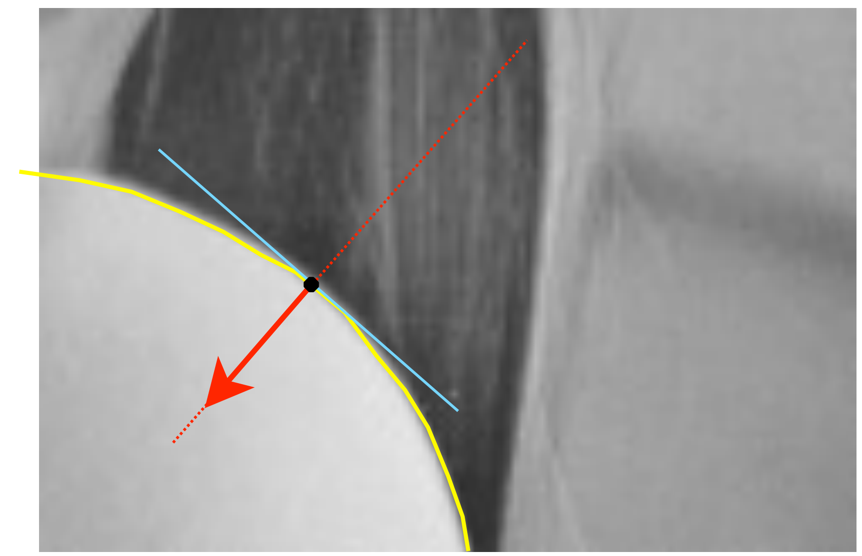


Image value at location $\boldsymbol{x}$: $f(\boldsymbol{x})$

Normal vector: $\boldsymbol{n} = \frac{\nabla f(\boldsymbol{x})}{\|\nabla f(\boldsymbol{x})\|_2}$

$\Rightarrow$ direction of maximum change

Gradient and Directional Derivatives

- Gradient of f at $\boldsymbol{x} = (x, y)$: $\nabla f(\boldsymbol{x}) = \left(\frac{\partial f(\boldsymbol{x})}{\partial x}, \frac{\partial f(\boldsymbol{x})}{\partial y} \right) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}))$

- Directional derivative of f along the unit vector $\boldsymbol{u}_\theta = (\cos\theta, \sin\theta)$

$$\begin{aligned} \mathrm{D}_{\boldsymbol{u}_\theta} f(\boldsymbol{x}) &= \lim_{\varepsilon \to 0} \frac{f(\boldsymbol{x} + \varepsilon \boldsymbol{u}_\theta) - f(\boldsymbol{x})}{\varepsilon} \\ &= f_1(\boldsymbol{x}) \cos\theta + f_2(\boldsymbol{x}) \sin\theta \\ &= \boldsymbol{u}_\theta^\mathsf{T} \nabla f(\boldsymbol{x}) \end{aligned}$$

Taylor-series argument:
$f(\boldsymbol{x} + \varepsilon \boldsymbol{u}) = f(\boldsymbol{x}) + \varepsilon \boldsymbol{u}^\mathsf{T} \nabla f(\boldsymbol{x}) + O(\varepsilon^2)$

Exercise: What is $\max_\theta \mathrm{D}_{\boldsymbol{u}_\theta} f(\boldsymbol{x})$?

$$\max = \mathrm{D}_{\boldsymbol{n}} f(\boldsymbol{x}) = \|\nabla f(\boldsymbol{x})\|_2 = \sqrt{f_1(\boldsymbol{x})^2 + f_2(\boldsymbol{x})^2}$$

$$\theta^* = \angle(\nabla f(\boldsymbol{x})) = \arctan\left(\frac{f_2(\boldsymbol{x})}{f_1(\boldsymbol{x})} \right) + k\pi, k \in \mathbb{Z} \qquad (\perp \text{ to edge})$$

General Criteria for Edge Detection

- Maximum of the gradient
- Zero crossings of the second-order (directional) derivative
- Combination of both

Remarks:

– Gradient magnitude and Laplacian are **rotationally invariant**, while gradient vectors and directional second-order derivatives are not

– Derivatives are usually estimated on a smoothed version of the image to improve robustness and/or reduce the effect of noise

Gradient-Based Edge Detection

How do we design discrete filters that mimic gradients?

- Discretized gradient operators

  Horizontal derivative: $g_1[\boldsymbol{k}] = (h_1 * f)[\boldsymbol{k}]$

  Vertical derivative: $g_2[\boldsymbol{k}] = (h_2 * f)[\boldsymbol{k}]$

  $$g[\boldsymbol{k}] = \sqrt{g_1[\boldsymbol{k}]^2 + g_2[\boldsymbol{k}]^2}$$

  $$\theta[\boldsymbol{k}] = \arctan\left(\frac{g_2[\boldsymbol{k}]}{g_1[\boldsymbol{k}]}\right)$$

  $$\partial_x \approx \begin{array}{|c|c|c|}\hline \frac{1}{2} & 0 & -\frac{1}{2} \\ \hline \end{array}$$

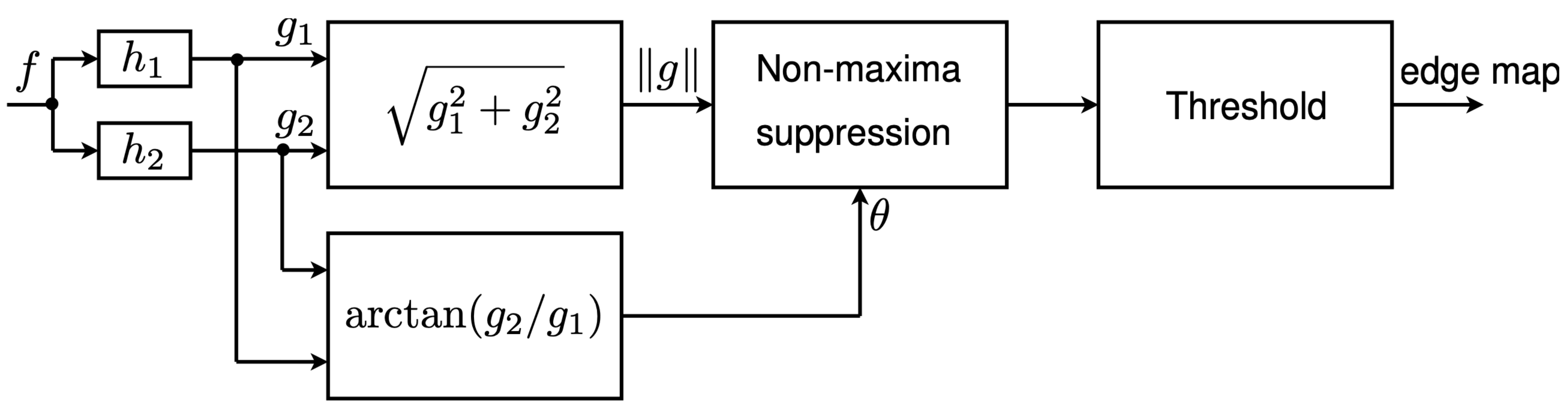
  $$\partial_y \approx \begin{array}{|c|}\hline \frac{1}{2} \\ \hline 0 \\ \hline -\frac{1}{2} \\ \hline \end{array}$$

- Threshold-based edge detection

  $$\mathrm{edge}[\boldsymbol{k}] = \begin{cases} 1, & g[k_1, k_2] \geq T \\ 0, & \text{else} \end{cases}$$

Canny's Edge Detection Algorithm

- Refinements:
  - Non-maxima suppression: Using knowledge of $\theta[\boldsymbol{k}]$
  - Intelligent thresholding



https://bigwww.epfl.ch/demo/ip/demos/edgeDetector/

Image Segmentation

- Segmentation: Art or Science?
- Amplitude Thresholding
  - Variational Thresholding
  - Statistical Thresholding
- Binary Segmentation Techniques

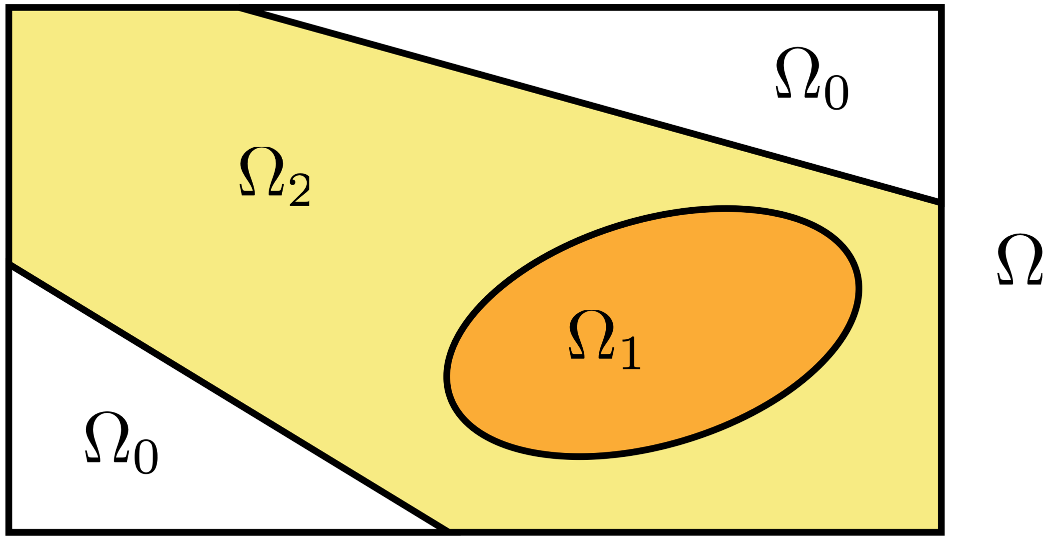
What is Segmentation?

- Definition

Image $f[\boldsymbol{k}]$, with $\boldsymbol{k} \in \Omega$

Image segmentation: Find a partition of the support Ω of the image f, with

$$\Omega = \bigcup_i \Omega_i \quad \text{with} \quad \Omega_i \cap \Omega_j = \varnothing \text{ for } i \neq j$$



such that the regions Ω_i satisfy some homogeneity (and connectivity) criterion.

The total number of regions is not necessarily known

- Three main approaches (not based on deep learning)
  - Pixel classification
  - Region-based segmentation
  - Boundary-based segmentation $\Rightarrow$ Edge detection

Segmentation: Art or Science?

Problem: lack of a universal definition of homogeneity
$\Rightarrow$ many application-specific approaches

- Approaches for specifying homogeneity
  - Empirical (e.g., similar graylevels; feature maps)
  - Statistical, based on some a priori model
    (e.g., constant mean + additive white noise)
- Approaches for enforcing connectivity (if required)
  - Prior information about object size or shape
  - Joint probability model for class labels
  - Contour length