ECE 172A: Introduction to Image Processing
Directional Image Analysis and Processing

Rahul Parhi
Assistant Professor, ECE, UCSD

Winter 2025

Outline

- Mathematical Foundations
  - Rotations and the Fourier Transform
  - Radon Transform
  - Rotation of Polynomials
  - Directional Derivatives
- Local Directional Analysis
  - Structure Tensor
- Steerable Filters
  - Derivative-Based Filters
- Edge Detector: Revisited

Directionality in Image Processing

- Importance of directional cues
  - Edges, ridges, patterns, texture
  - Visual perception is orientation-sensitive
  - Neurons in the primary visual cortex have orientation selectivity

    (Hubel and Wiesel, 1958)

- Invariant Processing and Feature Detection
  - Invariant operators: Gradient magnitude, Laplacian, . . .

- Computational challenges
  - Selectivity to orientation
  - Steerability (orientation can be arbitrary)
  - Separable filters are not orientation-sensitive

Mathematical Foundations

- Rotations and the Fourier Transform
- Radon Transform
- Rotation of Polynomials
- Directional Derivatives

Rotations and the Fourier Transform

Recall: Continuous-domain Fourier transform $\hat{f}(\boldsymbol{\omega}) = \int_{\mathbb{R}^2} f(\boldsymbol{x}) \mathrm{e}^{-\mathrm{j}\boldsymbol{\omega}^\mathsf{T}\boldsymbol{x}} \, \mathrm{d}\boldsymbol{x}$

Spatial-domain rotations correspond to what in the Fourier domain?

$$\mathbf{R}_\theta = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

counterclockwise rotation

$$f(\boldsymbol{x}) \quad \xleftrightarrow{\mathcal{F}} \quad \hat{f}(\boldsymbol{\omega})$$

$$f(\mathbf{R}_\theta \boldsymbol{x}) \quad \xleftrightarrow{\mathcal{F}} \quad \hat{f}(\mathbf{R}_\theta \boldsymbol{\omega})$$ (proof by change-of-variables)

Spatial-domain rotations correspond to Fourier-domain rotations

Radon Transform

The Radon transform of $f(x, y)$ corresponds to all line integrals of $f(x, y)$

Notation: $\boldsymbol{\theta} = (\cos\theta, \sin\theta) \in \mathbb{R}^2$

A line in $\mathbb{R}^2$ can be represented by all $\boldsymbol{x} \in \mathbb{R}^2$ such that

$$\boldsymbol{\theta}^\mathsf{T}\boldsymbol{x} = t \qquad \Leftrightarrow \qquad x\cos\theta + y\sin\theta = t$$

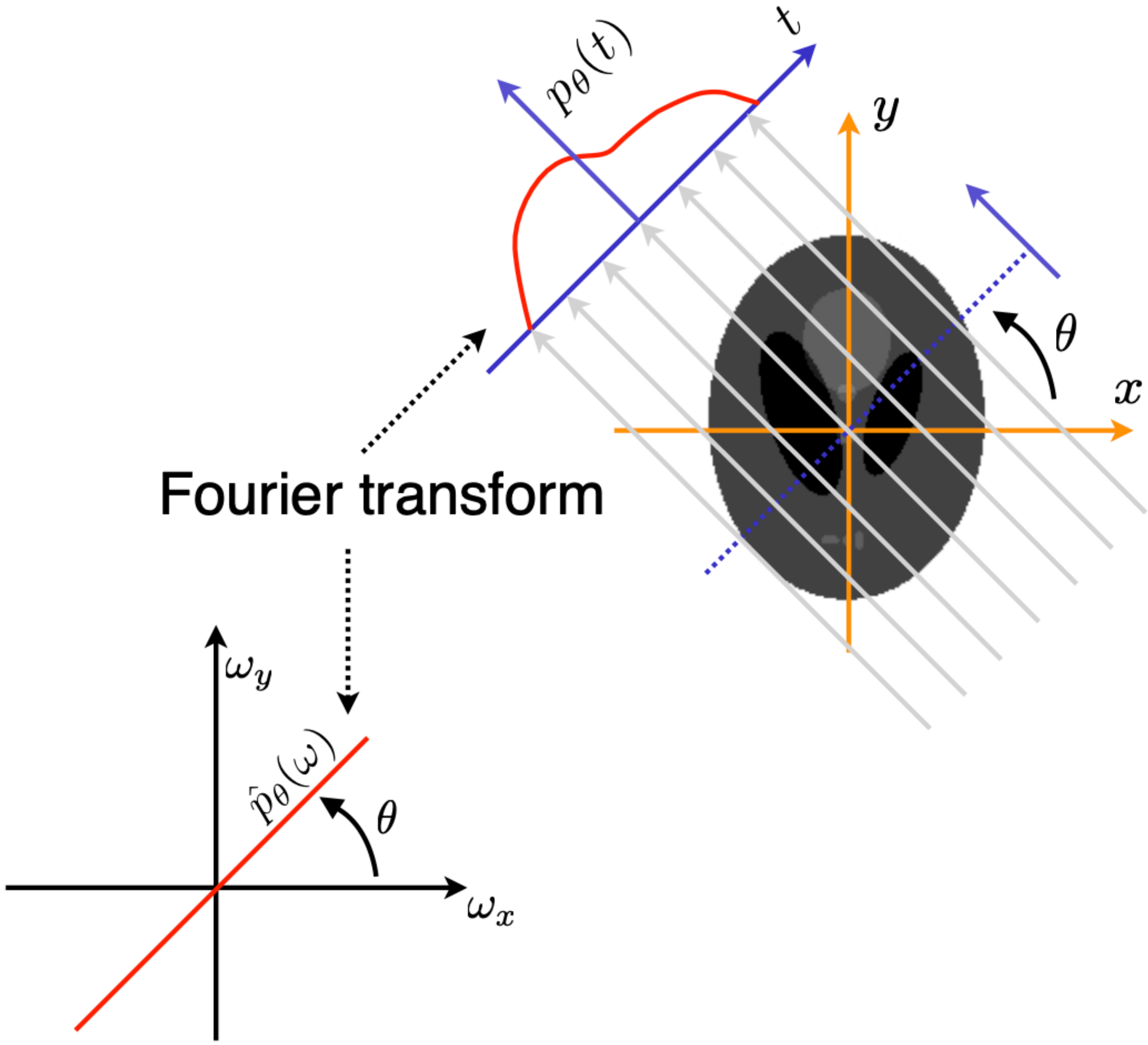
$$p_\theta(t) = \mathscr{R}\{f\}(\boldsymbol{\theta}, t) = \int_{\mathbb{R}^2} f(\boldsymbol{x})\delta(\boldsymbol{\theta}^\mathsf{T}\boldsymbol{x} - t)\,\mathrm{d}\boldsymbol{x}$$

How can we even compute this?

Theorem: Fourier slice theorem

$$\hat{p}_\theta(\omega) = \hat{f}(\omega\cos\theta, \omega\sin\theta)$$

Radon Transform



Steerability of Polynomials

Property: The rotated version of a 2D polynomial of degree p is a 2D polynomial of degree p. This implies that polynomials are "steerable".

Why is this useful?

How do we establish this property?

Key observations for establishing this property:

- A 2D polynomial of degree p is a linear combination of monomials of degree $n \leq p$: $x^{k_1} y^{k_2}$ with $k_1 + k_2 = n$
- A rotation of a monomial of degree k yields a polynomial of degree k

A rotation of a polynomial is a polynomial of the same degree

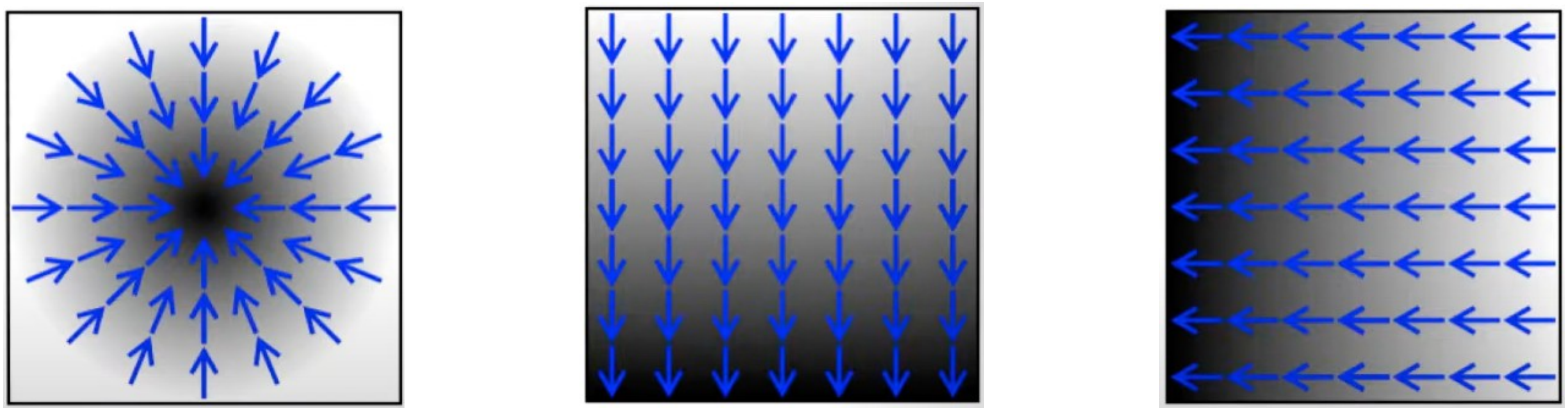
Gradient and Directional Derivatives

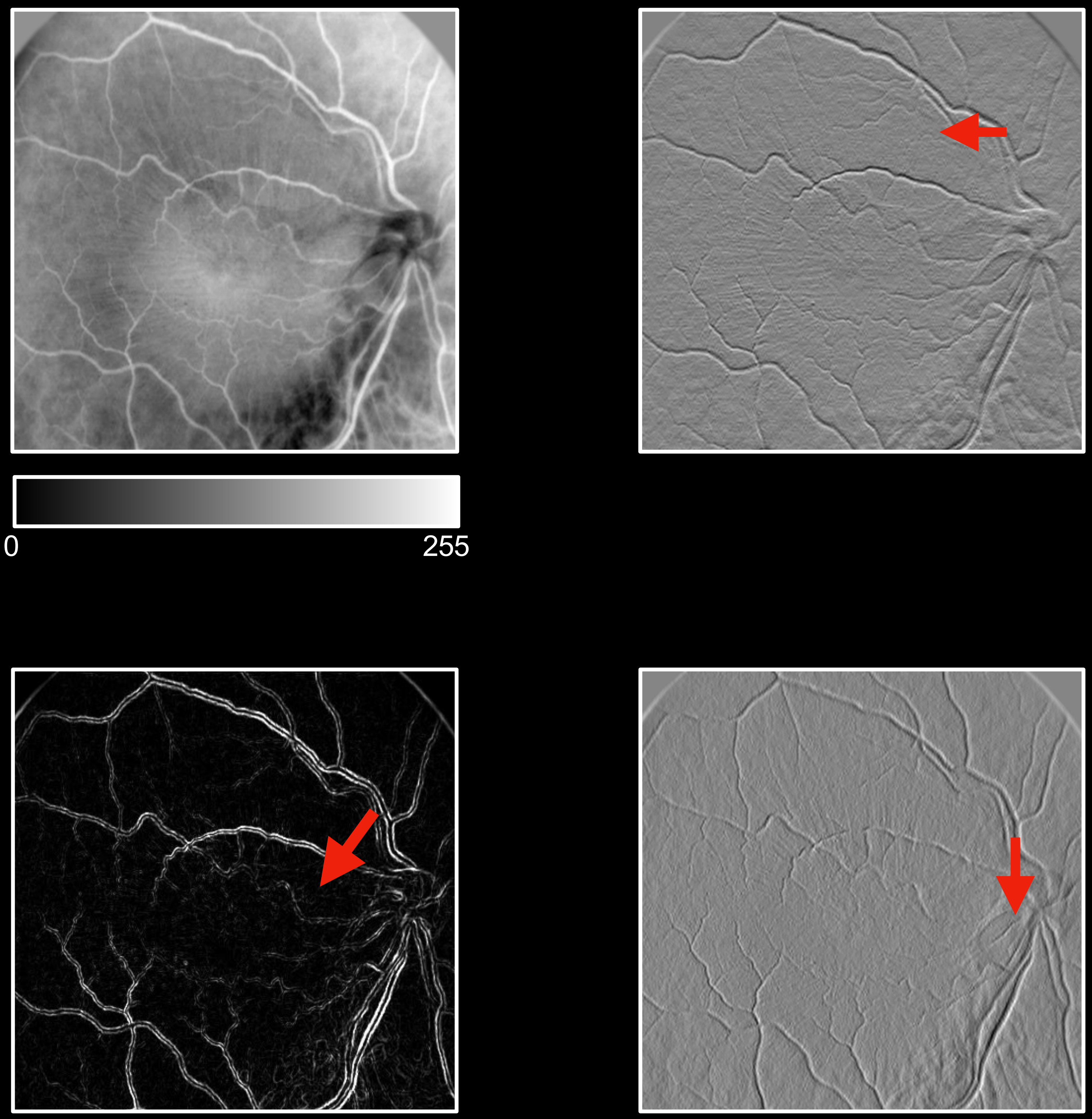
Direction specified by $\boldsymbol{u} \in \mathbb{R}^2$ with $\|\boldsymbol{u}\|_2 = 1$ (unit vector)

- First-order directional derivatives

$$
\begin{aligned}
\mathrm{D}_{\boldsymbol{u}} f(\boldsymbol{x}) &= \lim_{h \to 0} \frac{f(\boldsymbol{x}) - f(\boldsymbol{x} - h\boldsymbol{u})}{h} \\
&= \boldsymbol{u}^{\mathsf{T}} \nabla f(\boldsymbol{x}) && \xleftrightarrow{\mathcal{F}} && \mathrm{j}\boldsymbol{u}^{\mathsf{T}} \boldsymbol{\omega} \hat{f}(\boldsymbol{\omega}) \\
&= u_1 \frac{\partial f(\boldsymbol{x})}{\partial x} + u_2 \frac{\partial f(\boldsymbol{x})}{\partial y} && \xleftrightarrow{\mathcal{F}} && \mathrm{j}(u_1\omega_1 + u_2\omega_2) \hat{f}(\boldsymbol{\omega})
\end{aligned}
$$

$\boldsymbol{n}(\boldsymbol{x}) = \dfrac{\nabla f(\boldsymbol{x})}{\|\nabla f(\boldsymbol{x})\|_2}$ maximizes the directional derivative




0
255

Higher-Order Directional Derivatives

Direction specified by $\boldsymbol{u} \in \mathbb{R}^2$ with $\|\boldsymbol{u}\|_2 = 1$ (unit vector)

- Directional derivative of order n

$$\mathrm{D}^n_{\boldsymbol{u}} f(\boldsymbol{x}) = \underbrace{\mathrm{D}_{\boldsymbol{u}} \mathrm{D}_{\boldsymbol{u}} \cdots \mathrm{D}_{\boldsymbol{u}}}_{n \text{ times}} f(\boldsymbol{x}) \quad \xleftrightarrow{\mathcal{F}} \quad (\mathrm{j}\boldsymbol{u}^{\mathsf{T}}\boldsymbol{\omega})^n \hat{f}(\boldsymbol{\omega})$$

Exercise: Let $\boldsymbol{u}_\theta = (\cos\theta, \sin\theta)$. Explicitly determine $\mathrm{D}^2_{\boldsymbol{u}_\theta} f(\boldsymbol{x})$ as a function of θ and partial derivatives of f.

Directional Image Analysis

- Structure Tensor
- Implementation
- Examples of 2D Directional Analysis

Structure Tensor

- Structure tensor at location $\boldsymbol{x}_0$

$$\mathrm{J}(\boldsymbol{x}_0) = \int_{\mathbb{R}^2} w(\boldsymbol{x} - \boldsymbol{x}_0)\, \nabla f(\boldsymbol{x}) \nabla f(\boldsymbol{x})^{\mathsf{T}}\, \mathrm{d}\boldsymbol{x}$$

  - $w(\boldsymbol{x})$: nonnegative symmetric "observation window" (e.g., Gaussian)
  - $\mathbf{J}$: 2×2 symmetric matrix

Why are the eigenvalues real?

  - Eigenvectors and eigenvalues: $\mathbf{J}\boldsymbol{u}_i = \lambda_i \boldsymbol{u}_i$, $i = 1, 2$ with $\lambda_1 \geq \lambda_2$

- Interpretation for window centered at $\boldsymbol{x}_0 = \mathbf{0}$
  - Weighted inner product

$\mathbf{J} = \langle \nabla f, \nabla f \rangle_w$

e.g., $[\mathbf{J}]_{1,1} = \left\langle \frac{\partial f}{\partial x}, \frac{\partial f}{\partial x} \right\rangle_w$ with $\langle f_1, f_2 \rangle_w = \int_{\mathbb{R}^2} w(\boldsymbol{x}) f_1(\boldsymbol{x}) f_2(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x}$

  - Energy of u-directional derivative

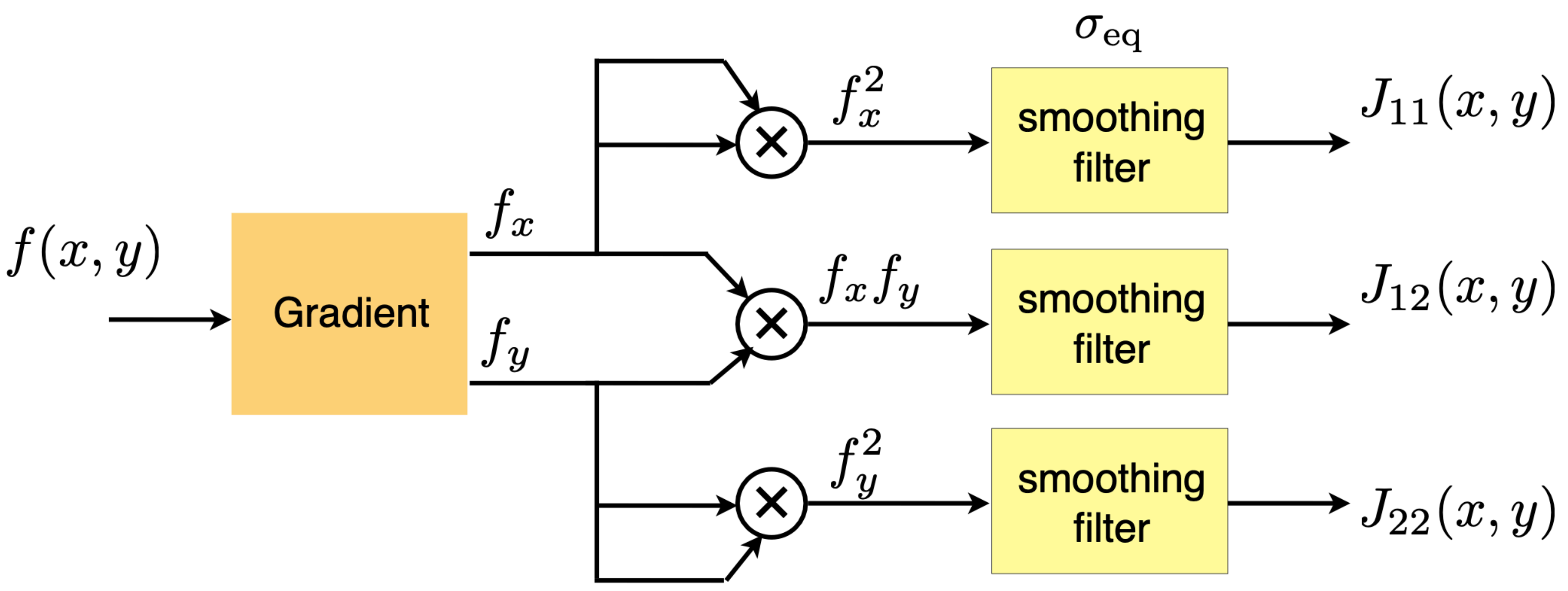
$$\|\mathrm{D}_{\boldsymbol{u}} f\|_w^2 = \langle \boldsymbol{u}^{\mathsf{T}} \nabla f, \boldsymbol{u}^{\mathsf{T}} \nabla f \rangle_w = \boldsymbol{u}^{\mathsf{T}} \langle \nabla f, \nabla f \rangle_w \boldsymbol{u} = \boldsymbol{u}^{\mathsf{T}} \mathbf{J} \boldsymbol{u}$$

  - Dominant orientation of a neighborhood: $\boldsymbol{u}_1 = \operatorname{argmax}_{\|\boldsymbol{u}\|=1} \|\mathrm{D}_{\boldsymbol{u}} f\|_w^2$

Eigenvalues: $\lambda_i = \boldsymbol{u}_i^{\mathsf{T}} \mathbf{J} \boldsymbol{u}_i$

Structure Tensor Implementation

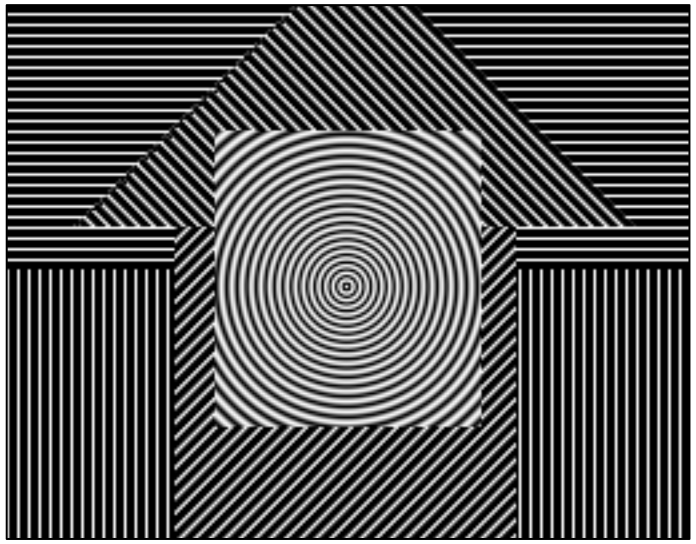
Exercise: How would you implement the structure tensor?



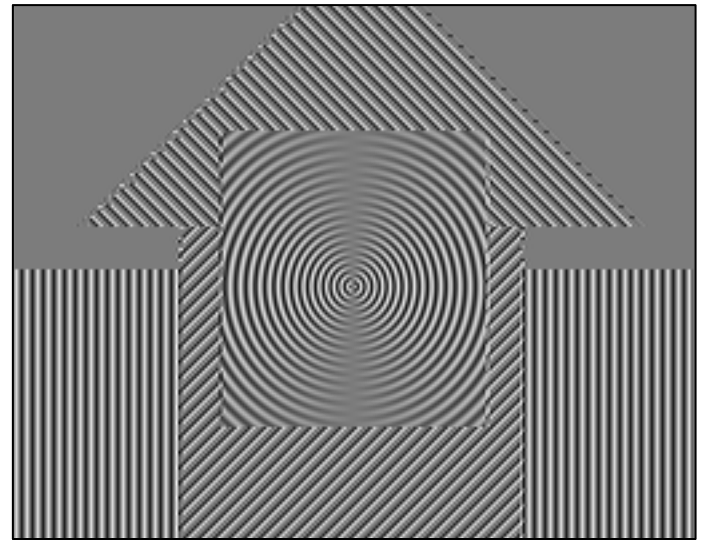
$$\mathbf{J}(x,y) = \begin{bmatrix} J_{11}(x,y) & J_{12}(x,y) \\ J_{12}(x,y) & J_{22}(x.y) \end{bmatrix}$$

- Structure tensor allows us to understand **local features**
  - Gradient "energy": $E = \mathrm{trace}(\mathbf{J}) = J_{11} + J_{22}$
  - Orientation: $\boldsymbol{u}_1 = (\cos\theta, \sin\theta)$ with $\theta = \frac{1}{2}\arctan\left(\frac{2J_{12}}{J_{22} - J_{11}}\right)$
  - Coherency: $0 \leq C = \frac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2} = \frac{\sqrt{(J_{22} - J_{11})^2 + 4J_{12}^2}}{J_{22} + J_{11}} \leq 1$
  - Harris corner index: $H = \det(\mathbf{J}) - \kappa\,\mathrm{trace}(\mathbf{J})^2$ with $\kappa \in [0.04, 0.06]$

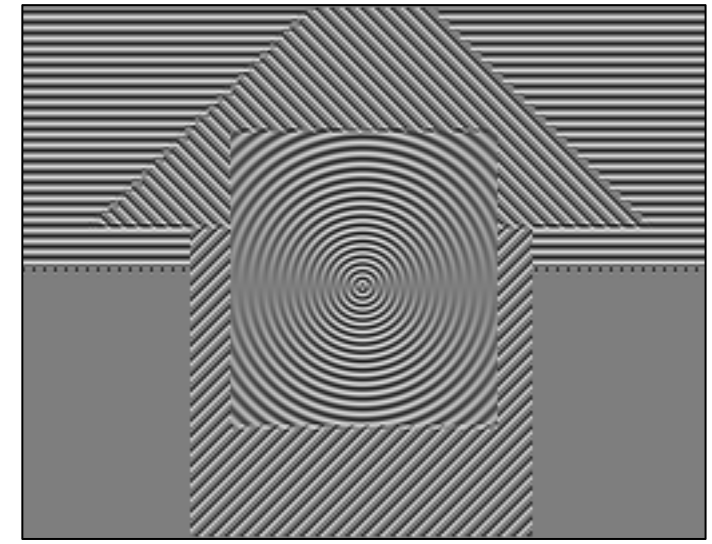
Examples of Directional Analysis



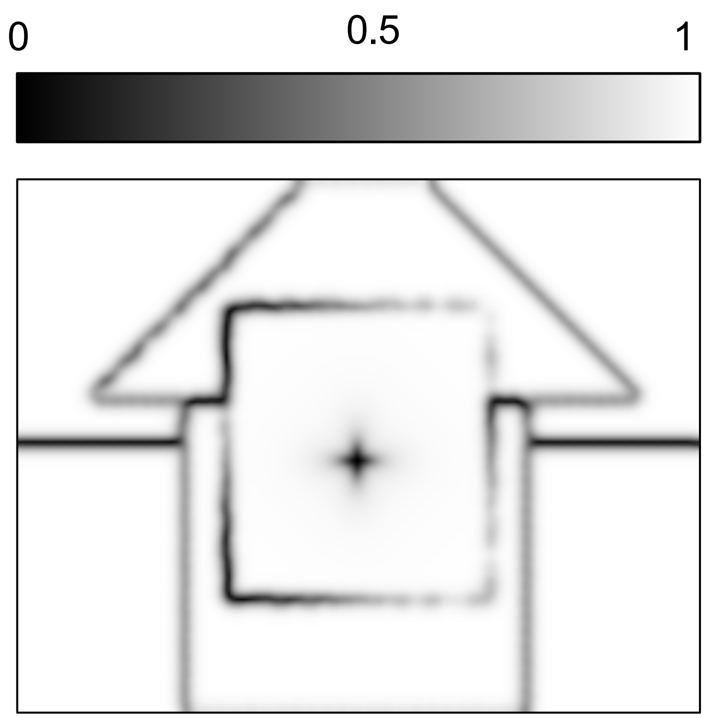
Input



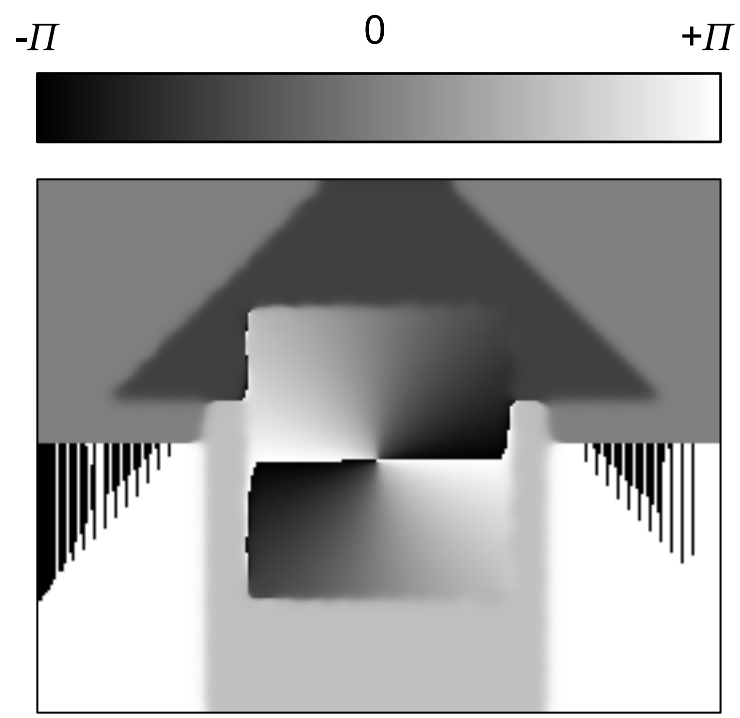
Dx



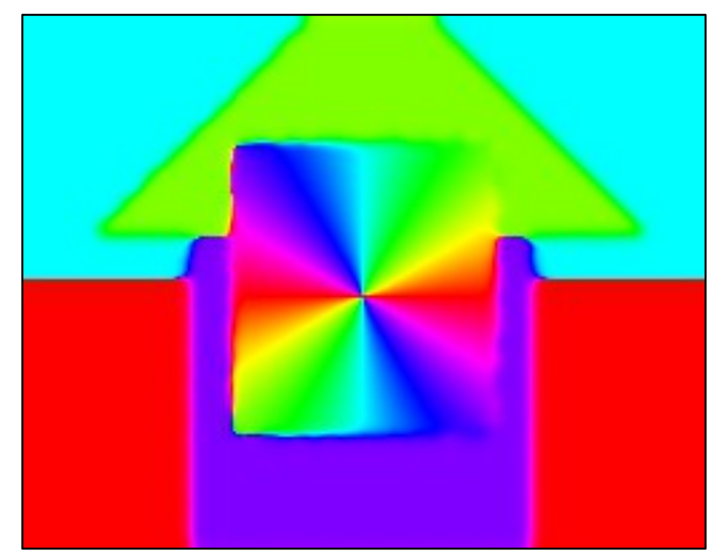
Dy



Coherency



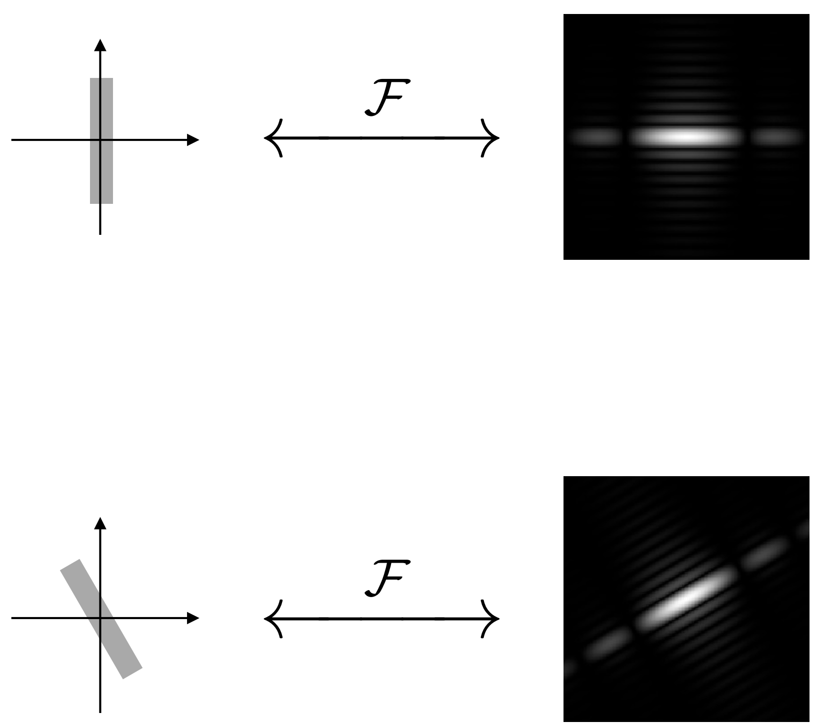
Orientation



Orientation

Orientation Estimation: Revisited

- **Problem:** Design a (real time?) system that can determine the orientation of a (linear) pattern placed at an arbitrary location in an image.



$$g_\theta(\boldsymbol{x}) = f(\mathbf{R}_\theta \boldsymbol{x})$$

$$\mathbf{R}_\theta = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

$$g_\theta(\boldsymbol{x}) \xleftrightarrow{\mathcal{F}} \hat{f}(\mathbf{R}_\theta \boldsymbol{\omega})$$

We want to find the orientation in the Fourier domain with the **least spread**.

Problem Solution

- Compute the "Fourier inertia" matrix (second-moment matrix)

$$\mathbf{M} = \begin{bmatrix} \iint \omega_1^2 |\hat{f}(\boldsymbol{\omega})|^2 \, \mathrm{d}\omega_1 \, \mathrm{d}\omega_2 & \iint \omega_1 \omega_2 |\hat{f}(\boldsymbol{\omega})|^2 \, \mathrm{d}\omega_1 \, \mathrm{d}\omega_2 \\ \iint \omega_2 \omega_1 |\hat{f}(\boldsymbol{\omega})|^2 \, \mathrm{d}\omega_1 \, \mathrm{d}\omega_2 & \iint \omega_2^2 |\hat{f}(\boldsymbol{\omega})|^2 \, \mathrm{d}\omega_1 \, \mathrm{d}\omega_2 \end{bmatrix}$$

$$= \begin{bmatrix} \langle \mathrm{j}\omega_1 \hat{f}(\boldsymbol{\omega}), \mathrm{j}\omega_1 \hat{f}(\boldsymbol{\omega}) \rangle & \langle \mathrm{j}\omega_1 \hat{f}(\boldsymbol{\omega}), \mathrm{j}\omega_2 \hat{f}(\boldsymbol{\omega}) \rangle \\ \langle \mathrm{j}\omega_2 \hat{f}(\boldsymbol{\omega}), \mathrm{j}\omega_1 \hat{f}(\boldsymbol{\omega}) \rangle & \langle \mathrm{j}\omega_2 \hat{f}(\boldsymbol{\omega}), \mathrm{j}\omega_2 \hat{f}(\boldsymbol{\omega}) \rangle \end{bmatrix}$$

Second-order moments measure spread

$$= (2\pi)^2 \begin{bmatrix} \langle \partial_x f, \partial_x f \rangle & \langle \partial_x f, \partial_y f \rangle \\ \langle \partial_y f, \partial_x f \rangle & \langle \partial_y f, \partial_y f \rangle \end{bmatrix}$$ (fast algorithm via Parseval-Plancherel)

Which direction will have the least spread?

The direction of the **smallest eigenvalue**

Problem Solution (cont'd)

- Eigendecomposition of $\mathbf{M}$ gives us the **axes of inertia**

$\lambda_1 \geq \lambda_2$

$$\begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} = \begin{bmatrix} \boldsymbol{u}_1^\top \\ \boldsymbol{u}_2^\top \end{bmatrix} \mathbf{M} \begin{bmatrix} \boldsymbol{u}_1 & \boldsymbol{u}_2 \end{bmatrix}$$

$\boldsymbol{u}_1$: eigenvector in the direction of the **long** axis
$\boldsymbol{u}_2$: eigenvector in the direction of the **short** axis

- Pipeline:

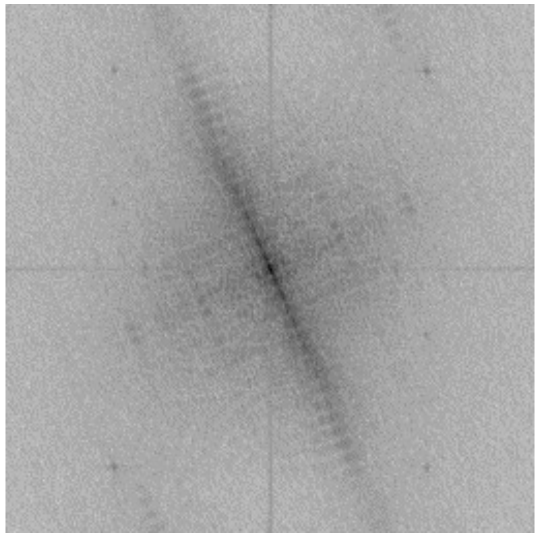
1. Compute the Fourier inertia matrix $\mathbf{M}$ via the fast algorithm
2. Compute the eigendecomposition of $\mathbf{M}$ and store $\boldsymbol{u}_2$
3. Return the angle of $\boldsymbol{u}_2$
   - $\theta = \arctan \dfrac{u_{22}}{u_{21}}$

Orientation Estimation in Action

- Image 1:

  Measured angle: $25° \pm 2°$

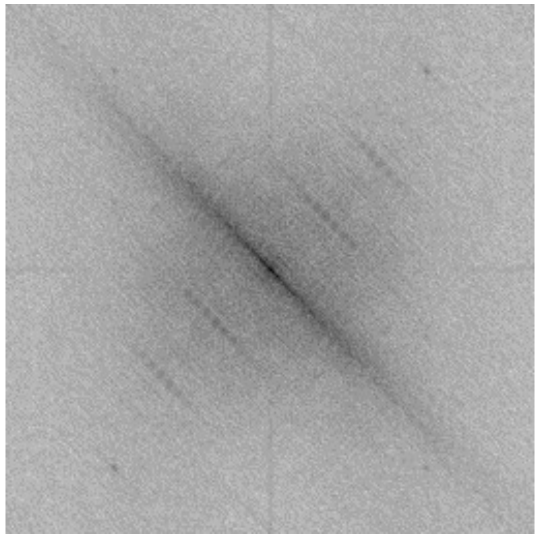
  Computed angle: $27°$

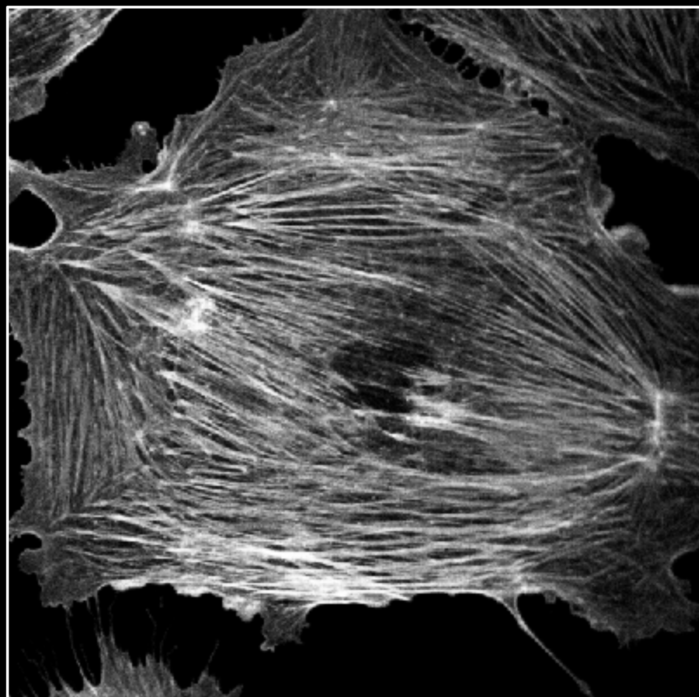


- Image 2:

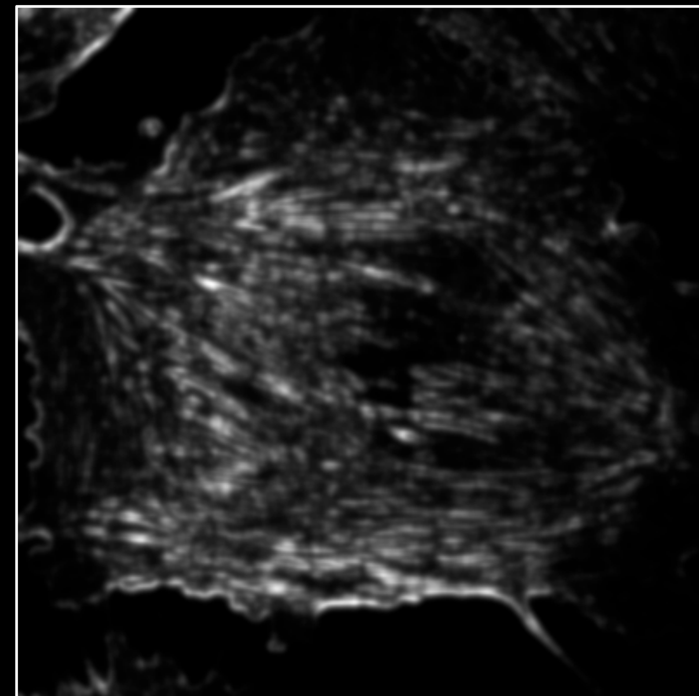
  Measured angle: $44° \pm 2°$

  Computed angle: $45.6°$



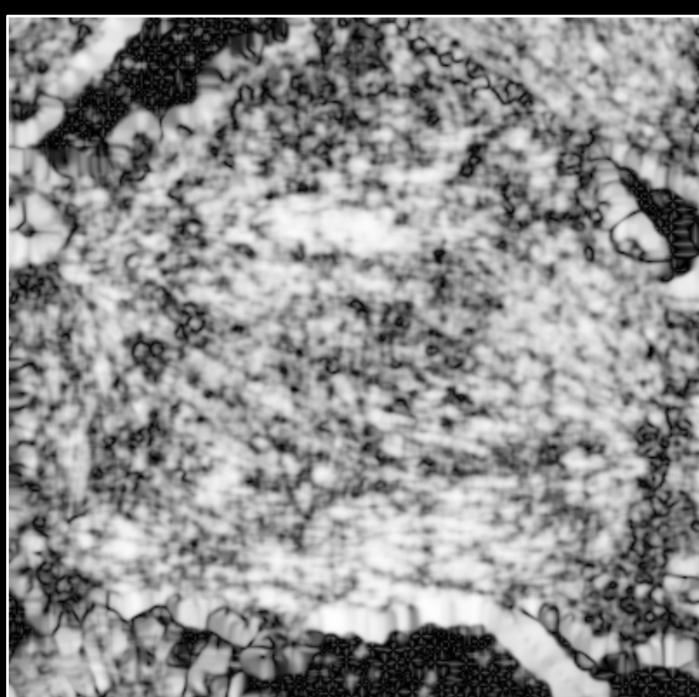


Input

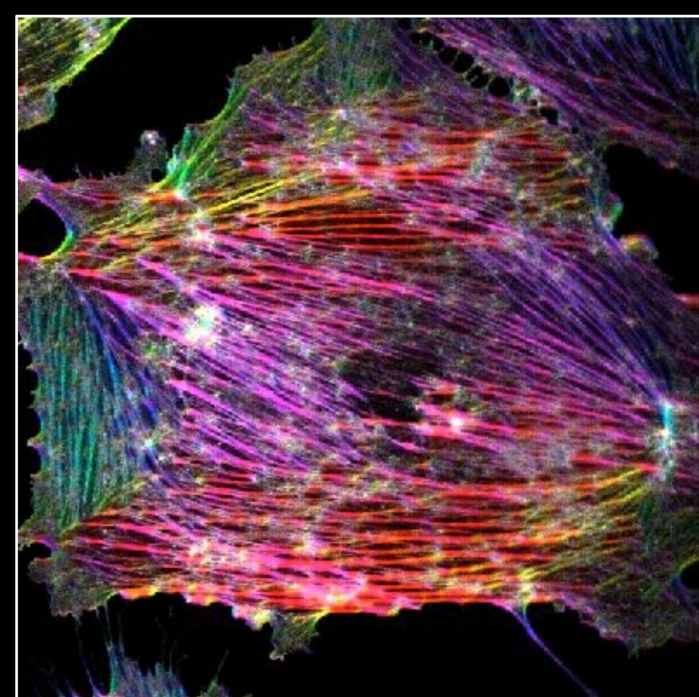


Energy

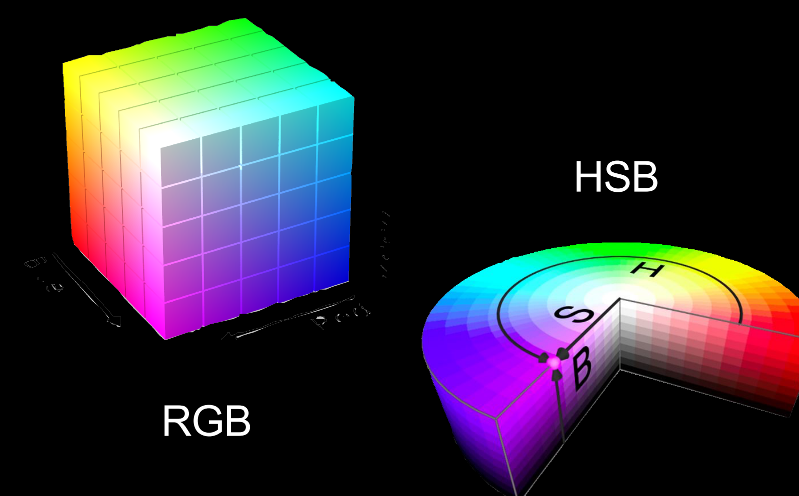
Orientation



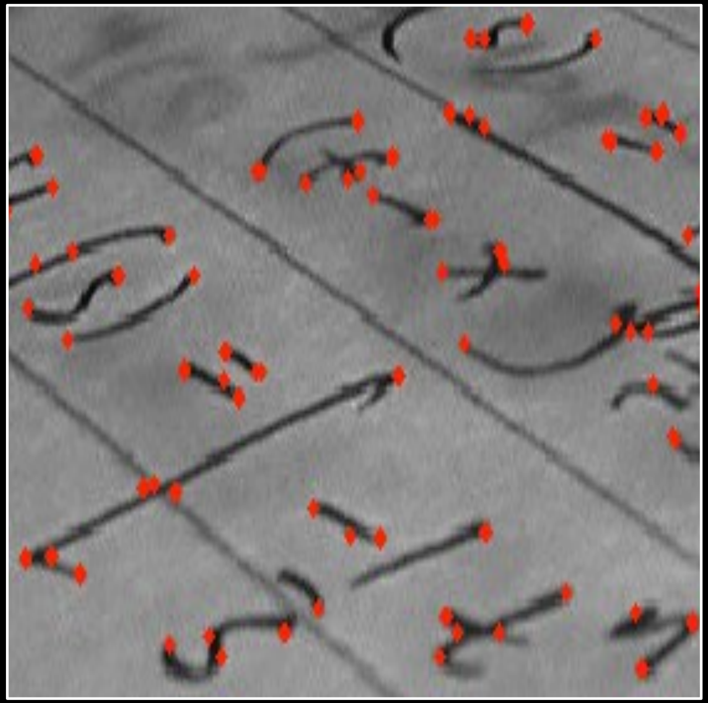
Coherency



Color HSB representation



Hue: orientation
Saturation: coherency
Brightness: input



Keypoints detector
(Harris Corner)

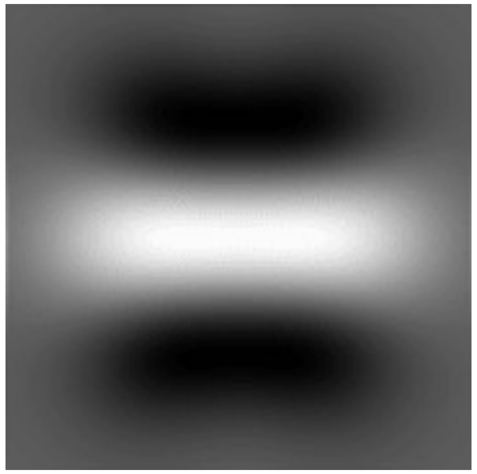
Steerable Filters

- Directional Pattern Matching
- Steerable Filters
- Derivative Filters

Directional Pattern Matching

Task: detection/enhancement of a given type of directional pattern
Example: edge, line, ridge, filament, corner, etc.

- Measurement model (signal + noise): $f(\boldsymbol{x}) = I f_0(\mathbf{R}_\theta(\boldsymbol{x} - \boldsymbol{x}_0)) + n(\boldsymbol{x})$
  - $f_0(\boldsymbol{x})$: template (e.g., elongated blob)
  - $\boldsymbol{x}_0$: spatial location (unknown)
  - $\mathbf{R}_\theta$: 2×2 rotation by θ (unknown)
  - I: intensity (unknown)
  - $n(\boldsymbol{x})$: additive white Gaussian noise



Have we seen this problem before?

- Maximum-likelihood estimator (rotating matched filter)

  Define $h(\boldsymbol{x}) = f_0(-\boldsymbol{x})$ and $h_\theta(\boldsymbol{x}) = h(\mathbf{R}_\theta \boldsymbol{x})$

  $\widetilde{\theta}(\boldsymbol{x}) = \mathrm{argmax}_\theta (f * h_\theta)(\boldsymbol{x})$

  $\widetilde{I}(\boldsymbol{x}) = (f * h_{\tilde{\theta}(\boldsymbol{x})})(\boldsymbol{x})$

Why is this approach bad?

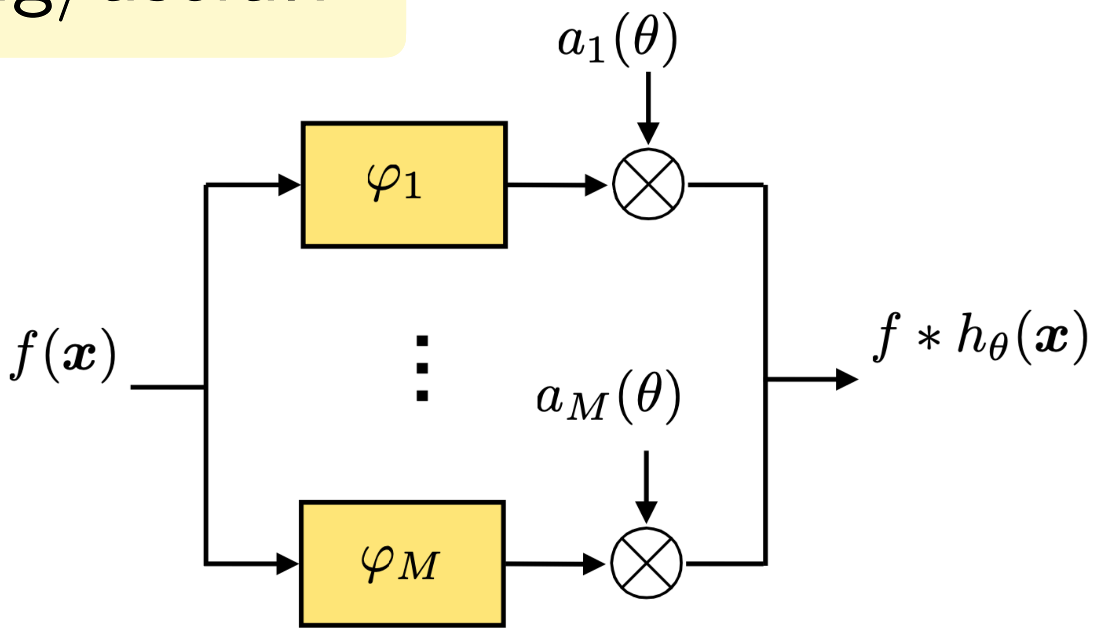
computationally expensive

Definition: A 2D filter $h(\boldsymbol{x})$, $\boldsymbol{x} \in \mathbb{R}^2$ is sterrable of order M if and only if there exist "basis filters" $\varphi_m(\boldsymbol{x})$ and coefficients $a_m(\theta)$ such that

$$h_\theta(\boldsymbol{x}) = h(\mathbf{R}_\theta \boldsymbol{x}) = \sum_{m=1}^{M} a_m(\theta)\varphi_m(\boldsymbol{x}) \quad \text{for all} \quad \theta \in [-\pi, \pi]$$

Why is this interesting/useful?

- Fast implementation



Exercise: Prove that $h(\boldsymbol{x})$ is steerable $\Leftrightarrow$ $\hat{h}(\boldsymbol{\omega})$ is steerable

Steerable Filters (Freeman & Adelson, 1991)

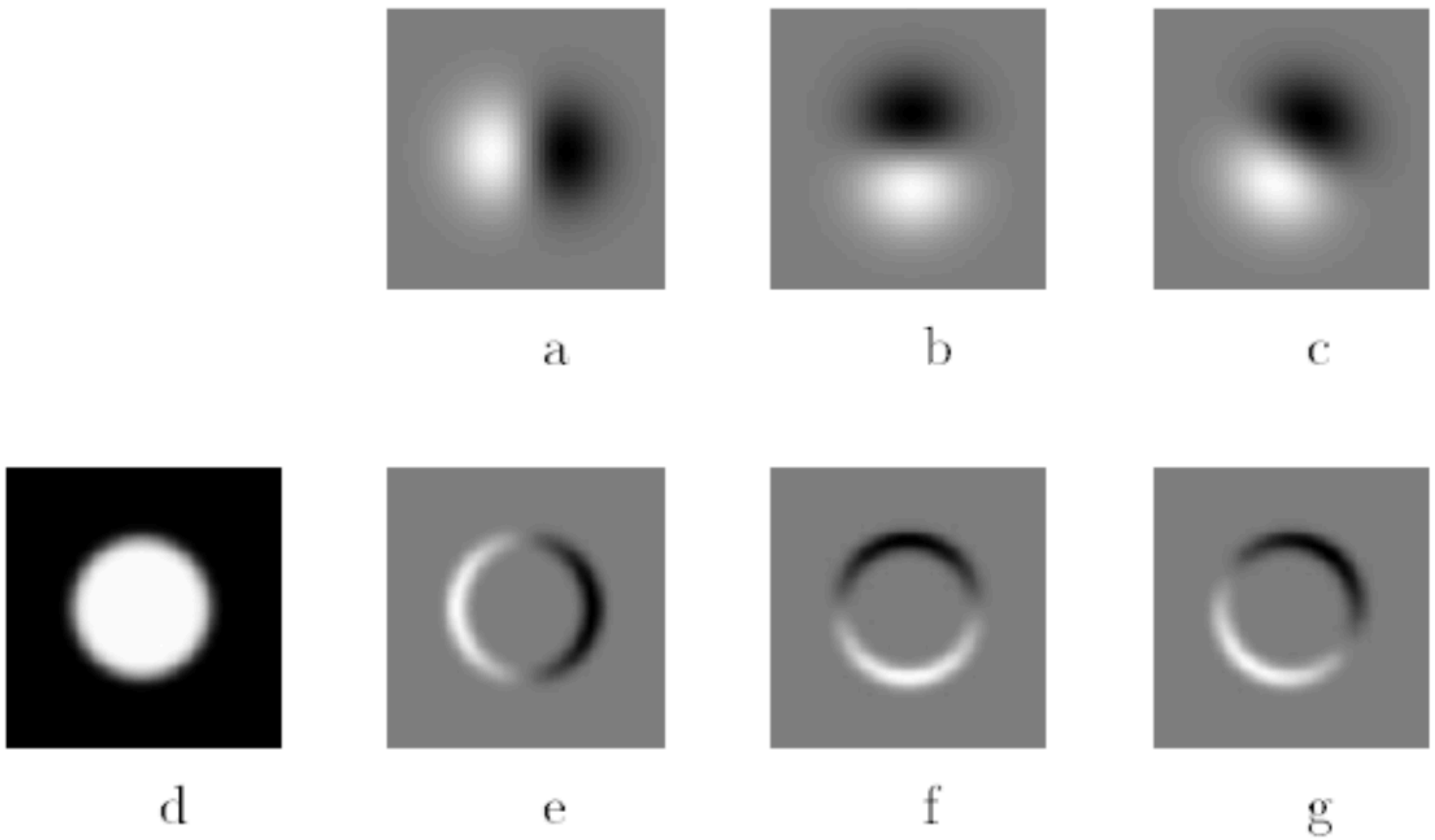


Fig. 1. Example of steerable filters: (a) $G_1^{0^\circ}$ first derivative with respect to x (horizontal) of a Gaussian; (b) $G_1^{90^\circ}$, which is $G_1^{0^\circ}$, rotated by 90°. From a linear combination of these two filters, one can create G_1^θ, which is an arbitrary rotation of the first derivative of a Gaussian; (c) $G_1^{60^\circ}$, formed by $\frac{1}{2}G_1^{0^\circ} + \frac{\sqrt{3}}{2}G_1^{90^\circ}$. The same linear combinations used to synthesize G_1^θ from the basis filters will also synthesize the response of an image to G_1^θ from the responses of the image to the basis filters; (d) image of circular disk; (e) $G_1^{0^\circ}$ (at a smaller scale than pictured above) convolved with the disk (d); (f) $G_1^{90^\circ}$ convolved with (d); (g) $G_1^{60^\circ}$ convolved with (d), obtained from $\frac{1}{2}$ (image (e)) $+ \frac{\sqrt{3}}{2}$ (image (f)).

Steerable Filter Design

Isotropic low-pass function (e.g., Gaussian): $\varphi(x, y)$

Subspace of steerable derivative-based templates:

basis functions

$$h(x, y) = \sum_{m=0}^{M} \sum_{n=0}^{m} \boxed{b_{m,n}} \boxed{\frac{\partial^m \varphi(x, y)}{\partial x^{m-n} \partial y^n}}$$

expansion coefficients

Exercise: Prove that this is steerable.

$\varphi(\boldsymbol{x})$ isotropic $\Leftrightarrow \varphi(\boldsymbol{x}) = \varphi_{\text{iso}}(\|\boldsymbol{x}\|_2) \Leftrightarrow \hat{\varphi}(\boldsymbol{\omega}) = \rho(\|\boldsymbol{\omega}\|_2)$

$h(\boldsymbol{x})$ steerable $\Leftrightarrow \hat{h}(\boldsymbol{\omega})$ steerable

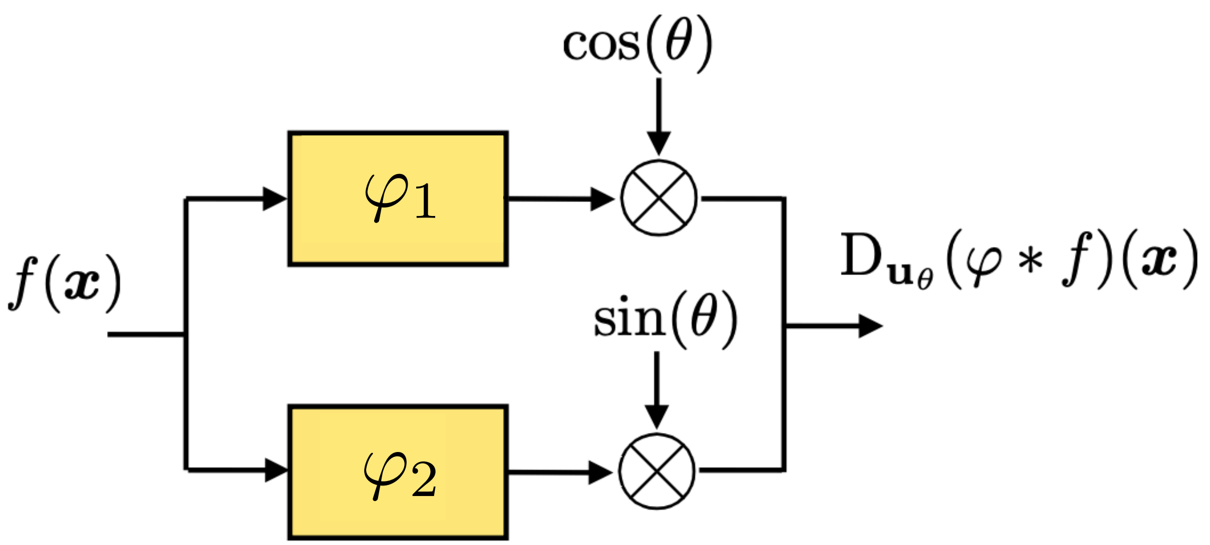
Steerable Filters for Edge and Ridge Detection

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

- Gradient-based edge detector

$$h(\boldsymbol{x}) = \varphi_1(\boldsymbol{x}) = \frac{\partial \varphi(\boldsymbol{x})}{\partial x}$$

$$\varphi_2(\boldsymbol{x}) = \frac{\partial \varphi(\boldsymbol{x})}{\partial y}$$

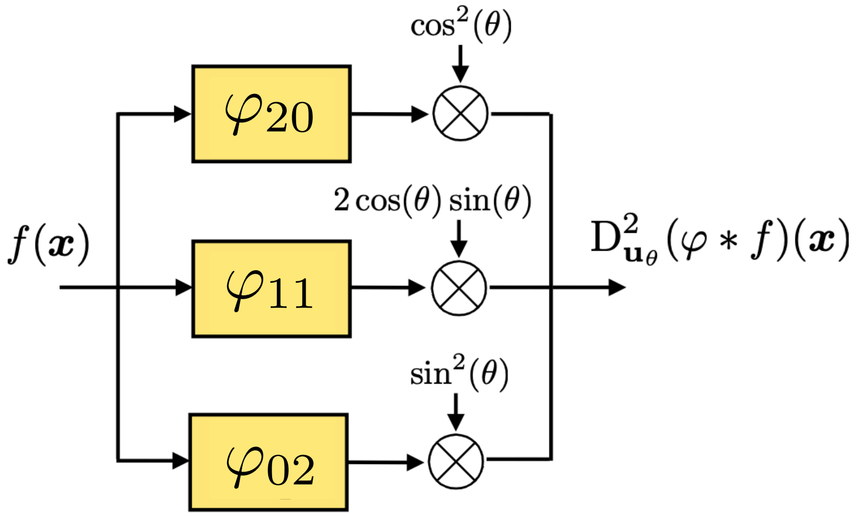


$$h(\mathbf{R}_\theta \boldsymbol{x}) = \mathrm{D}_{\boldsymbol{u}_\theta} \varphi(\boldsymbol{x}) = \boldsymbol{u}_\theta^\mathsf{T} \nabla \varphi(\boldsymbol{x}) = \cos\theta \varphi_1(\boldsymbol{x}) + \sin\theta \varphi_2(\boldsymbol{x})$$

- Second-order derivatives = ridge detector

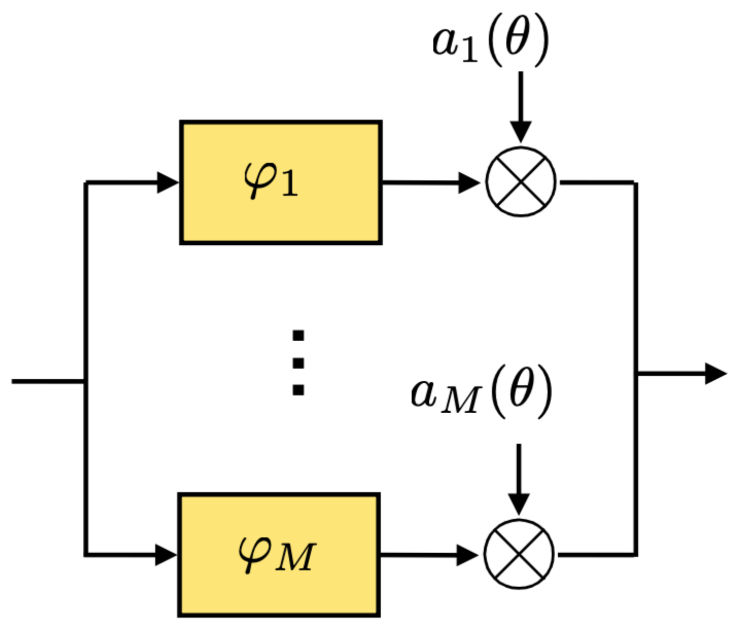
$$h(\boldsymbol{x}) = \varphi_{20}(\boldsymbol{x}) = \frac{\partial^2 \varphi(\boldsymbol{x})}{\partial x^2}$$

$\varphi_{02}, \varphi_{11}$

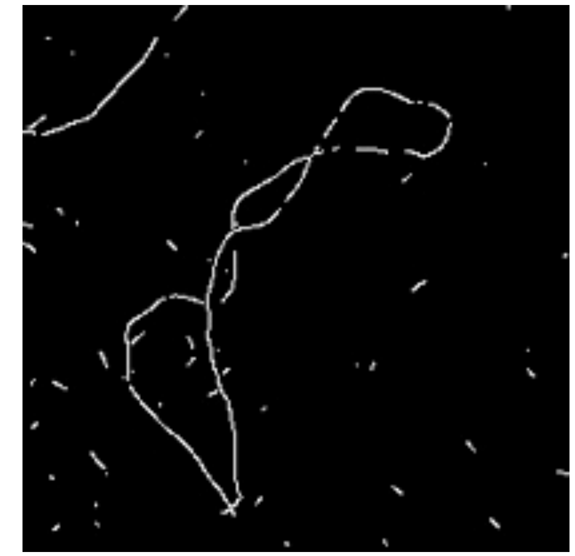


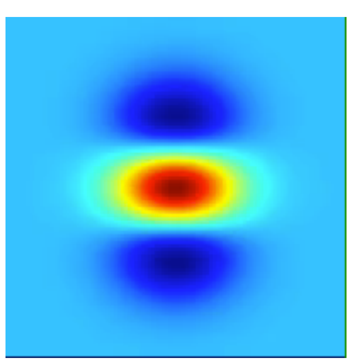
$$h(\mathbf{R}_\theta \boldsymbol{x}) = \mathrm{D}^2_{\boldsymbol{u}_\theta} \varphi(\boldsymbol{x}) = (\cos\theta)^2 \varphi_{20}(\boldsymbol{x}) + 2\cos\theta\sin\theta \varphi_{11}(\boldsymbol{x}) + (\sin\theta)^2 \varphi_{02}(\boldsymbol{x})$$

Ridge Detection Example

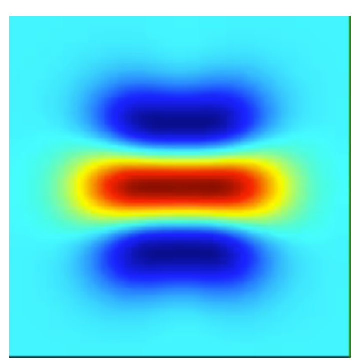


$$\theta^*(\boldsymbol{x}) = \arg\max_{\theta} \{(h_\theta * f)(\boldsymbol{x})\}$$

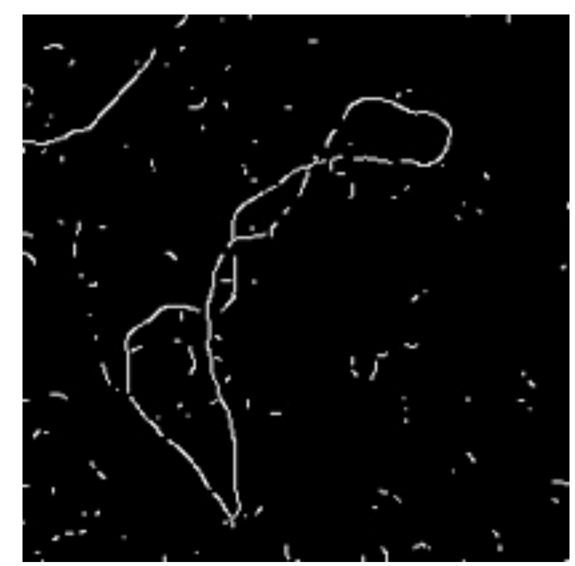




2nd order



4th order

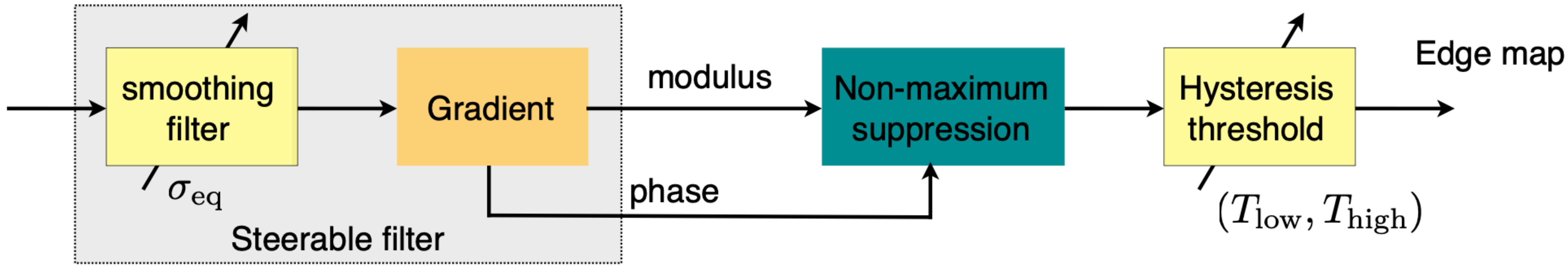


without steering

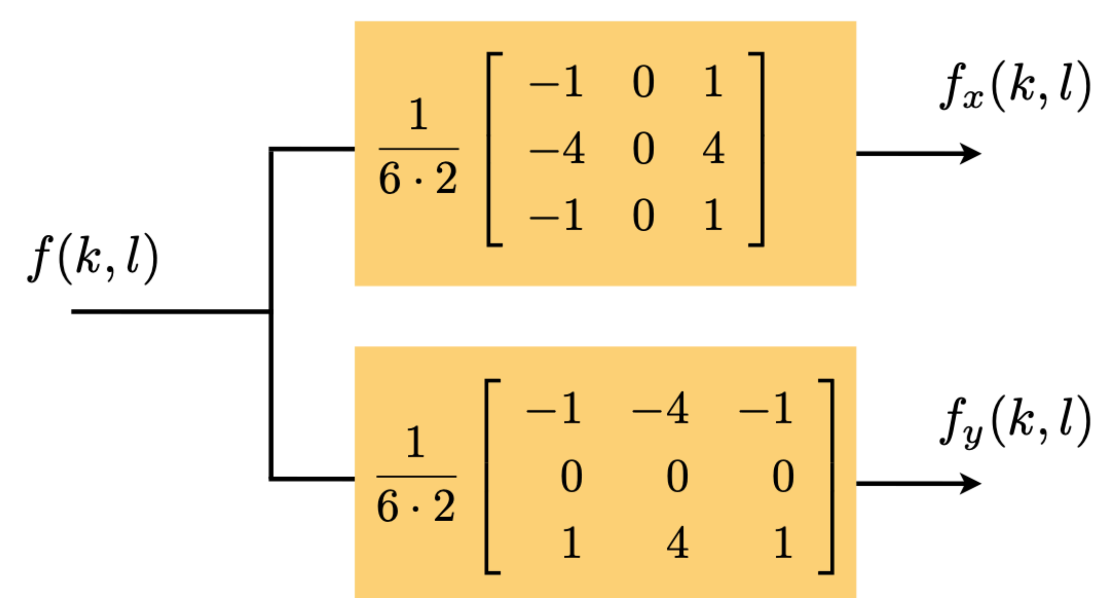
Canny Edge Detector Revisited

- State-of-the-art edge detector

Edge point = local maximum of first directional derivative



- Smoothing
  - Gaussian filter: isotropic + separable (the only one)
  - Implementation: cascade of simple recursive filters
- Discrete gradient filters



Canny Edge Detector Revisited

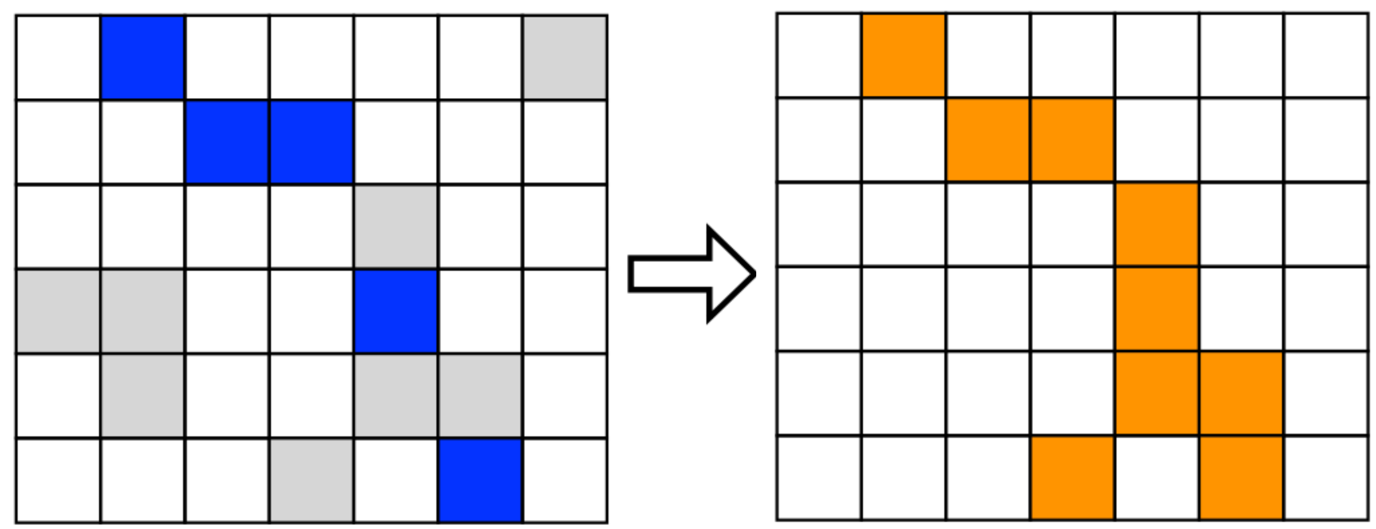
- Non-maximum suppression at $\boldsymbol{x}_0$

$$\boldsymbol{u} = \frac{\nabla f(\boldsymbol{x}_0)}{\|\nabla f(\boldsymbol{x}_0)\|_2} \;:\; \text{unit vector in gradient direction}$$

$$\boxed{\begin{array}{ll} \text{if } \|\nabla f(\boldsymbol{x}_0)\|_2 \geq \|\nabla f(\boldsymbol{x}_0 \pm \boldsymbol{u})\|_2 & \text{then } g(\boldsymbol{x}_0) = \|\nabla f(\boldsymbol{x}_0)\|_2 \\ & \text{else } g(\boldsymbol{x}_0) = 0 \end{array}}$$

- Hysteresis threshold

Set of points: $\boldsymbol{k} \in \mathbb{Z}^2$



Two auxiliary edge maps:

$E_{\text{low}} = \{\boldsymbol{k} \;:\; T_{\text{low}} \leq g[\boldsymbol{k}] \leq T_{\text{high}}\}$

$E_{\text{high}} = \{\boldsymbol{k} \;:\; T_{\text{high}} \leq g[\boldsymbol{k}]\}$

Final edge map:

$E = \{\boldsymbol{k} \in E_{\text{low}} \cup E_{\text{high}} \;:\;$ there exists a path that connects $\boldsymbol{k}$ to $E_{\text{high}}\}$