Edge detection

Winter in Kraków photographed by Marcin Ryczek

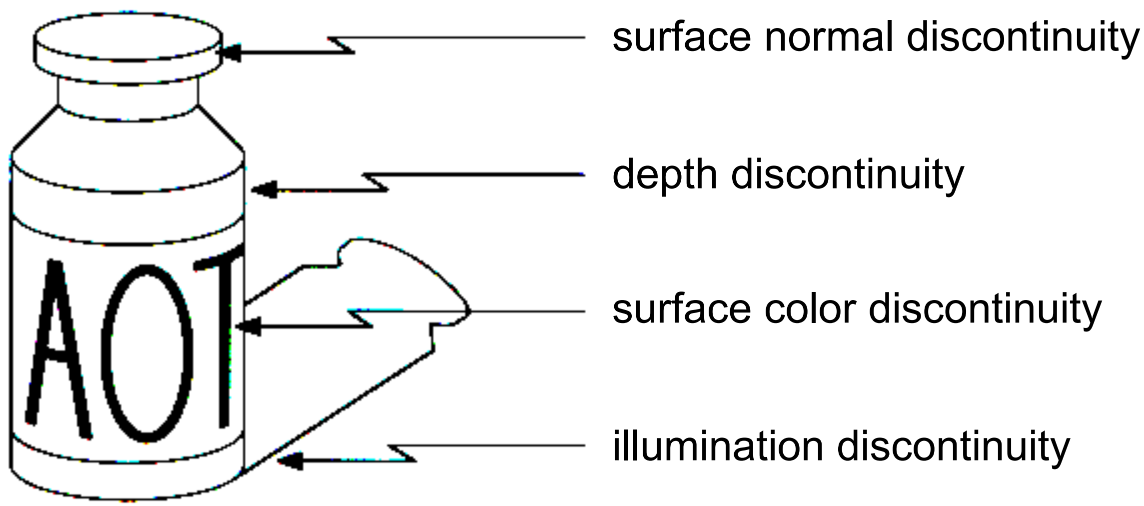
Edge detection

- **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

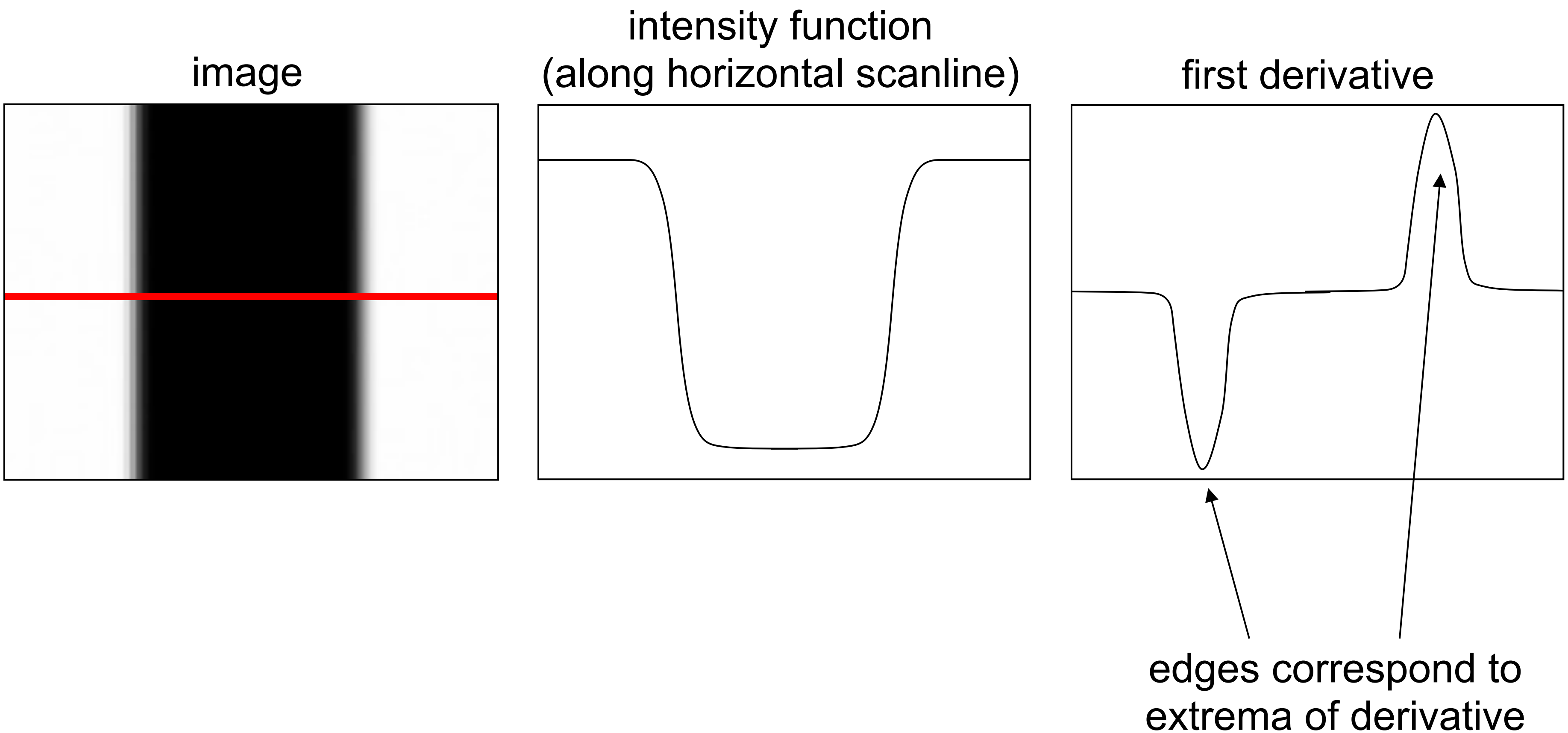
Edges are caused by a variety of factors:



Source: Steve Seitz

Edge detection

- 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:

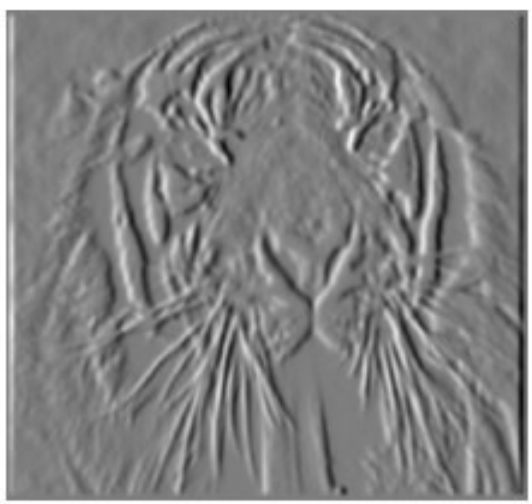
$$\frac{\partial f(x,y)}{\partial x} \approx \frac{f(x+1, y) - f(x,y)}{1}$$

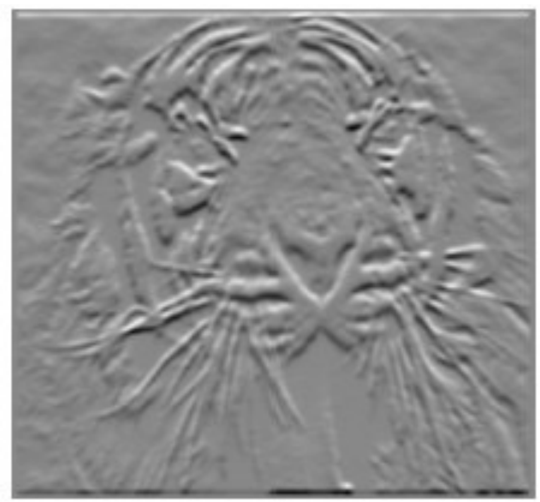
To implement the above as convolution, what would be the associated filter?

Source: K. Grauman

Partial derivatives of an image

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





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

-1	**1**

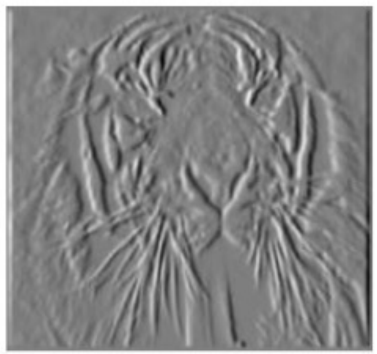
-1
1

or

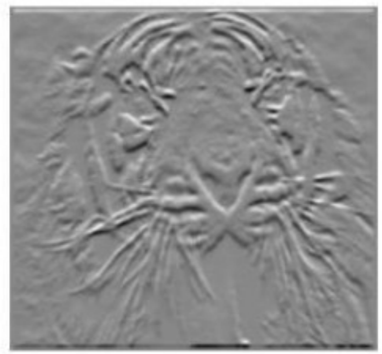
1
-1

Which shows changes with respect to x?

Gradient Magnitude



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

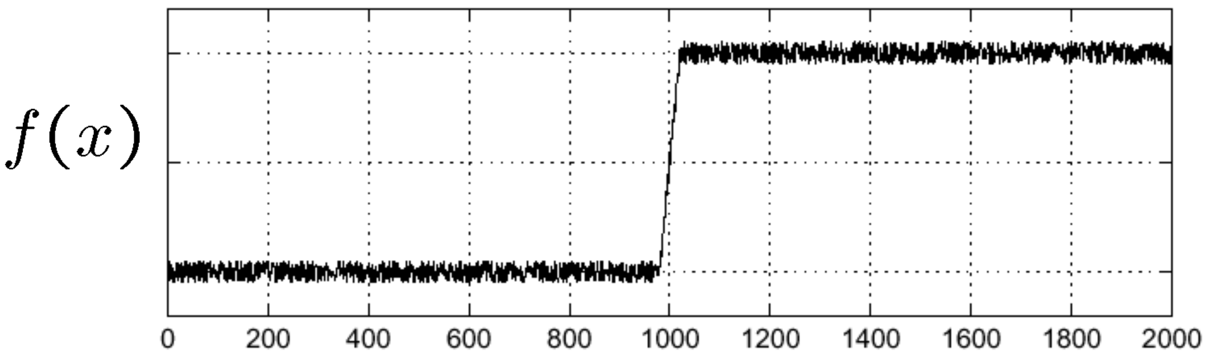


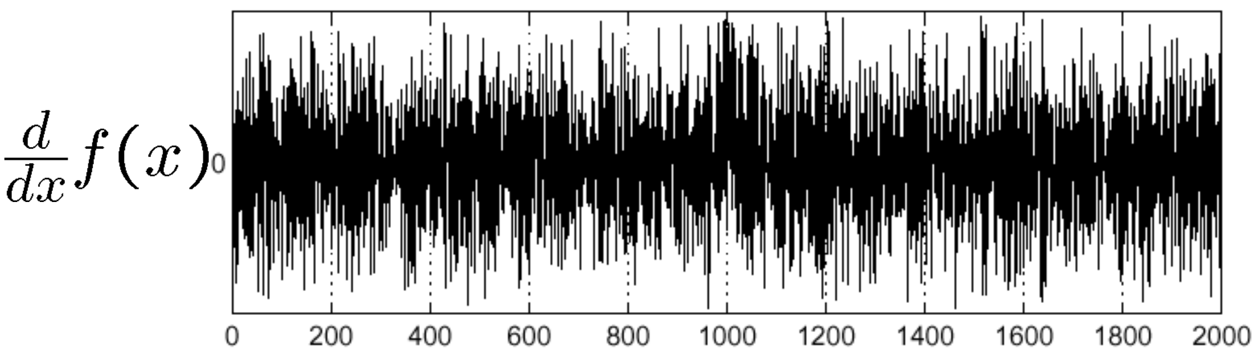
$$\frac{\partial f(x,y)}{\partial \mathrm{y}}$$

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

Effects of noise

Consider a single row or column of the image

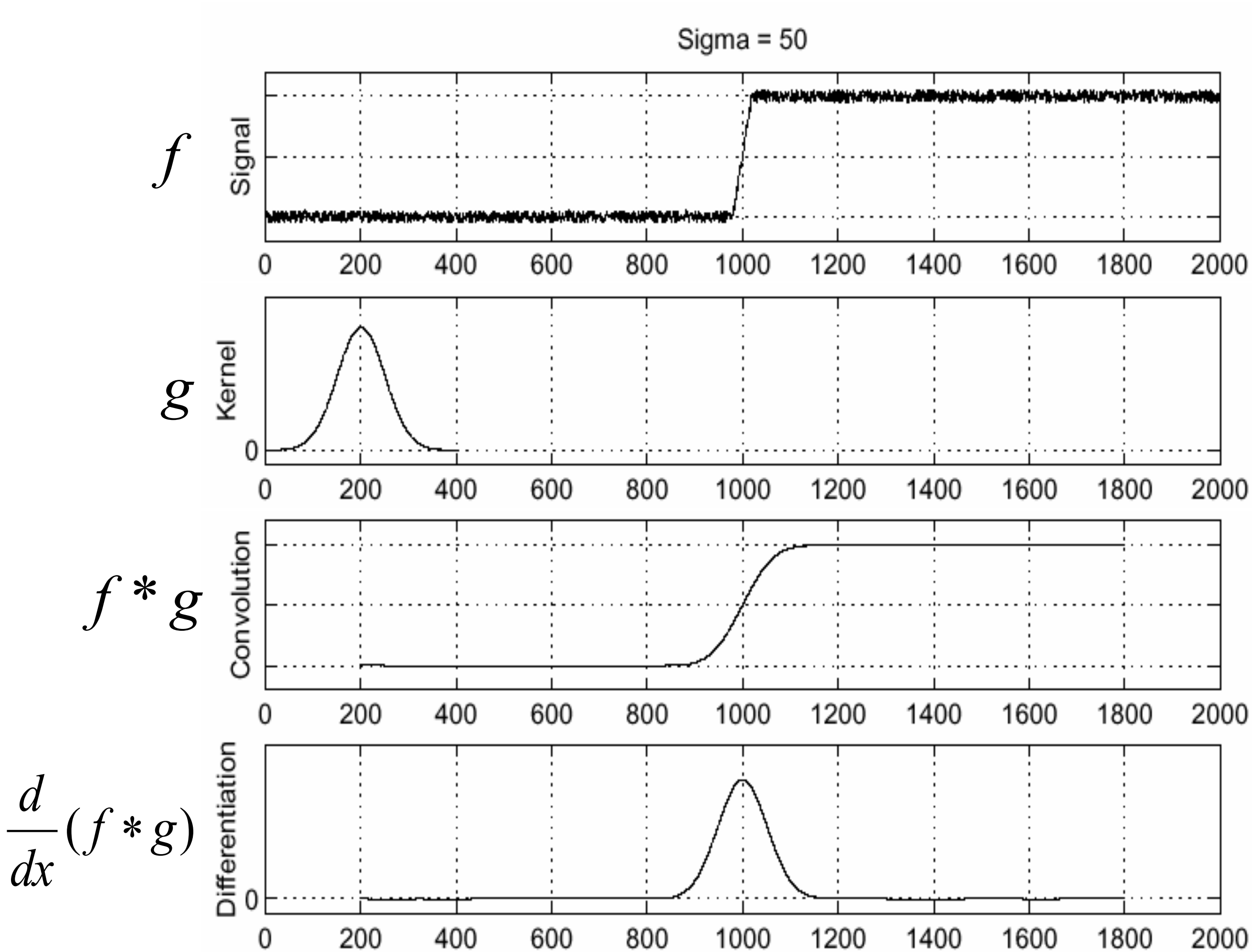




Where is the edge?

Source: S. Seitz

Solution: smooth first



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

Source: S. Seitz

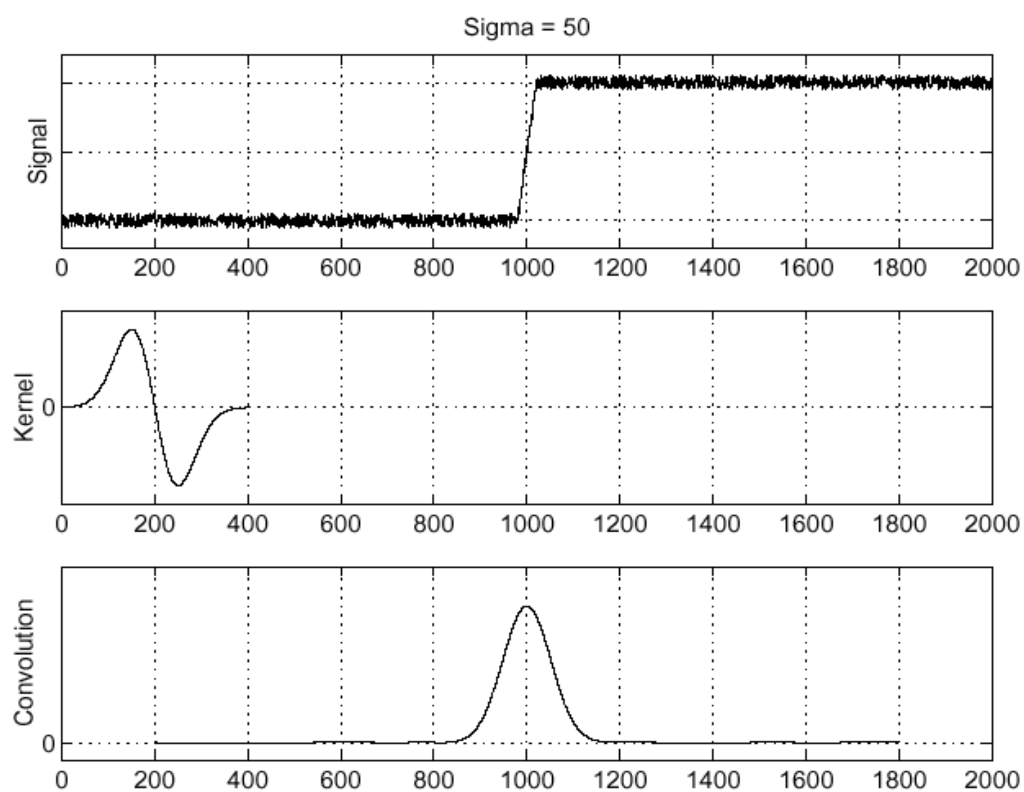
Derivative theorem of convolution

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

f

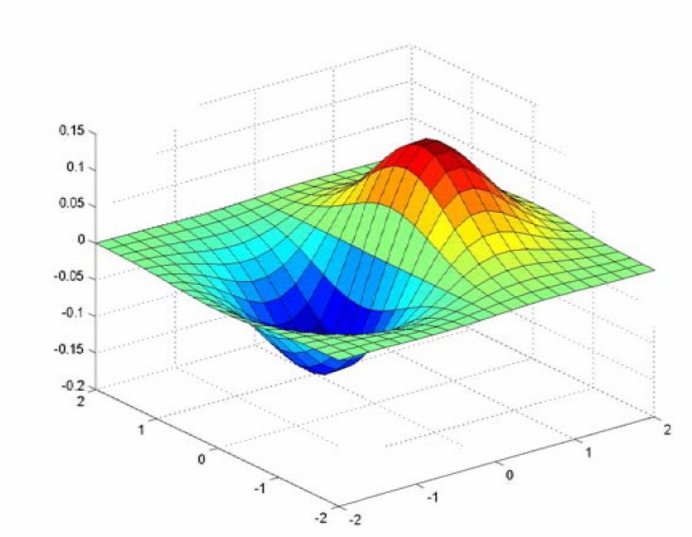
$\frac{d}{dx} g$

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

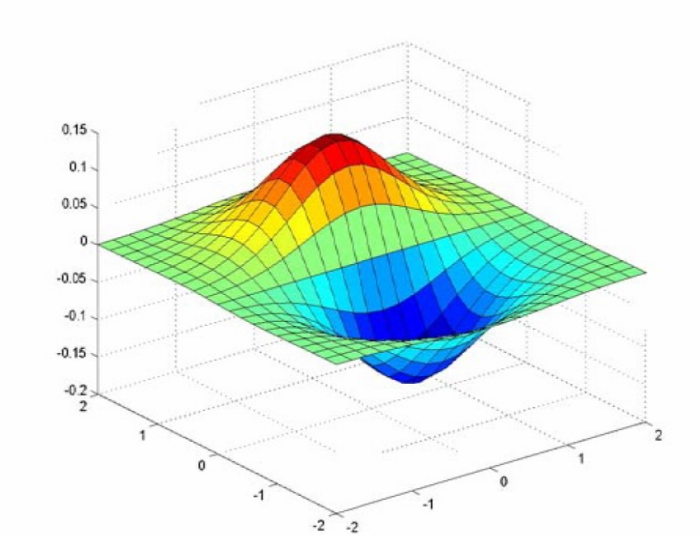


Source: S. Seitz

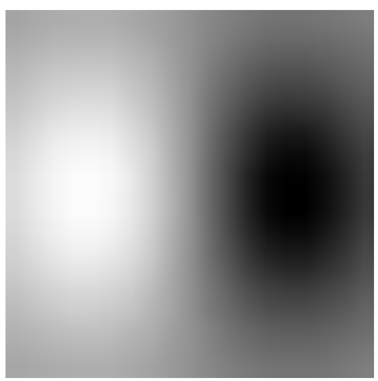
Derivative of Gaussian filters

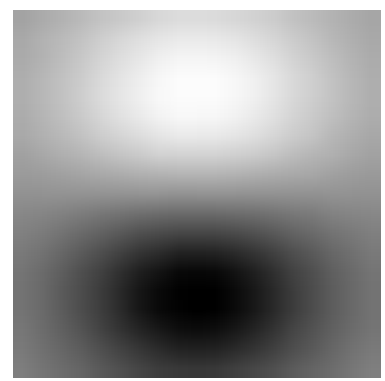


x-direction



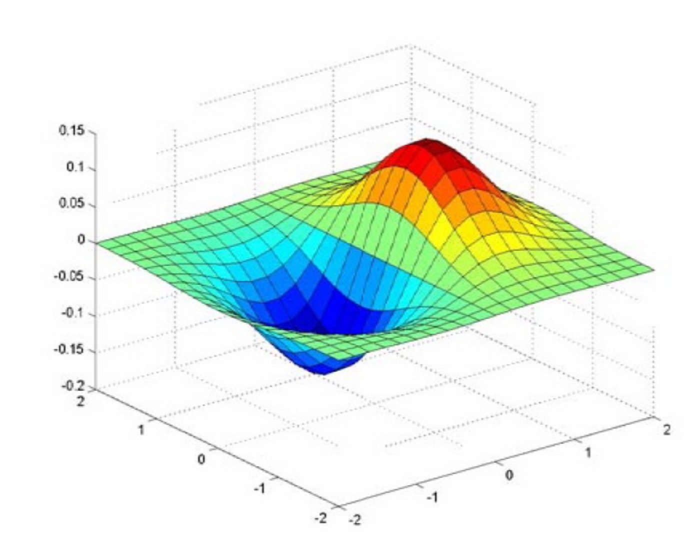
y-direction



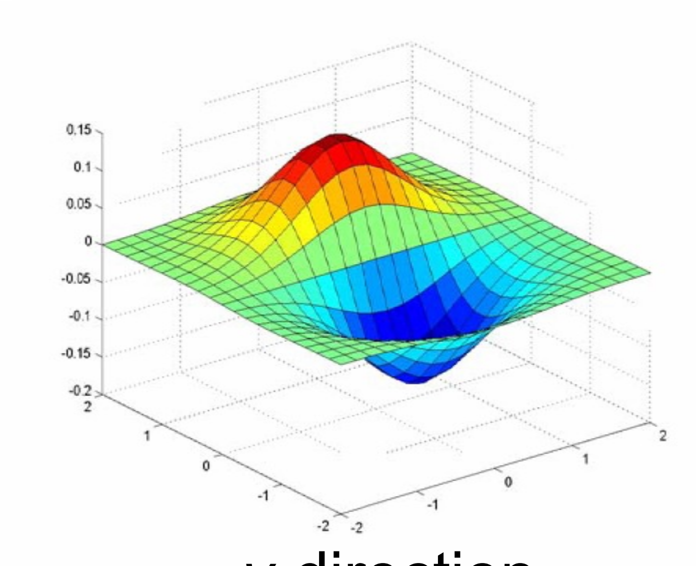


Which one finds horizontal/vertical edges?

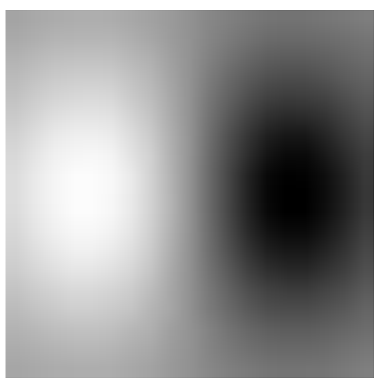
Derivative of Gaussian filters

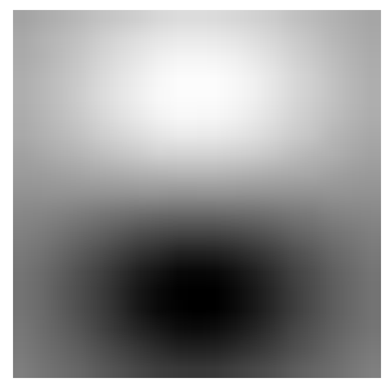


x-direction



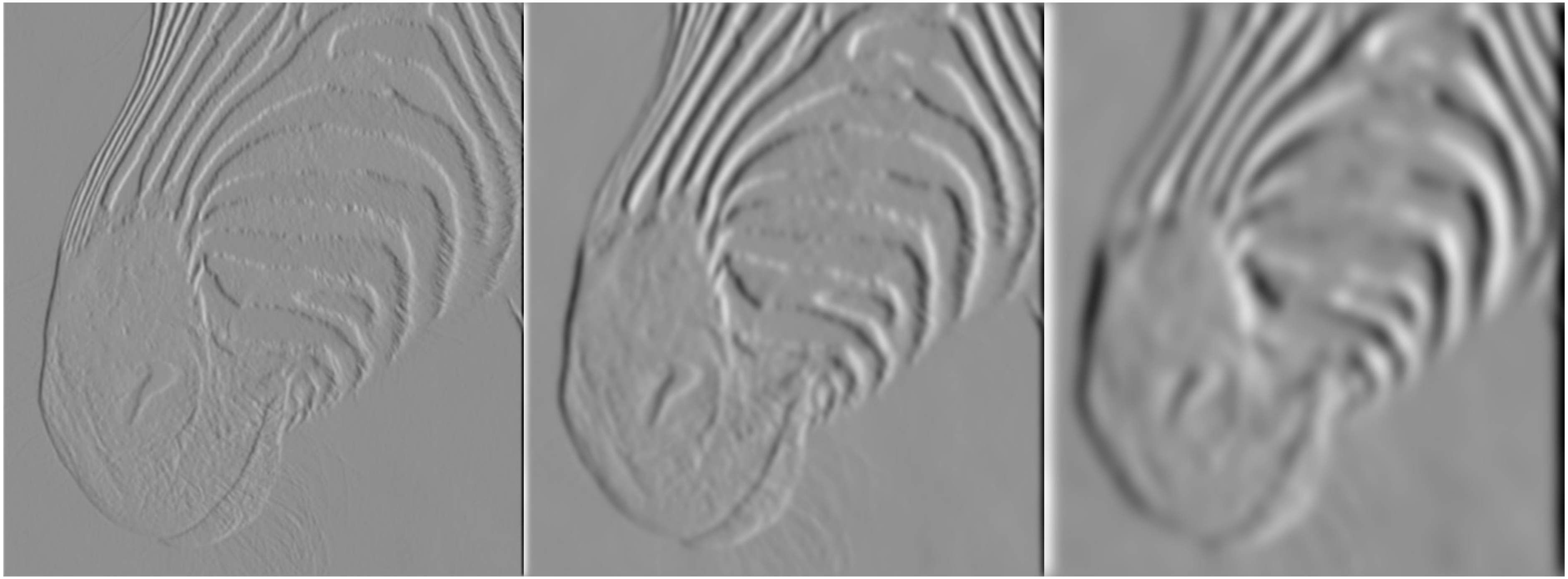
y-direction





Are these filters separable?

Scale of Gaussian derivative filter



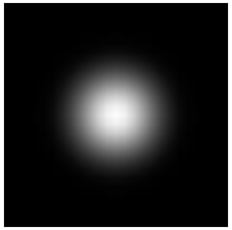
1 pixel 3 pixels 7 pixels

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

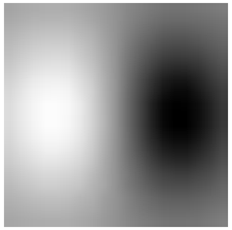
Source: D. Forsyth

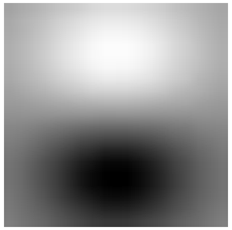
Review: Smoothing vs. derivative filters

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

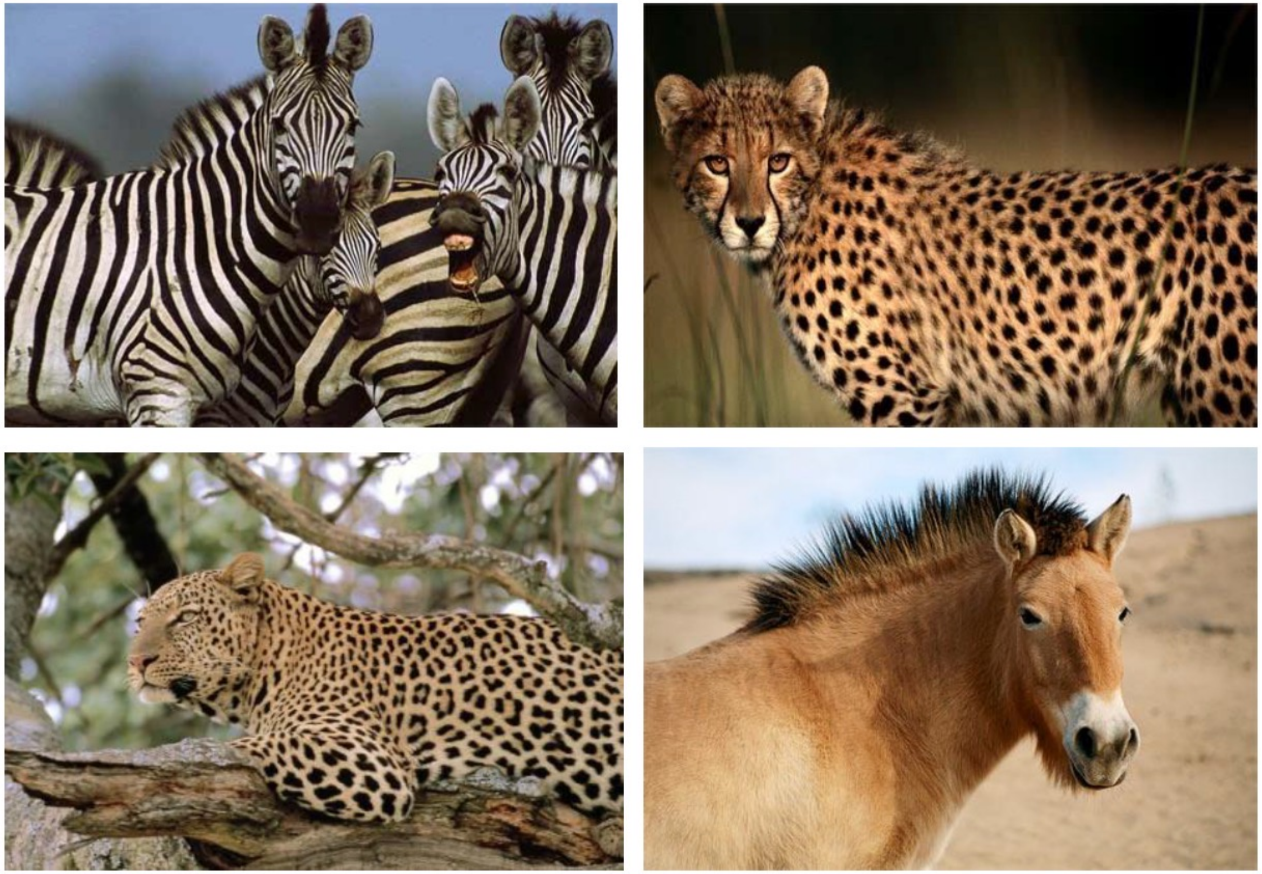


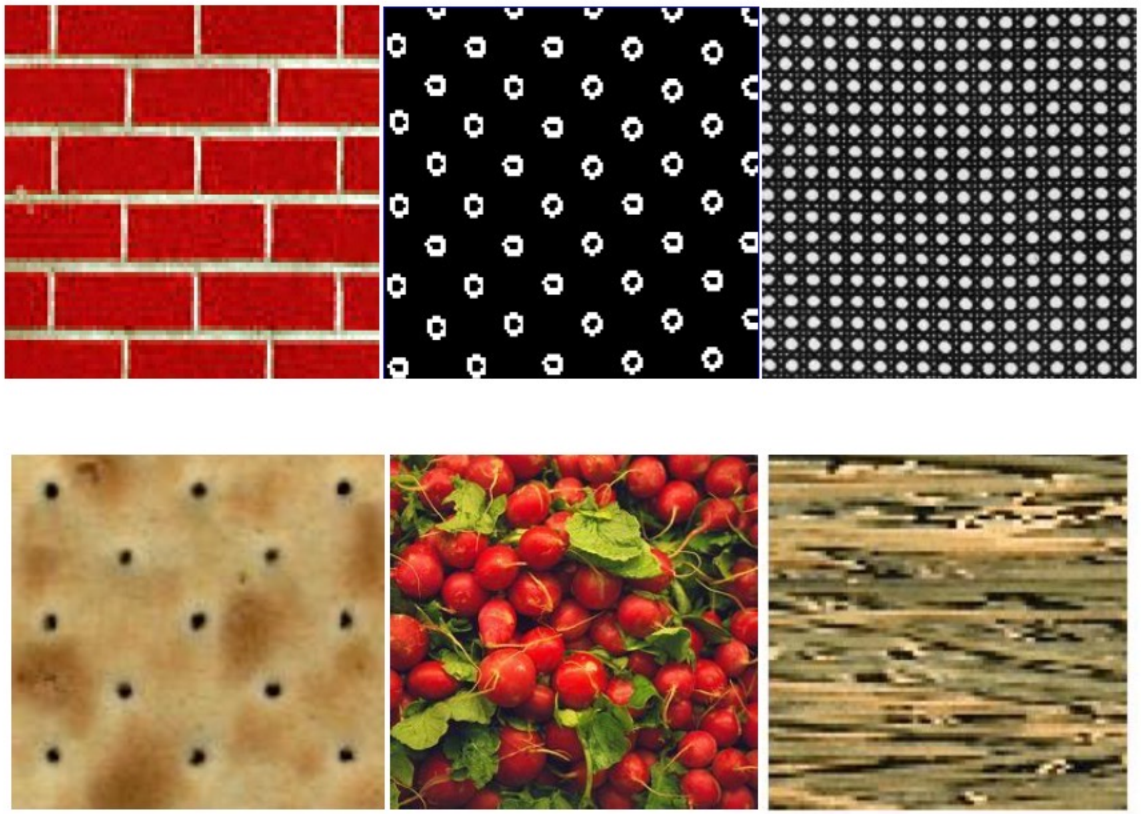


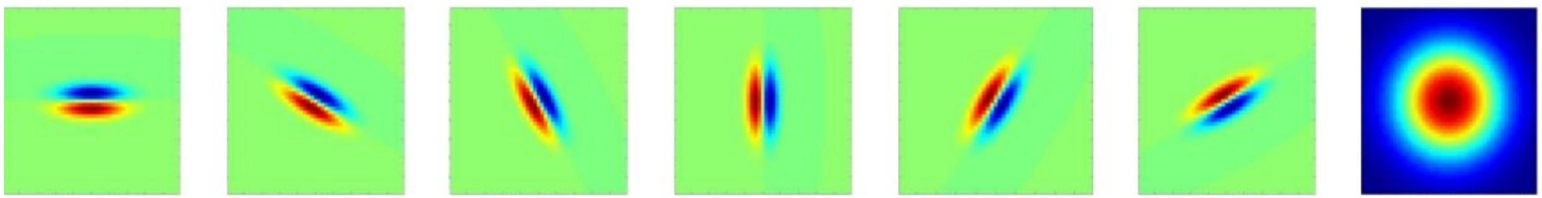
Derivative filters

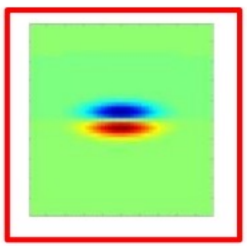
- Derivatives of Gaussian
- Can the values of a derivative filter be negative?
- What should the values sum to?
  - **Zero:** no response in constant regions

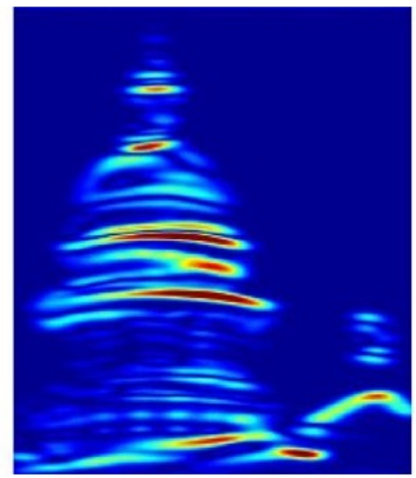
Texture

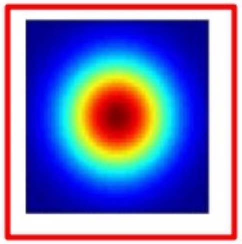


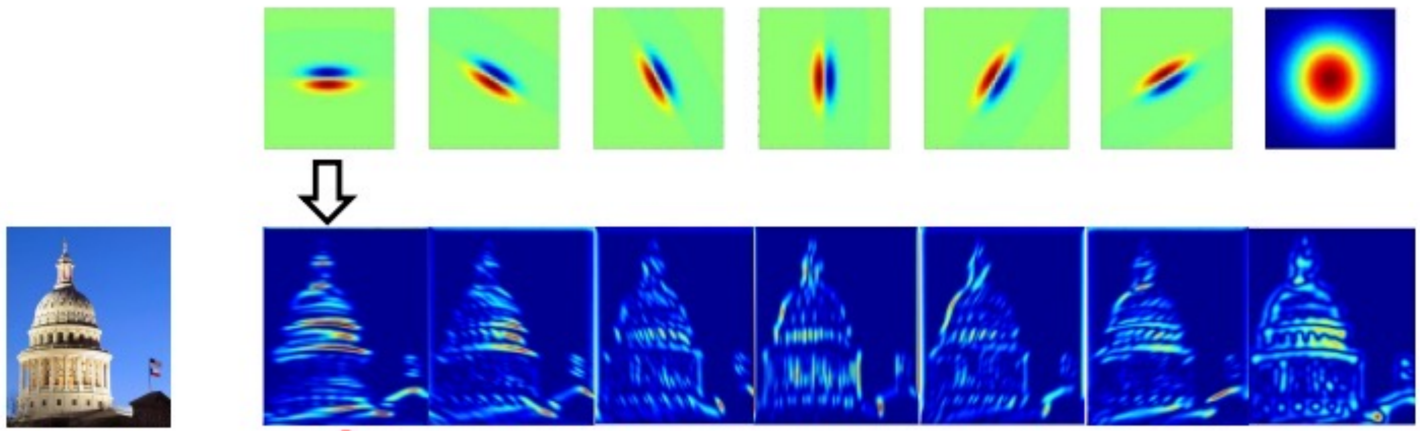




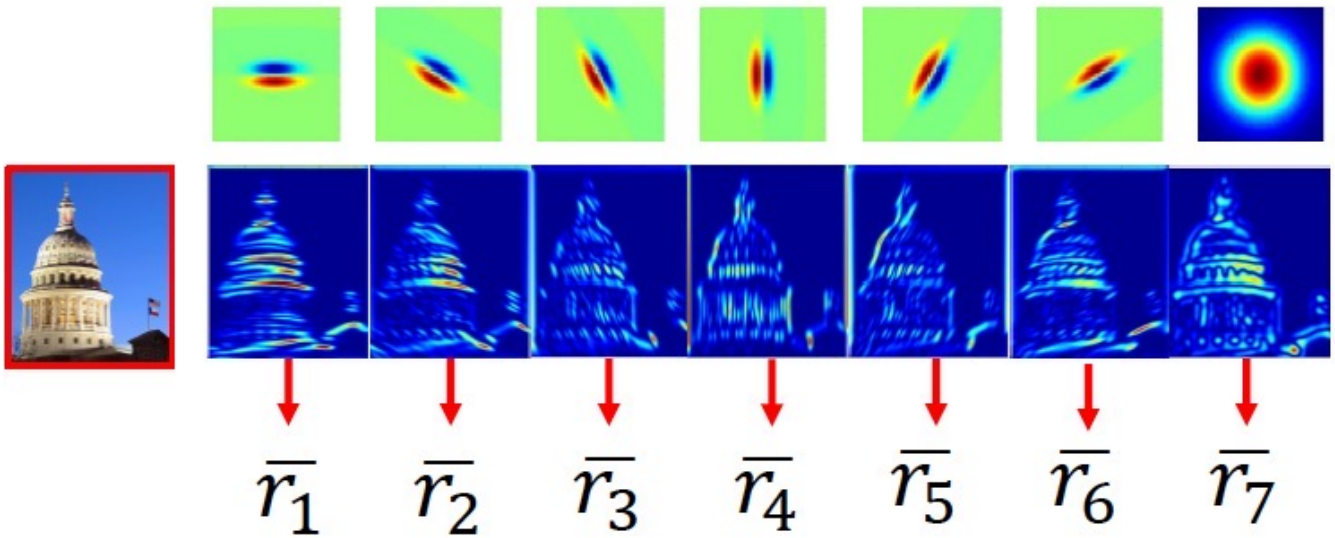


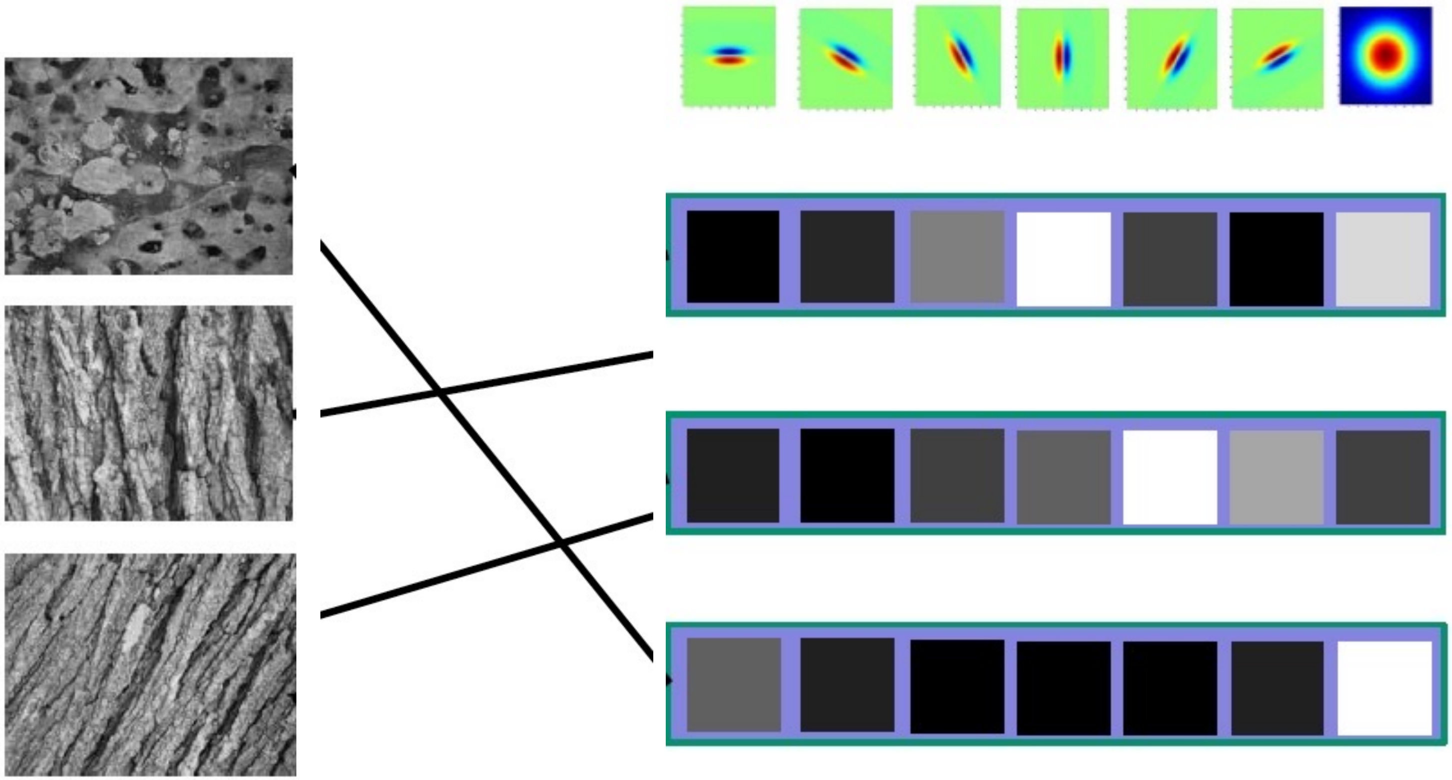






$$\boldsymbol{r_1} = [r_{11}, r_{12}, \cdots r_{1\times n}]$$


$\bar{r}_1$ $\bar{r}_2$ $\bar{r}_3$ $\bar{r}_4$ $\bar{r}_5$ $\bar{r}_6$ $\bar{r}_7$



How to design ?