上海科技大学
ShanghaiTech University

# Edge

CS172 Computer Vision I

Instructor: Jiayuan Gu

# Agenda

- Edge
  - Image gradient
  - Derivatives with convolution
  - Gaussian derivative filter

- Texture

# Edge Detection

Winter in Kraków photographed by Marcin Ryczek

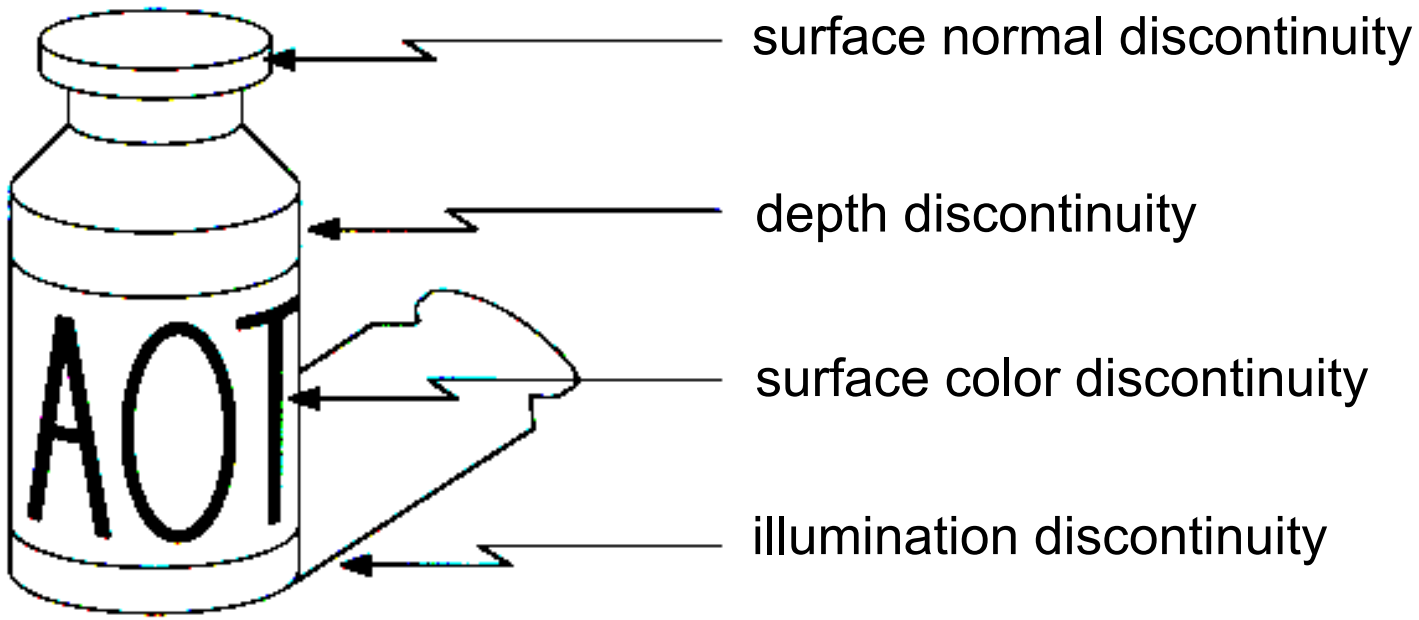
# Edge Detection

- **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

# Origin of Edges

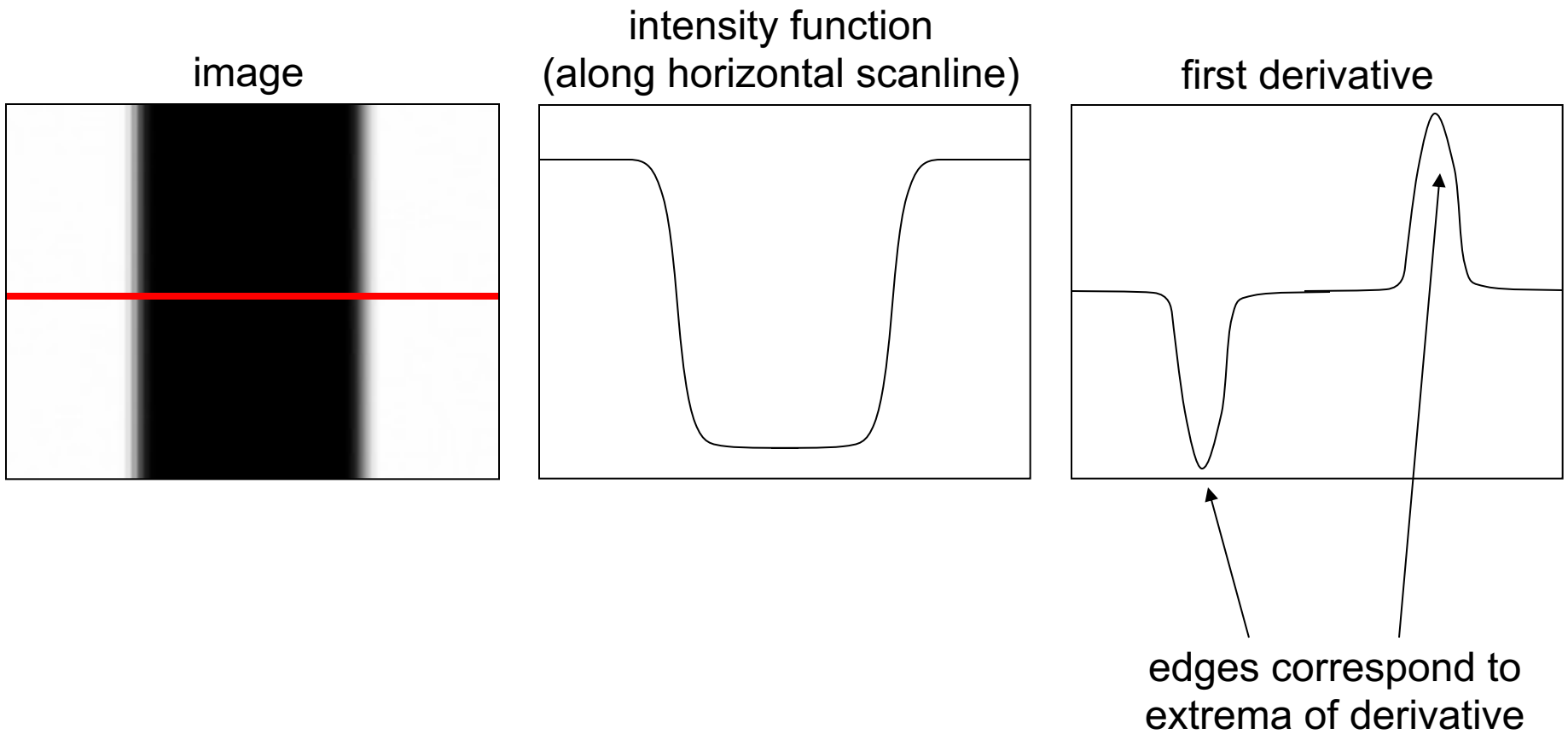
Edges are caused by a variety of factors:




Source: Steve Seitz

# Characterizing Edge

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



# Derivatives with Convolution

For 2D function $f(x, y)$, the partial derivative is:

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

For discrete data, we can approximate using finite differences:

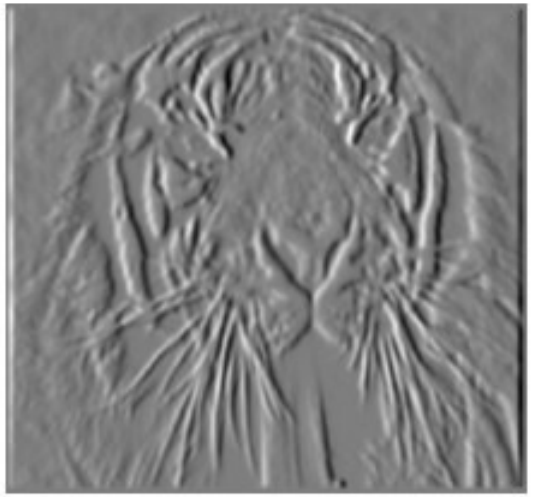
$$\frac{\partial f(x,y)}{\partial x} \approx \frac{f(x+1, y) - f(x,y)}{1}$$

To implement the above as convolution, what would be the associated filter?

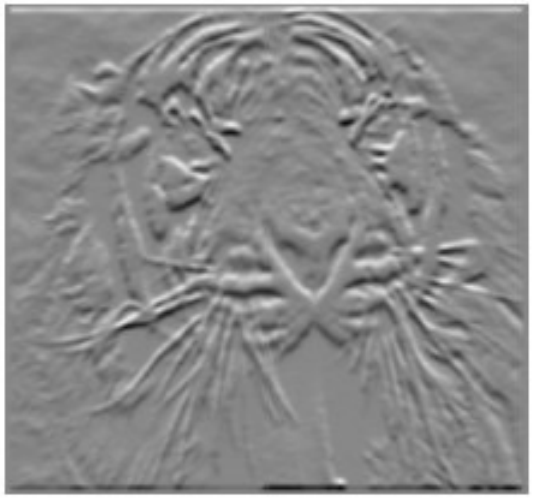
Source: K. Grauman

# Partial Derivatives of an Image

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



| -1 | 1 |
|---|---|



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

| -1 |
|---|
| 1 |

or

| 1 |
|---|
| -1 |

Which shows changes with respect to x?

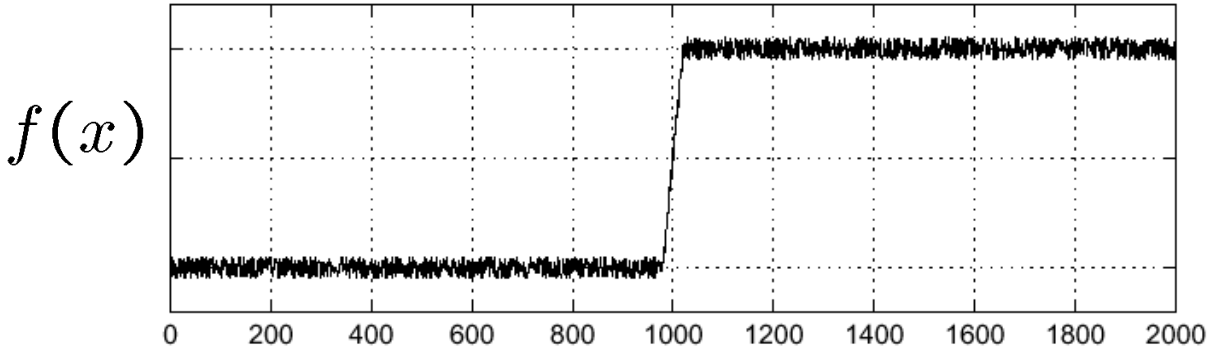
The kernel is already flipped for simplicity

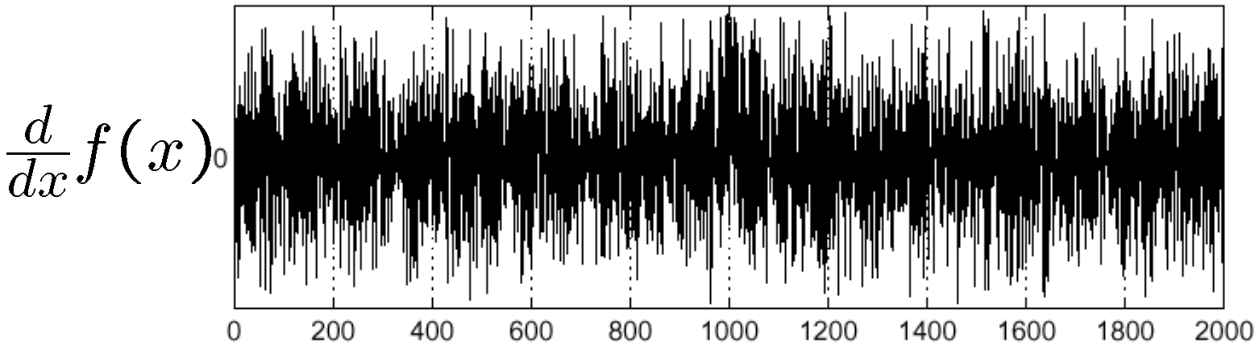
# Gradient Magnitude

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

# Effects of Noise

- Consider a single row or column of the image

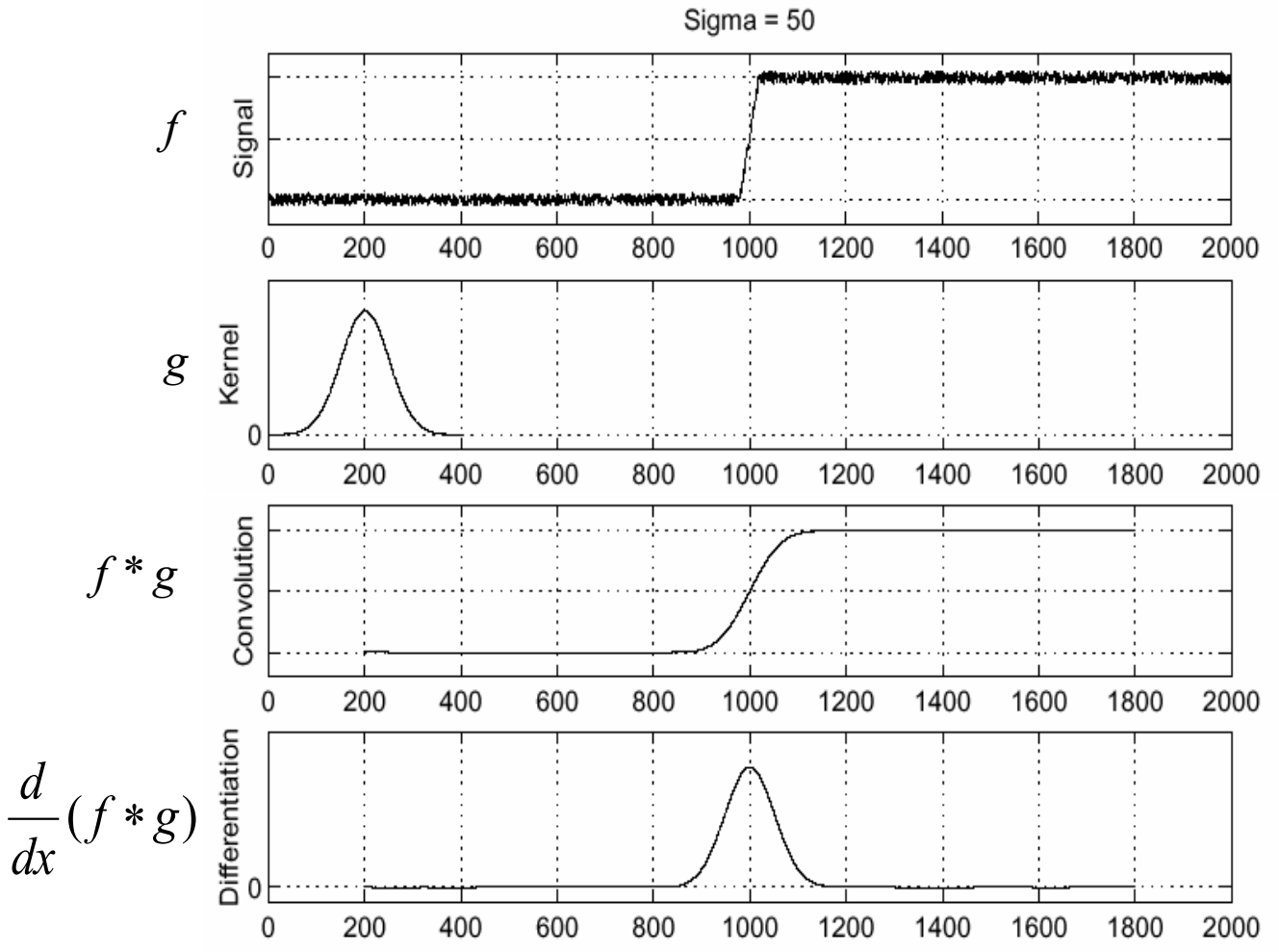



Where is the edge?

Source: S. Seitz

# Solution: Smooth First




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

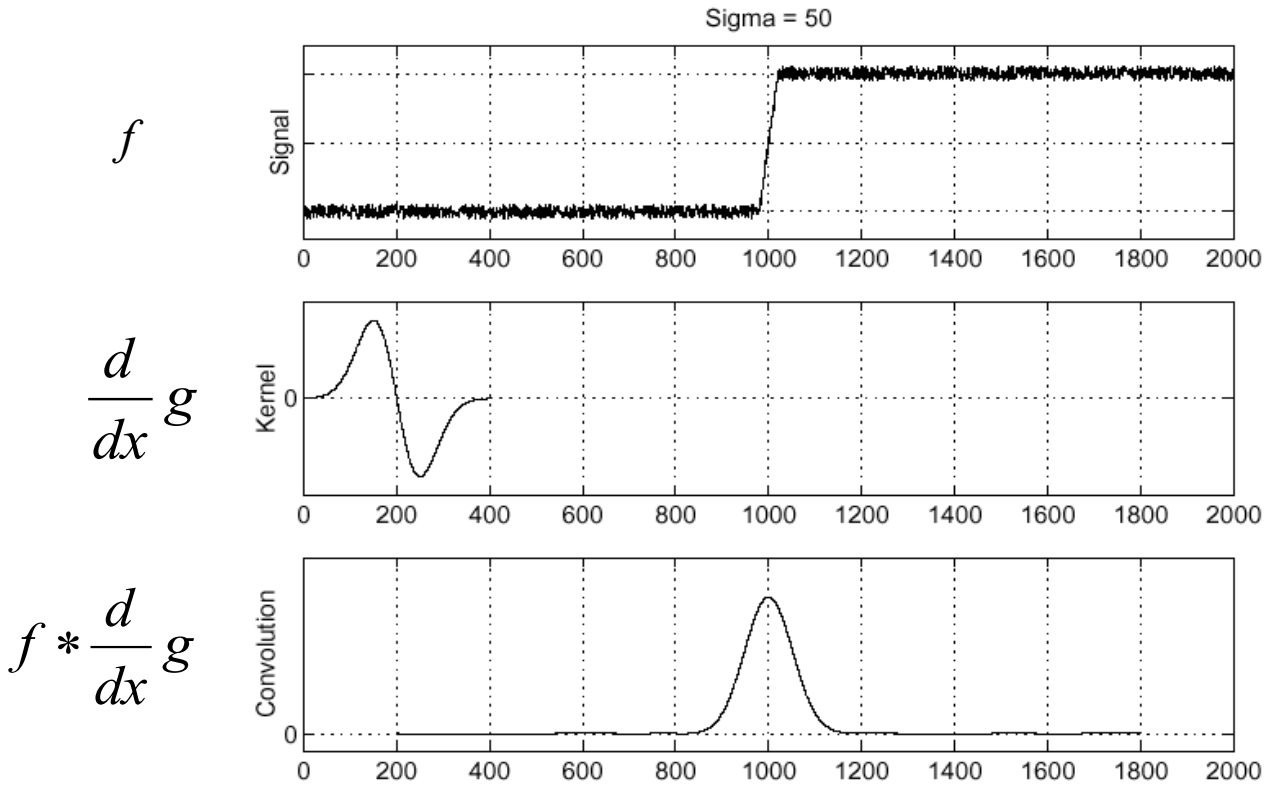
Source: S. Seitz

# Derivative Theorem of Convolution

- Differentiation is convolution, and convolution is associative:

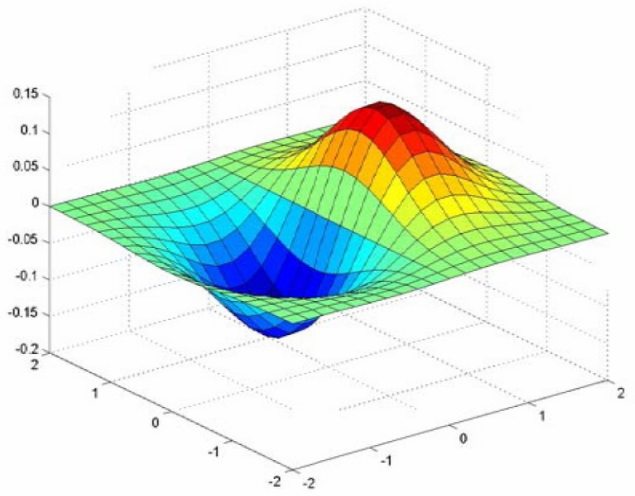
$$\frac{d}{dx}(f * g) = f * \frac{d}{dx} g$$

- This saves us one operation:

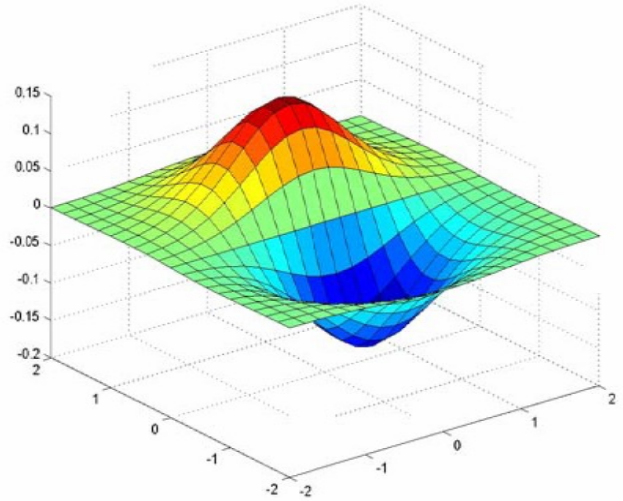



Source: S. Seitz

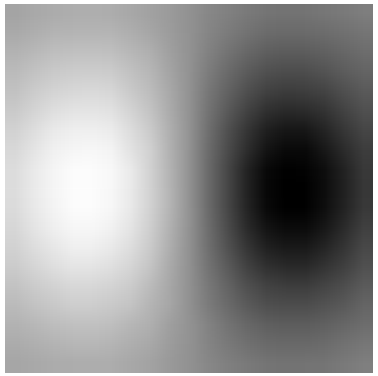
# Derivative of Gaussian Filters

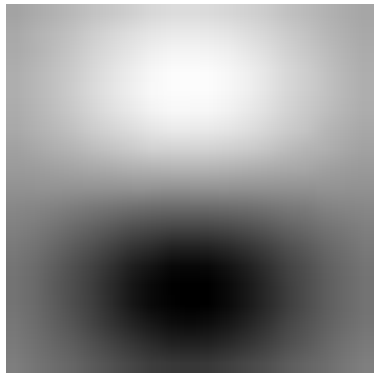



*x*-direction




*y*-direction

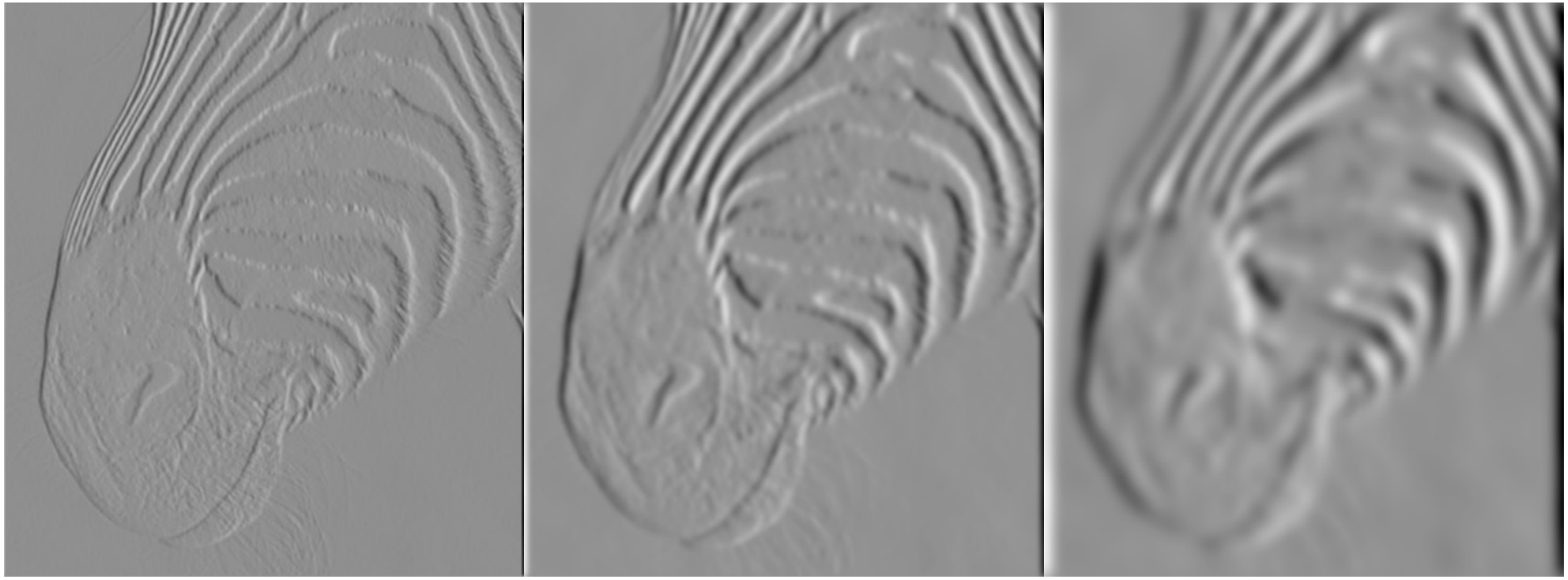




Which one finds horizontal/vertical edges?

Are these filters separable?

# Scale of Gaussian Derivative Filter



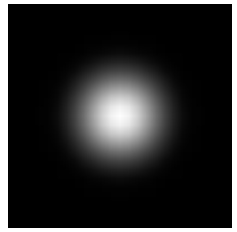
1 pixel 3 pixels 7 pixels

Smoothed derivative removes noise, but blurs edge.
Also finds edges at different “scales”

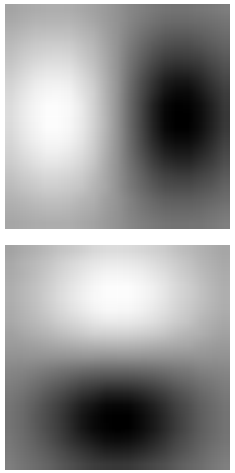
Source: D. Forsyth

# Review: Smoothing vs. Derivative Filters

- Smoothing filters
  - Gaussian: remove “high-frequency” components; “low-pass” filter
  - Can the values of a smoothing filter be negative?
  - What should the values sum to?
    - **One:** constant regions are not affected by the filter



- Derivative filters
  - Derivatives of Gaussian
  - Can the values of a derivative filter be negative?
  - What should the values sum to?
    - **Zero:** no response in constant regions



# Sobel Operator

$$g_x = [-1 \quad 0 \quad 1] * \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

$g_x =$

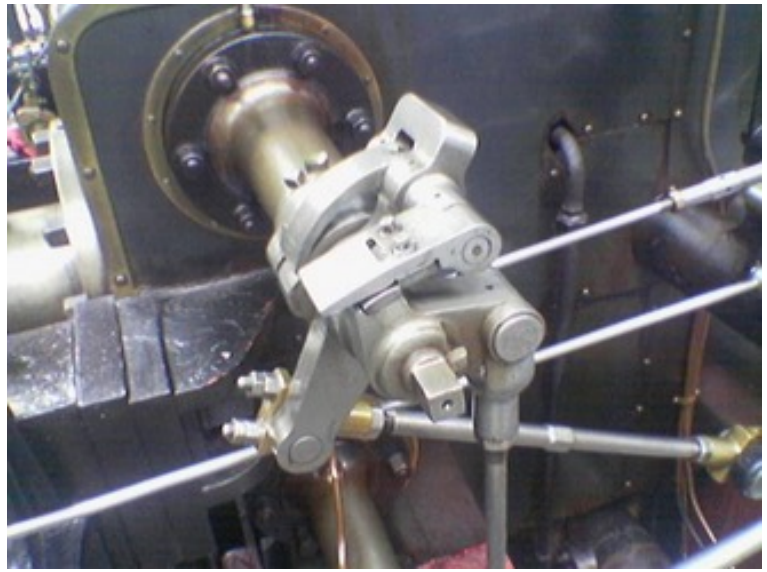
| | | |
|---|---|---|
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

$g_y =$

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

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

The kernel is already flipped for simplicity



An example image from wiki

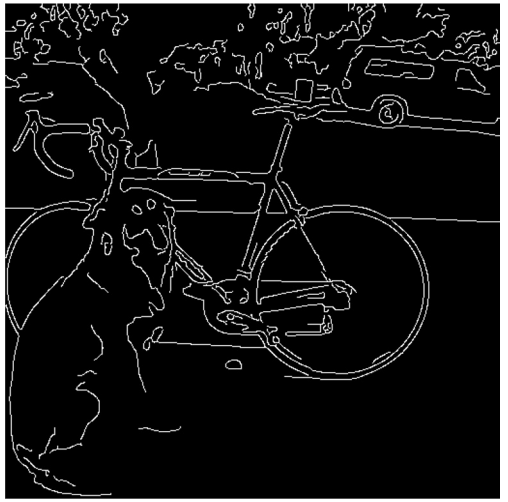
The Sobel operator applied to that image

Self-reading

# Canny Edge Detection

Algorithm:

- 1. Smooth image (only want “real” edges, not noise)
- 2. Calculate gradient direction and magnitude
- 3. Non-maximum suppression perpendicular to edge
- 4. Threshold into strong, weak, no edge
- 5. Connect together components



Self-reading

Credit to: UW CSE455 Linda Shapiro

# Canny Edge Detection: Smooth

Gaussian filter

Credit to: UW CSE455 Linda Shapiro

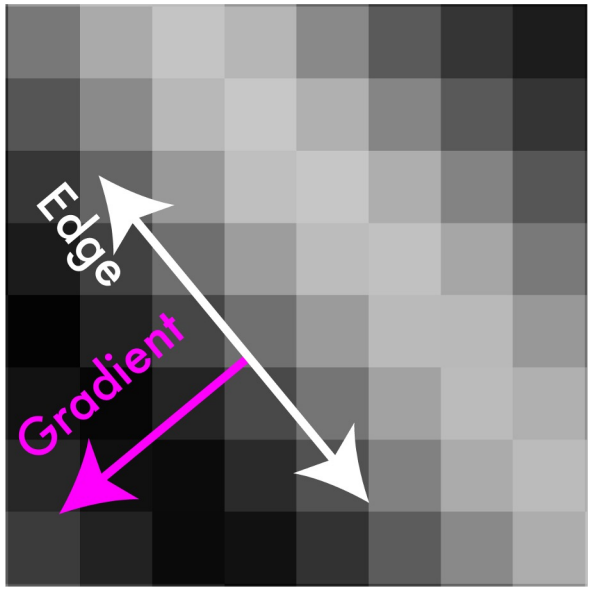
# Canny Edge Detection: Gradient

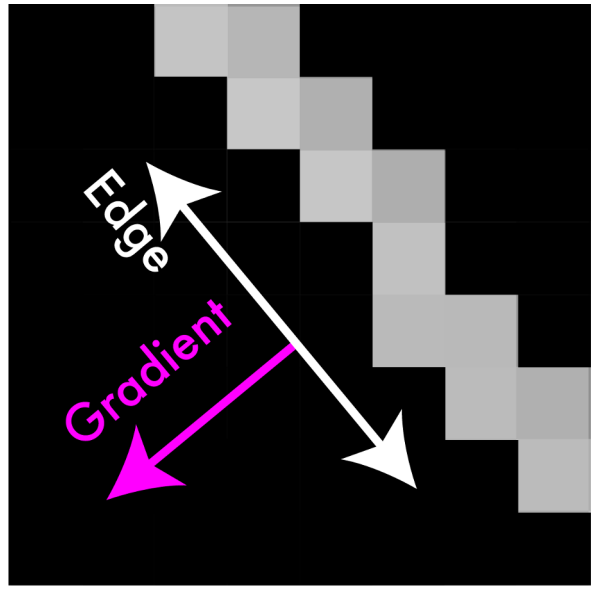
Sobel operator

Credit to: UW CSE455 Linda Shapiro

# Canny Edge Detection: Non-Maximum Suppression

-Want single pixel edges, not thick blurry lines
-Need to check nearby pixels
-See if response is highest



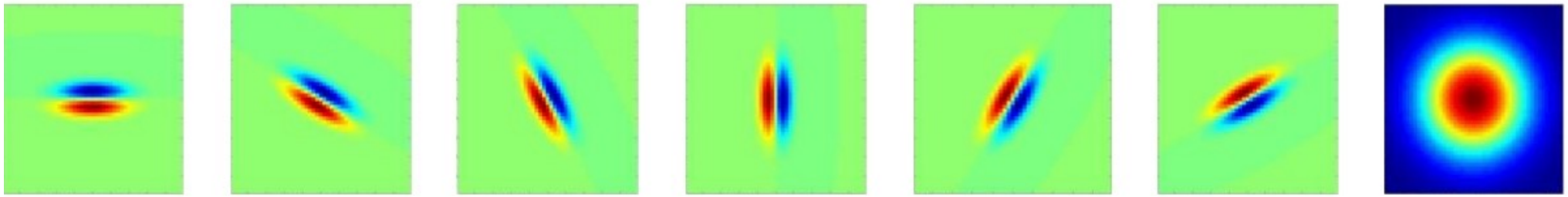




Credit to: UW CSE455 Linda Shapiro

# Texture

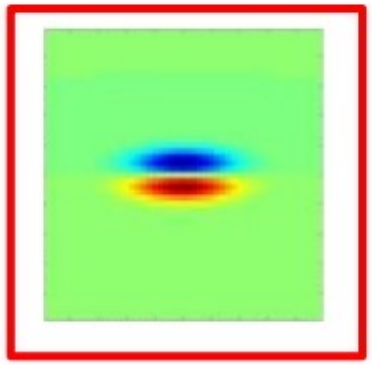
An **image texture** is the small-scale structure perceived on an image, based on the **spatial arrangement of color or intensities**.

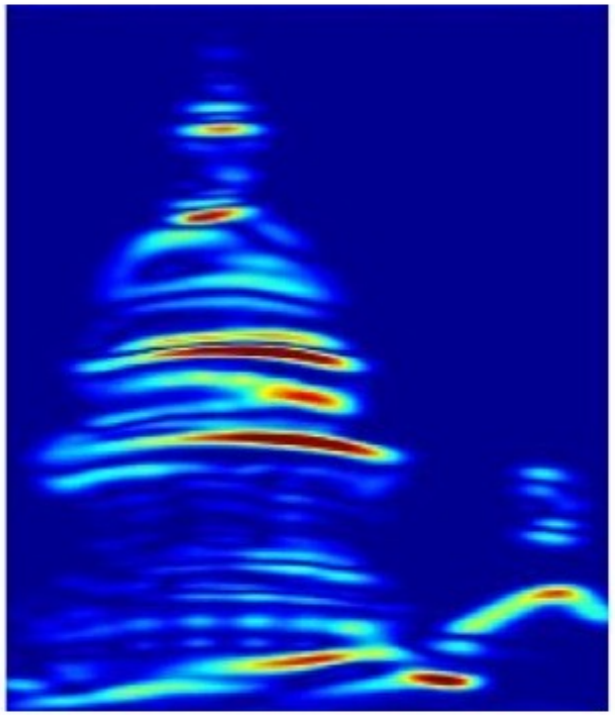
# Kernels to Detect Texture



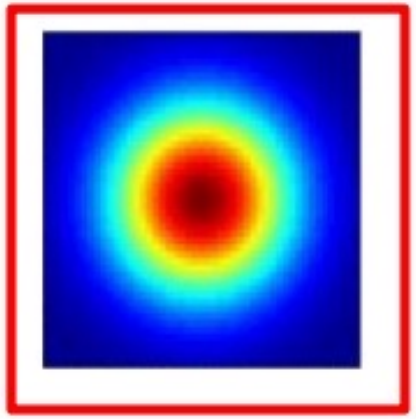
Describe what kind of pattern should be detected

# Response of One Kernel

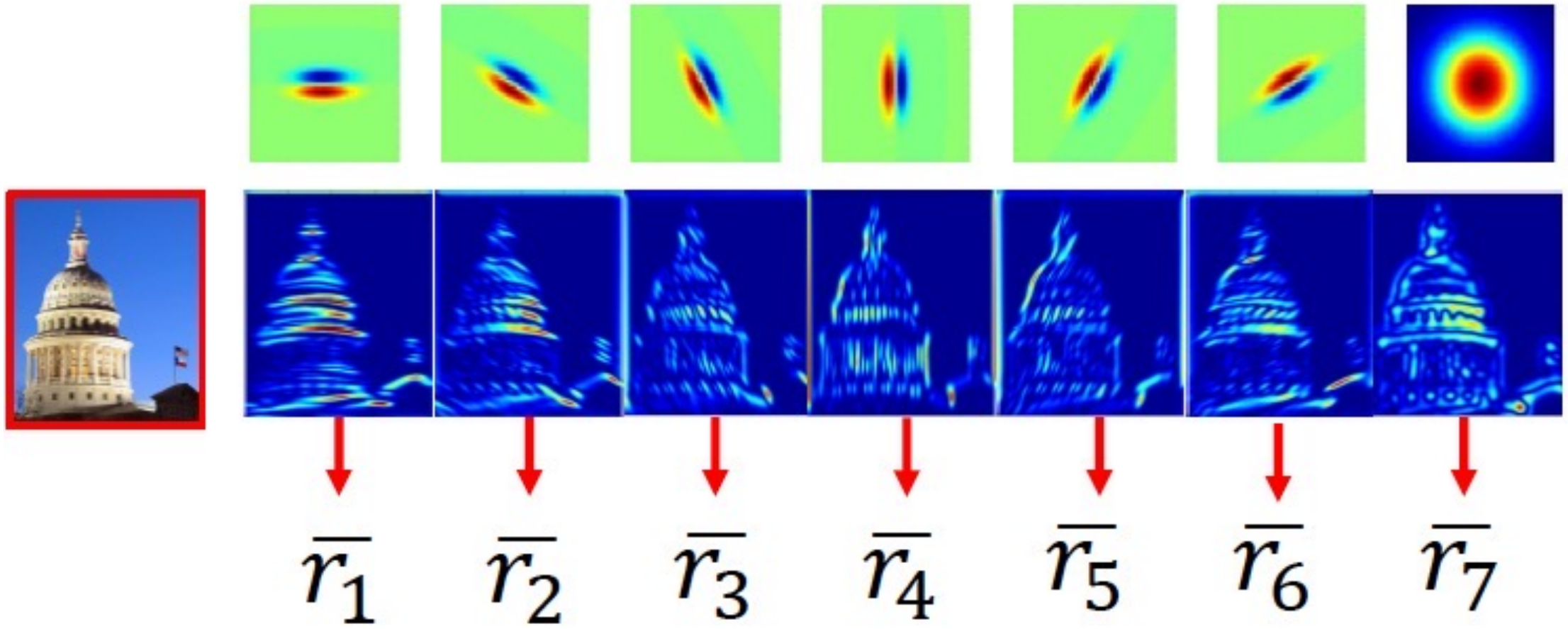




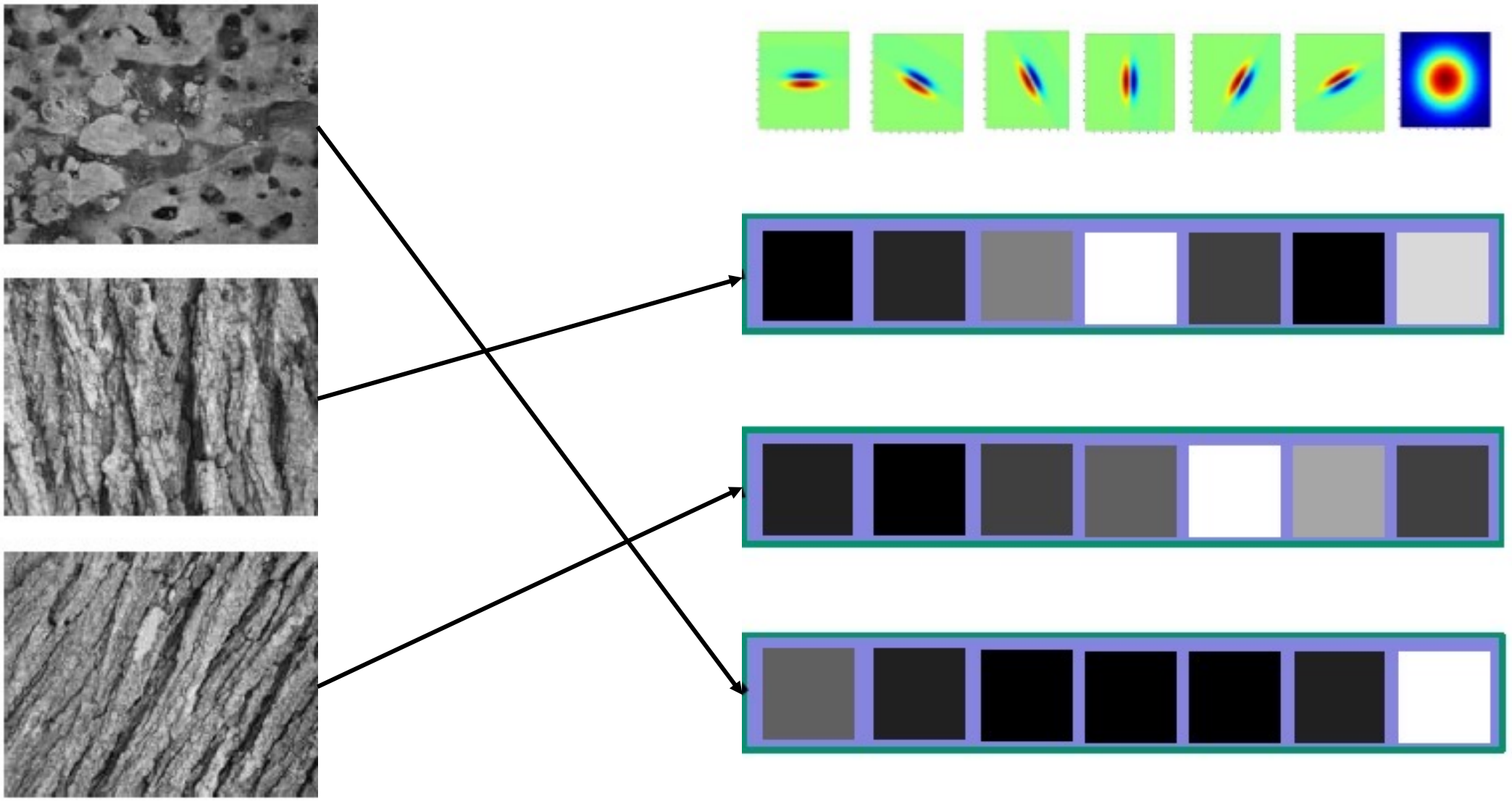
# Response of One Kernel



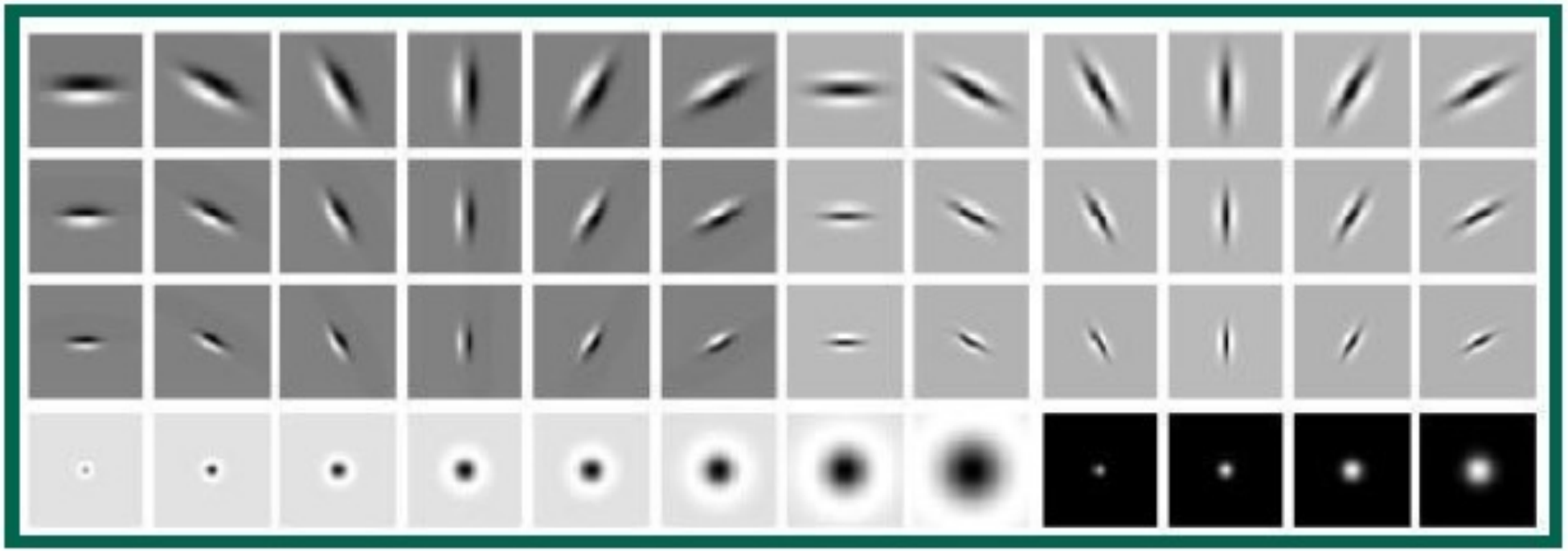
# Feature: Response Pattern



# Feature can be Used for Recognition



# More Kernels




How to design kernels?