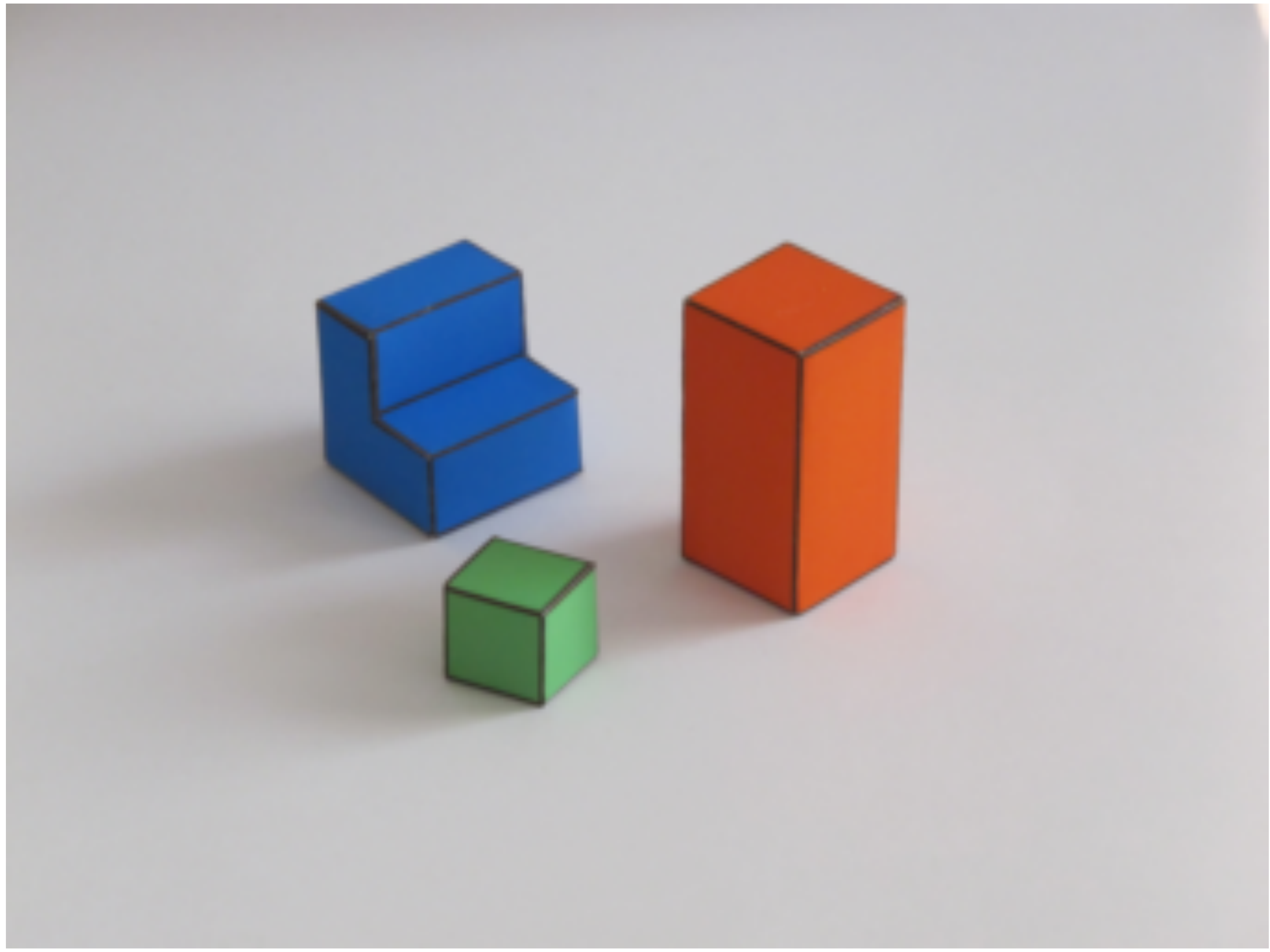
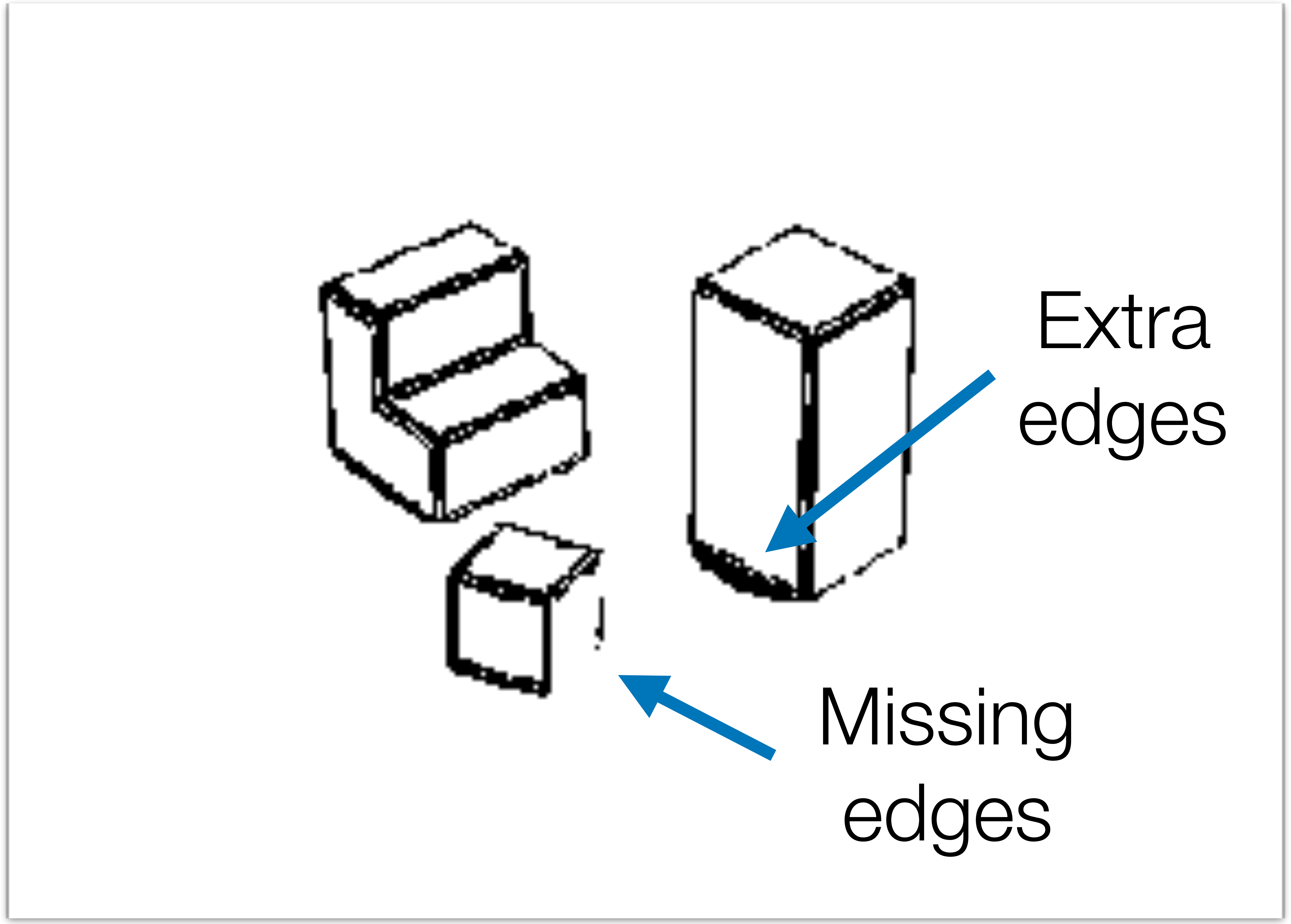
Lecture 2: Image filtering

- PS1 due next Tuesday
- Updated office hours next week, due to holiday. New times will be on Piazza.
- Questions?

Recall last week...



Input image



Edges

In this lecture

What *other* transformations can we do?

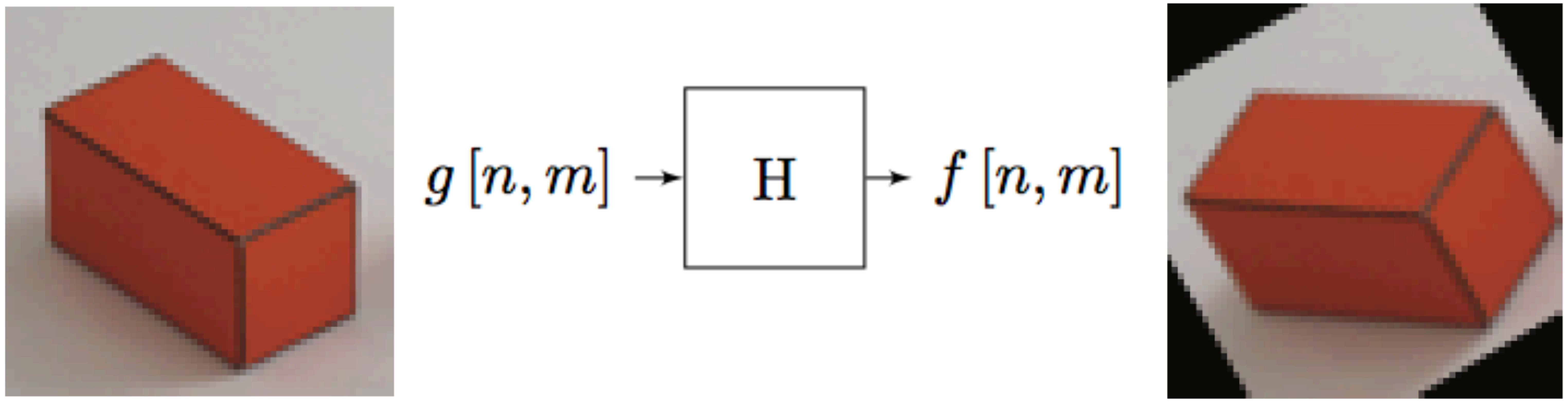
Filtering

$$g[n, m] \rightarrow \boxed{H} \rightarrow f[n, m]$$

Our goal: remove unwanted sources of variation, and keep the information relevant for whatever task we need to solve.

Source: Torralba, Freeman, Isola

Linear filtering



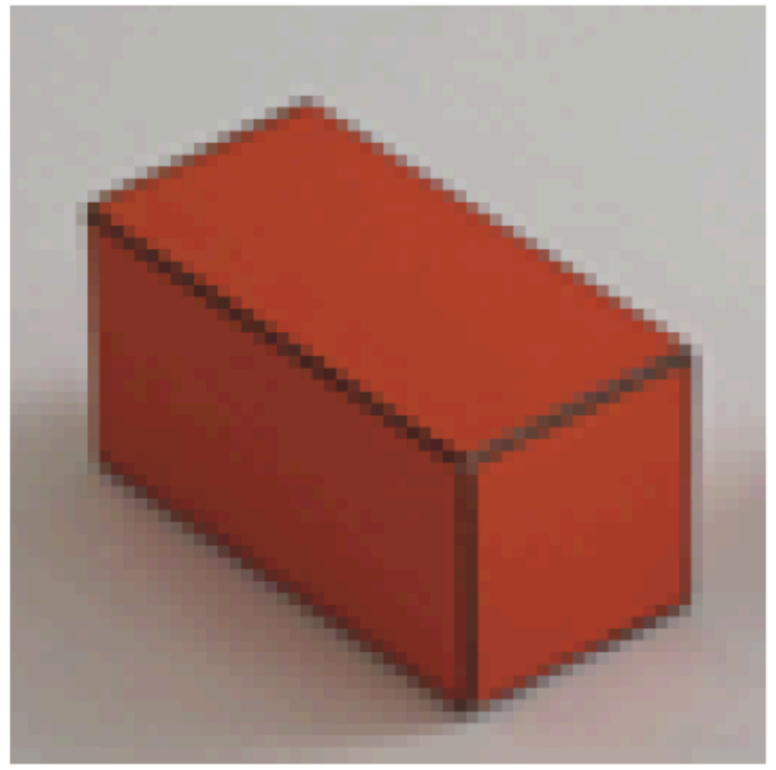
Very general! For a filter, H, to be linear, it has to satisfy:

$$H(a[m,n] + b[m,n]) + H(a[m,n]) + H(b[m,n])$$

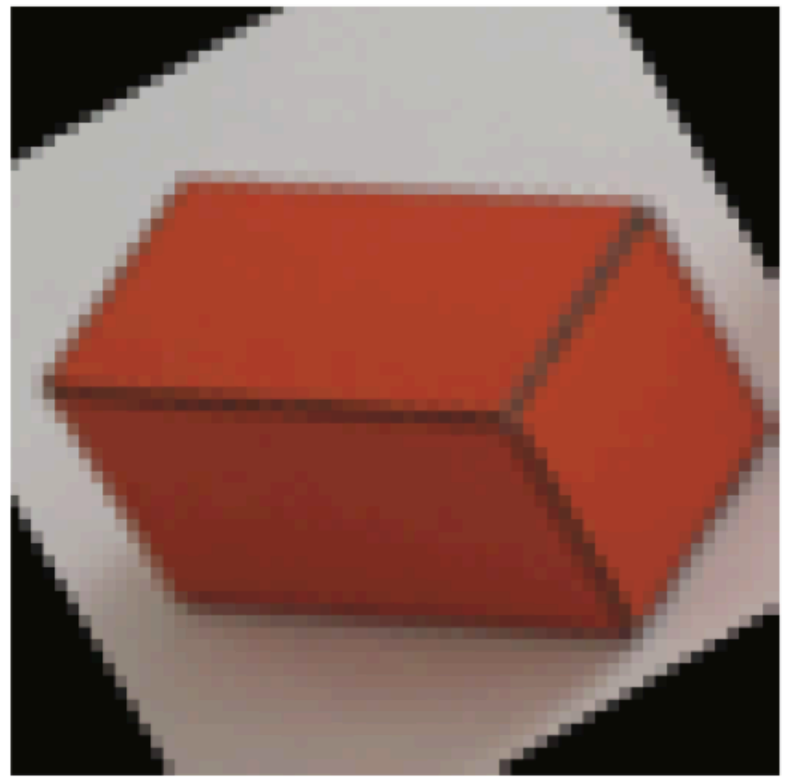
$$H(Ca[m,n]) = CH(a[m,n])$$

Source: Torralba, Freeman, Isola

Linear filtering



$$g[n,m] \rightarrow \boxed{\mathrm{H}} \rightarrow f[n,m]$$



A linear filter in its most general form can be written as (for a 1D signal of length N):

$$f[n] = \sum_{k=0}^{N-1} h[n,k]\, g[k]$$

In matrix form:

$$\begin{bmatrix} f[0] \\ f[1] \\ \vdots \\ f[M-1] \end{bmatrix} = \begin{bmatrix} h[0,0] & h[0,1] & \dots & h[0,N-1] \\ h[1,0] & h[1,1] & \dots & h[1,N-1] \\ \vdots & \vdots & \vdots & \vdots \\ h[M-1,0] & h[M-1,1] & \dots & h[M-1,N-1] \end{bmatrix} \begin{bmatrix} g[0] \\ g[1] \\ \vdots \\ g[N-1] \end{bmatrix}$$

Source: Torralba, Freeman, Isola

$$\begin{bmatrix} f[0] \\ f[1] \\ \vdots \\ f[M-1] \end{bmatrix} = \begin{bmatrix} h[0,0] & h[0,1] & \dots & h[0,N-1] \\ h[1,0] & h[1,1] & \dots & h[1,N-1] \\ \vdots & \vdots & \vdots & \vdots \\ h[M-1,0] & h[M-1,1] & \dots & h[M-1,N-1] \end{bmatrix} \begin{bmatrix} g[0] \\ g[1] \\ \vdots \\ g[N-1] \end{bmatrix}$$

Why handle each spatial position differently?

Want translation invariance!

Photo by Fredo Durand, slide by Torralba, Freeman, Isola

Image denoising

Photo by Fredo Durand

Moving average

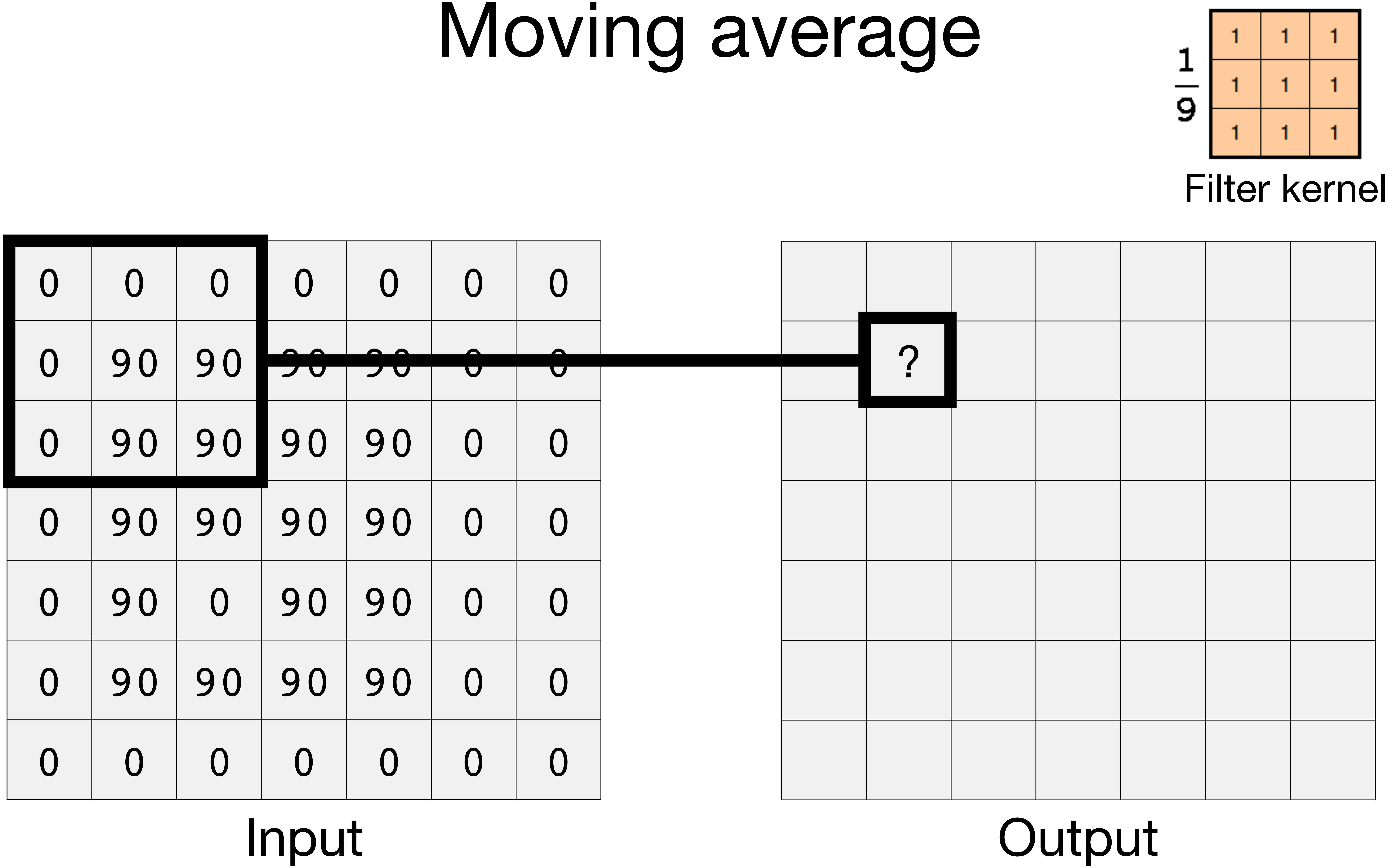
- Let’s replace each pixel with a weighted average of its neighborhood
- The weights are called the **filter kernel**
- What are the weights for the average of a 3x3 neighborhood?

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

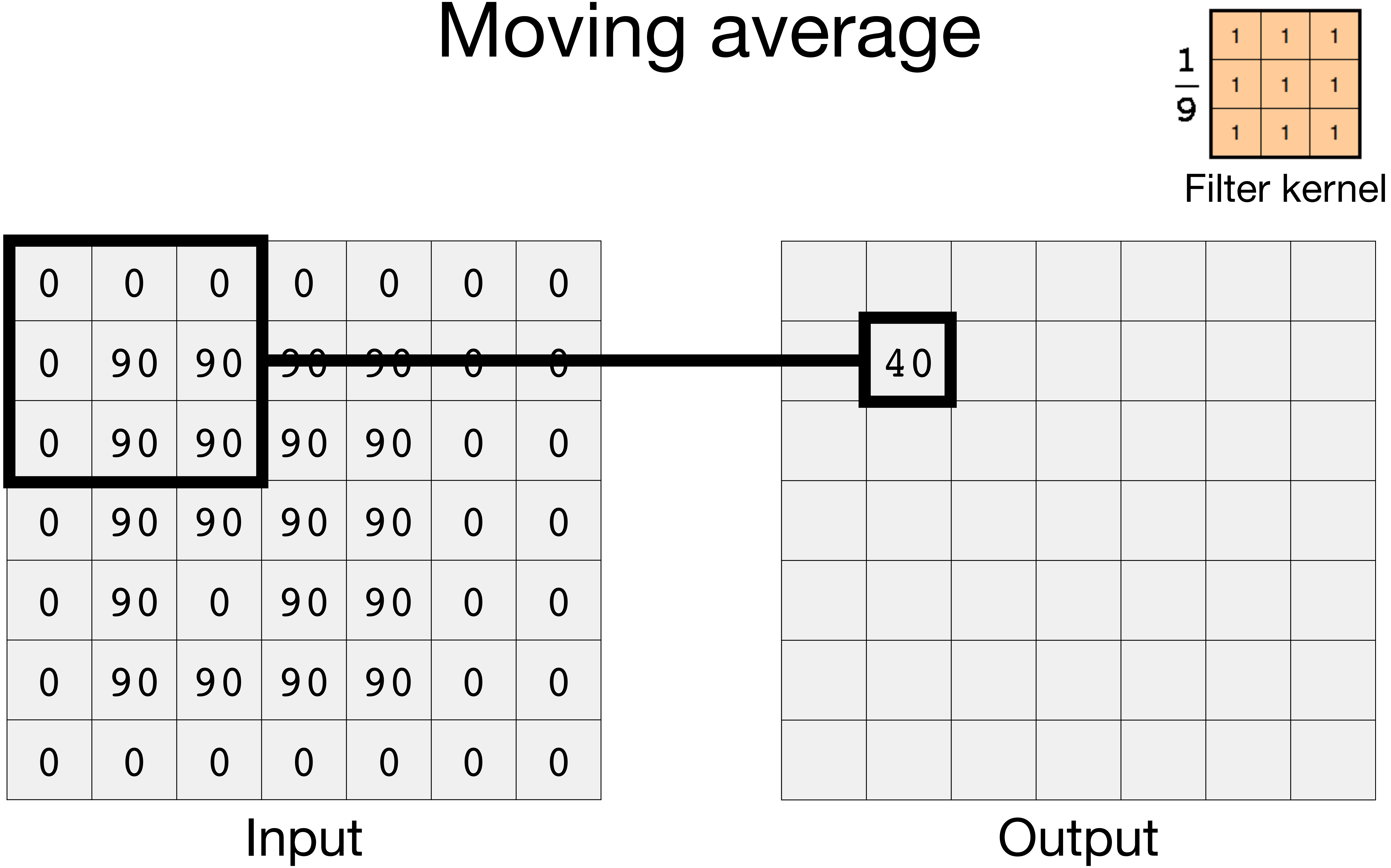
“box filter”

Source: D. Lowe

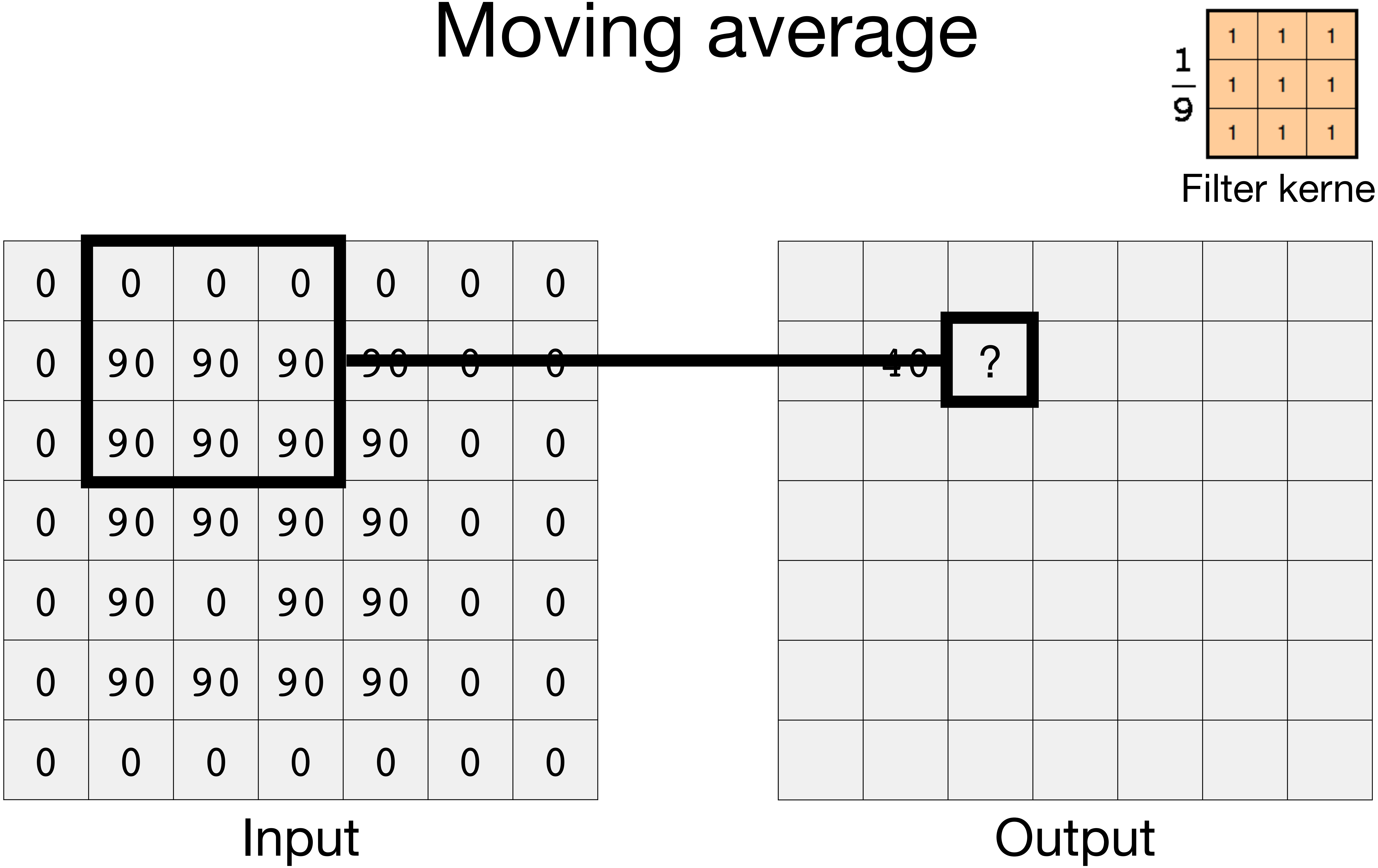
Moving average



Moving average



Moving average



Moving average

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

Filter kernel

0	0	0	0	0	0	0
0	90	90	90	90	0	0
0	90	90	90	90	0	0
0	90	90	90	90	0	0
0	90	0	90	90	0	0
0	90	90	90	90	0	0
0	0	0	0	0	0	0

Input

	40	60				

Output

Moving average

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

Filter kernel

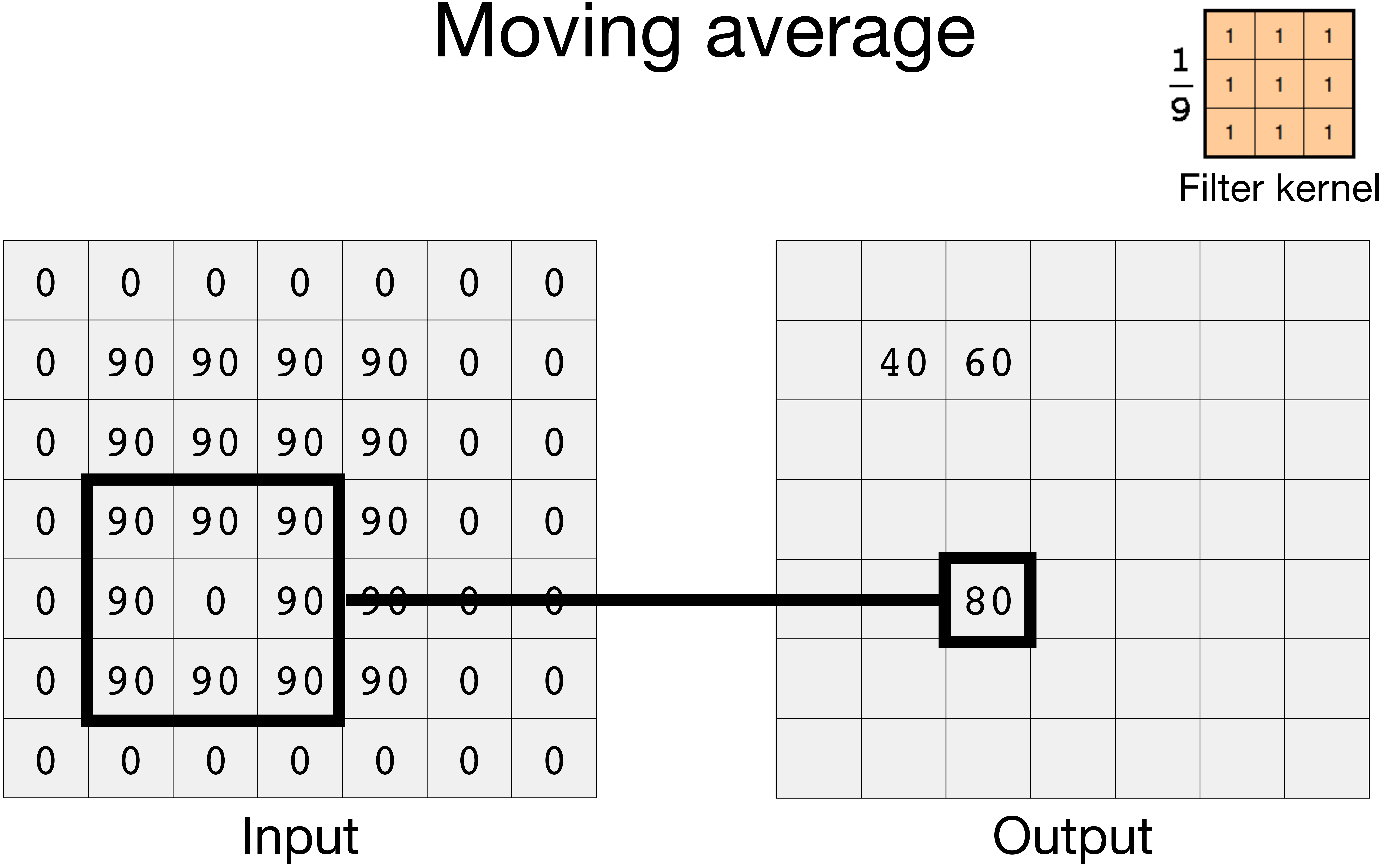
0	0	0	0	0	0	0
0	90	90	90	90	0	0
0	90	90	90	90	0	0
0	90	90	90	90	0	0
0	90	0	90	90	0	0
0	90	90	90	90	0	0
0	0	0	0	0	0	0

Input

	40	60				
		?				

Output

Moving average



Moving average

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

Filter kernel

0	0	0	0	0	0	0
0	90	90	90	90	0	0
0	90	90	90	90	0	0
0	90	90	90	90	0	0
0	90	0	90	90	0	0
0	90	90	90	90	0	0
0	0	0	0	0	0	0

Input

	40	60	60	40	20	
	60	90	60	40	20	
	50	80	80	60	30	
	50	80	80	60	30	
	30	50	50	40	20	

Output

Moving average

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Filter kernel

0	0	0	0	0	0	0
0	90	90	90	90	0	0
0	90	90	90	90	0	0
0	90	90	90	90	0	0
0	90	0	90	90	0	0
0	90	90	90	90	0	0
0	0	0	0	0	0	0

Input

	40	60	60	40	20	
	60	90	60	40	20	
	50	80	80	60	30	
?	50	80	80	60	30	
	30	50	50	40	20	

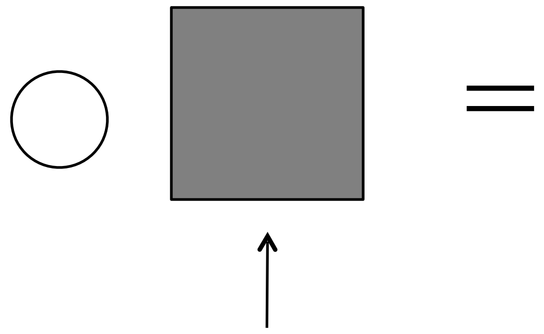
Output

Handling boundaries

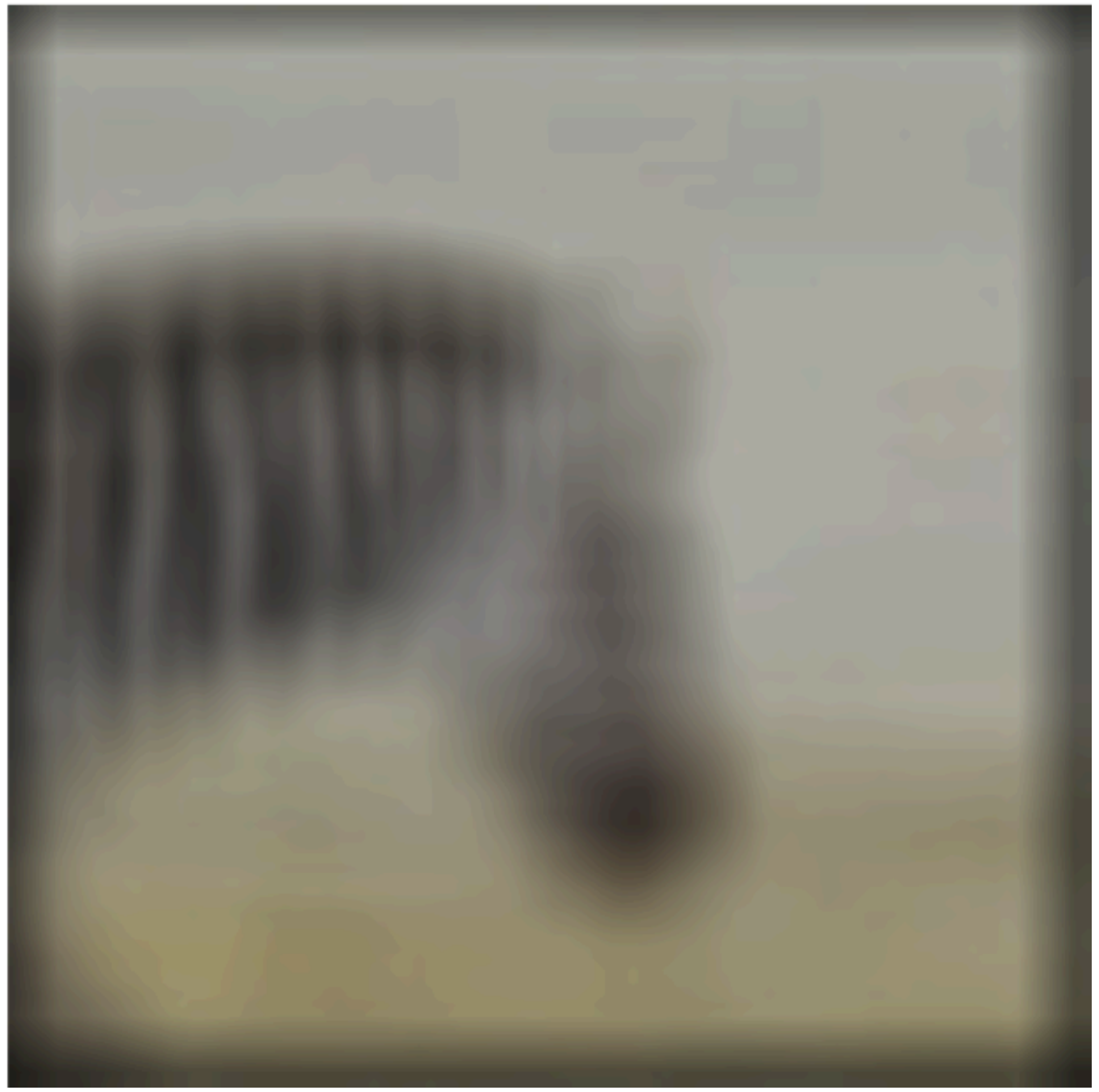
Source: Torralba, Freeman, Isola

Handling boundaries

Zero padding

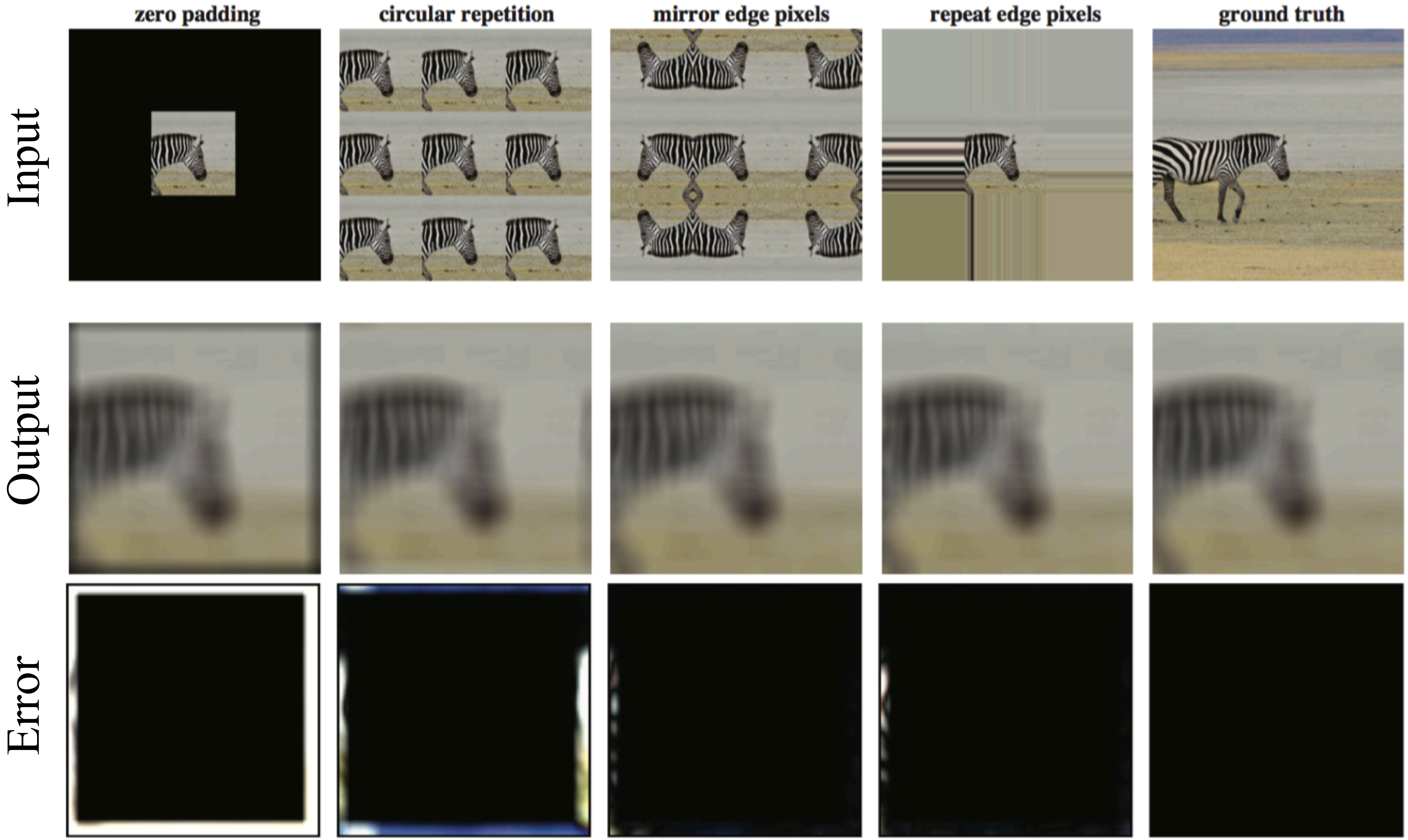


11x11 box



Source: Torralba, Freeman, Isola

Handling boundaries



Source: Torralba, Freeman, Isola

Moving average

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

Filter kernel

0	0	0	0	0	0	0	0
0	0	90	90	90	90	0	0
0	0	90	90	90	90	0	0
0	0	90	90	90	90	0	0
0	0	90	0	90	90	0	0
0	0	90	90	90	90	0	0
0	0	0	0	0	0	0	0

Input

	40	60	60	40	20	
	60	90	60	40	20	
	50	80	80	60	30	
30	50	80	80	60	30	
	30	50	50	40	20	

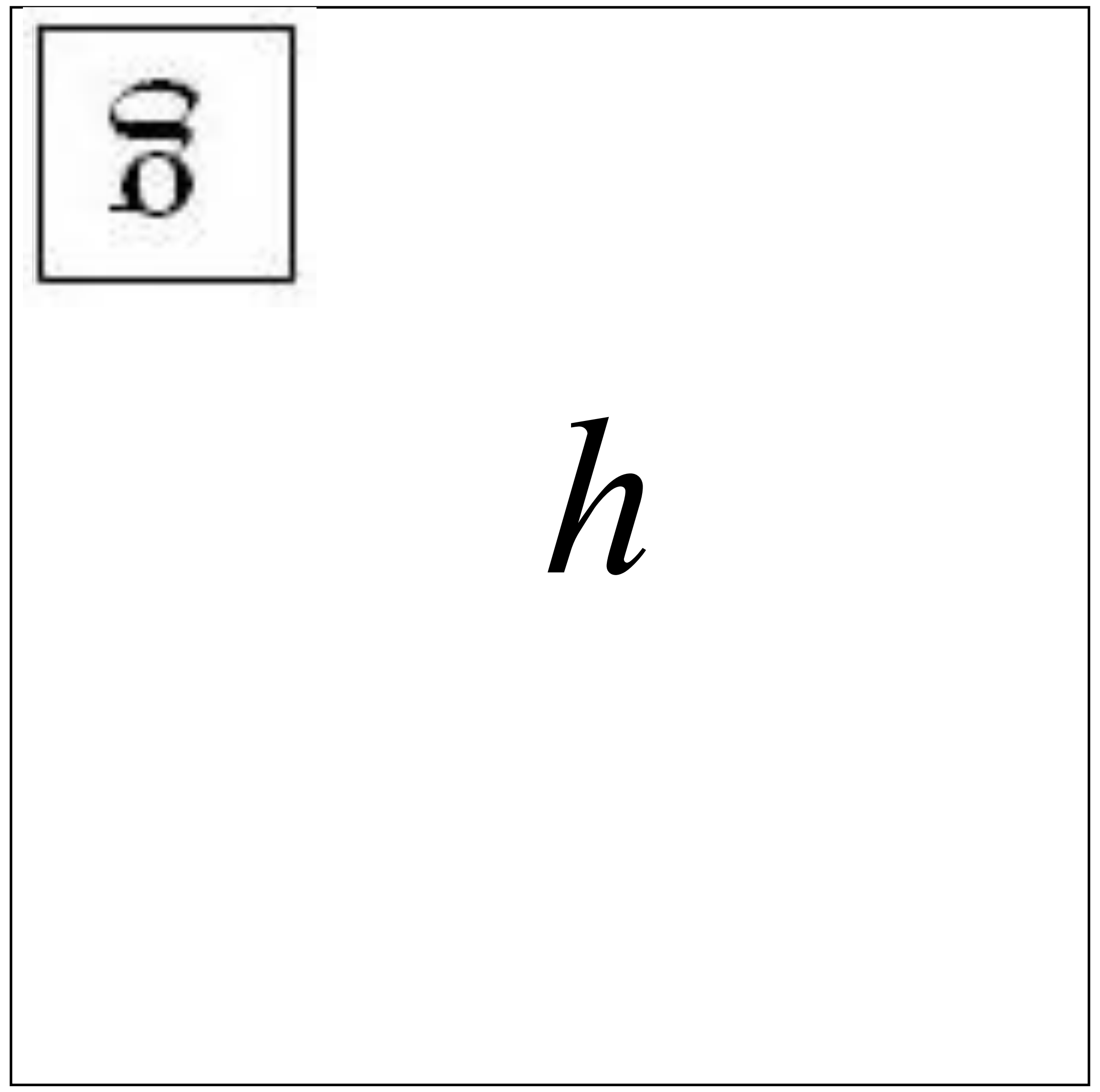
Output

Convolution

- Let *h* be the image and *g* be the kernel. The output of convolving *h* with *g* is:

$$f\left[m,n\right] = h \circ g = \sum_{k,l} h\left[m-k, n-l\right] g\left[k,l\right]$$

Convention:
kernel is “flipped”



Source: F. Durand

Properties of the convolution

Commutative

$$h[n] \circ g[n] = g[n] \circ h[n]$$

Associative

$$h[n] \circ g[n] \circ q[n] = h[n] \circ (g[n] \circ q[n]) = (h[n] \circ g[n]) \circ q[n]$$

Distributive with respect to the sum

$$h[n] \circ (f[n] + g[n]) = h[n] \circ f[n] + h[n] \circ g[n]$$

Why flip the kernel?

$$f[m,n] = h \circ g = \sum_{k,l} h[m-k, n-l]\, g[k,l]$$

Indexes go backward!

Cross correlation

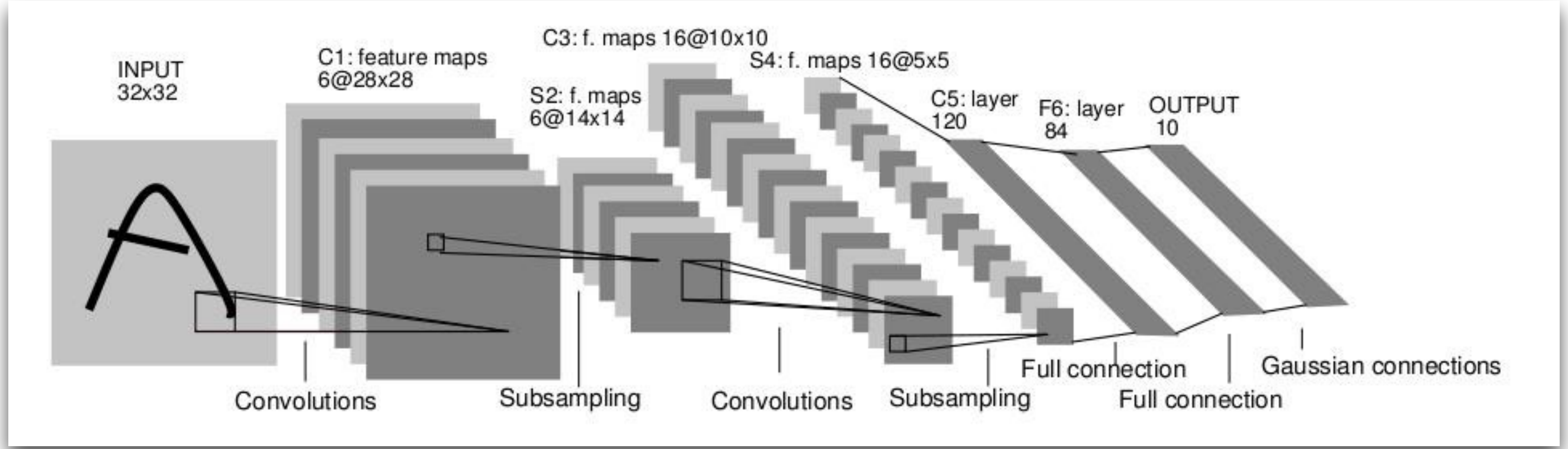
$$f[m,n] = h * g = \sum_{k,l} h[m+k, n+l]\, g[k,l]$$

No flipping!

- Sometimes called just **correlation**
- Neither associative nor commutative
- In the literature, people often just call both “convolution”
- Filters often symmetric, so won’t matter

Convolutional neural networks

- Neural network with specialized connectivity structure
- Mostly just convolutions!

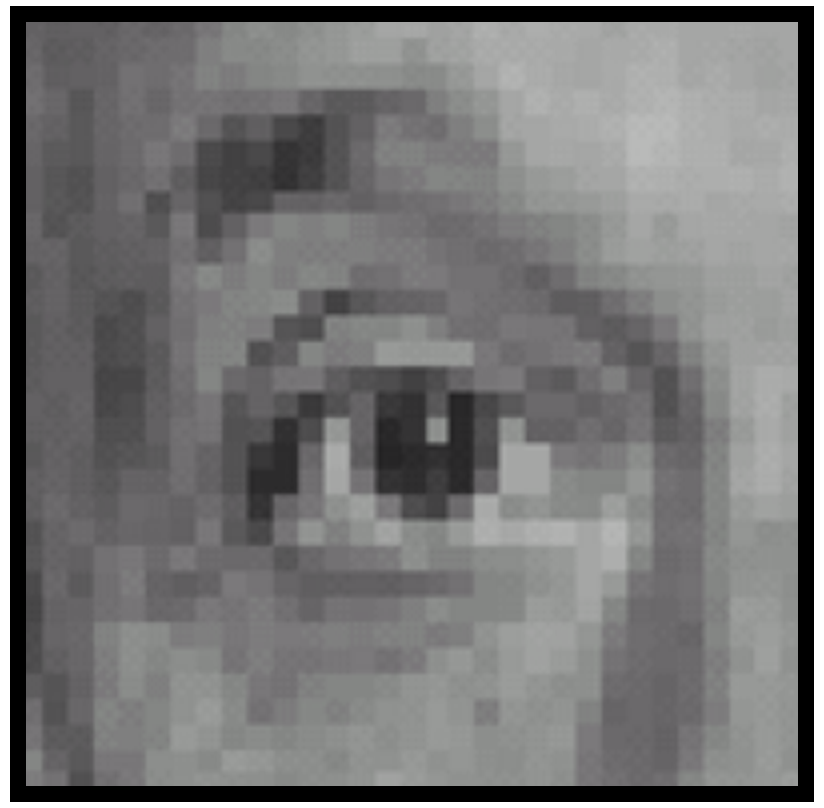


(LeCun et al. 1989)

Source: Torralba, Freeman, Isola

Filtering examples

Practice with linear filters



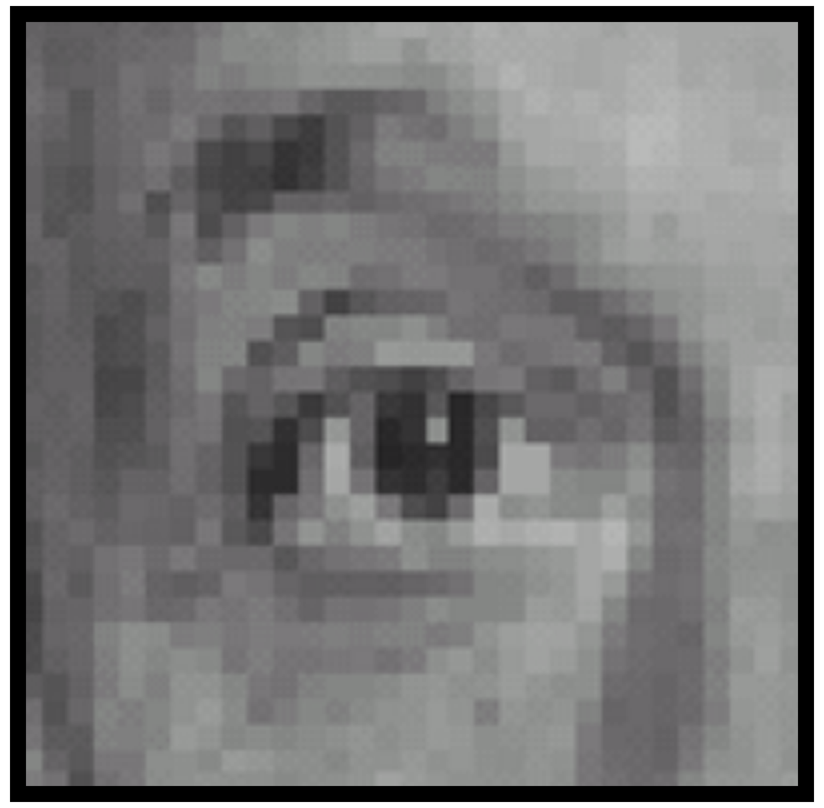
Original

0	0	0
0	1	0
0	0	0

?

Source: D. Lowe

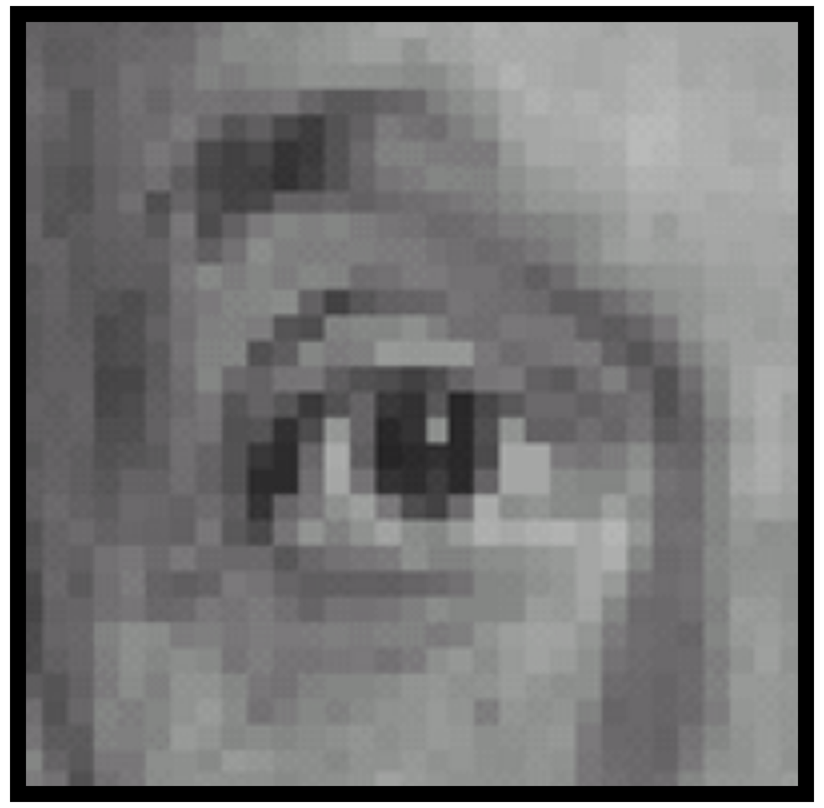
Practice with linear filters



Original

0	0	0
0	1	0
0	0	0

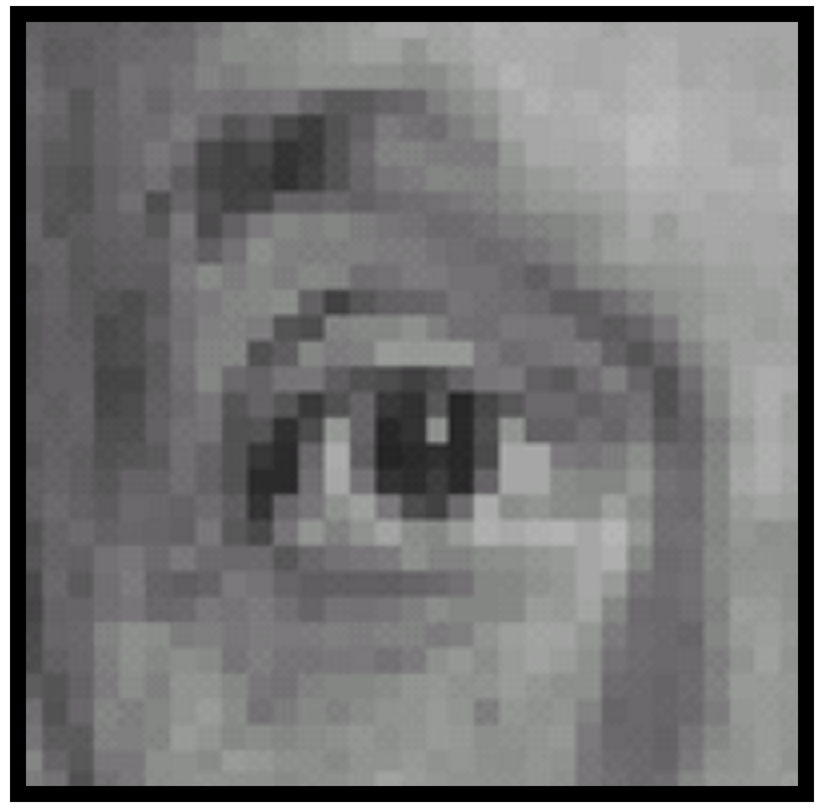
“Impulse”



Filtered
(no change)

Source: D. Lowe

Practice with linear filters



Original

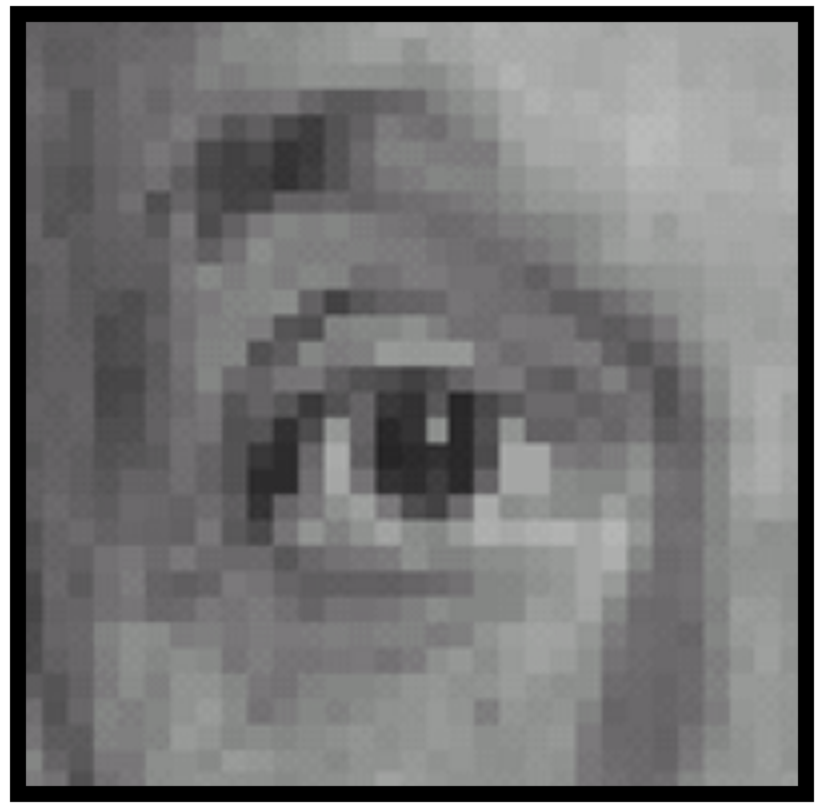
0	0	0
0	0	1
0	0	0

“Translated Impulse”

?

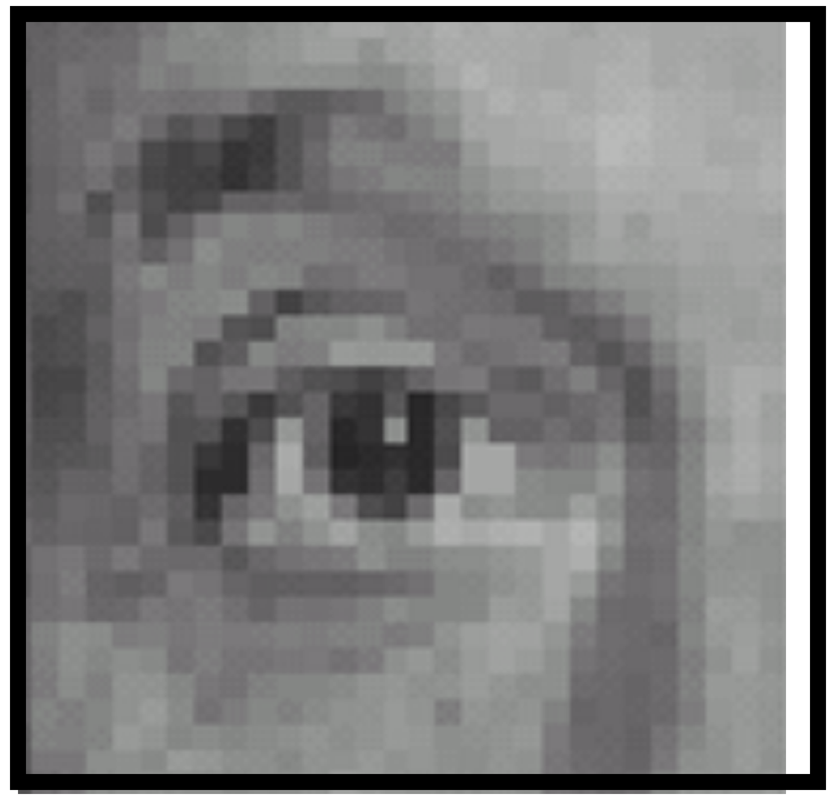
Source: D. Lowe

Practice with linear filters



Original

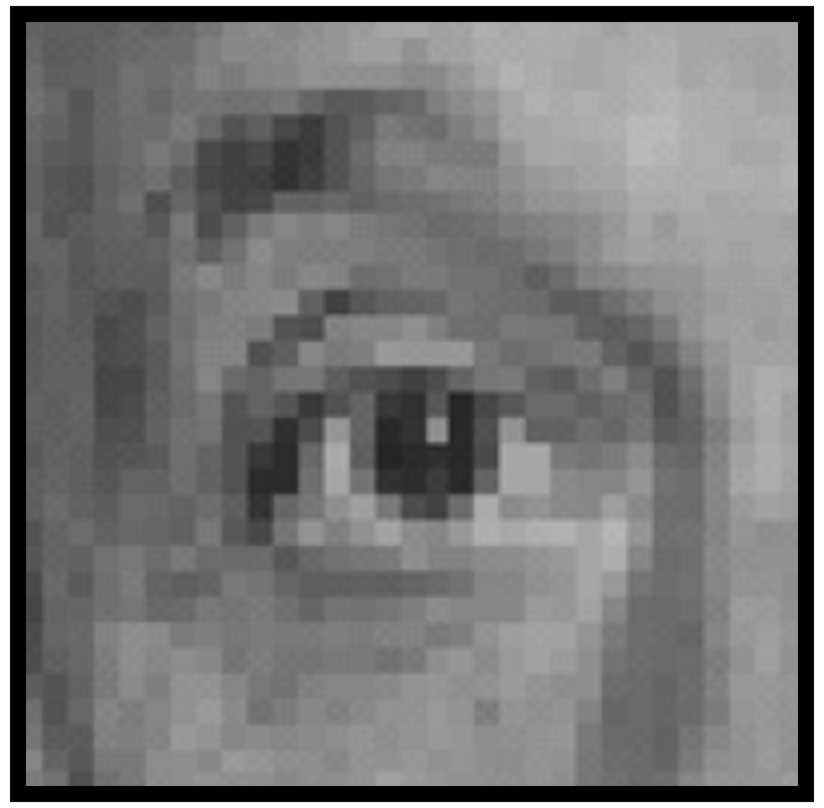
0	0	0
0	0	1
0	0	0



Shifted *left*
By 1 pixel

Source: D. Lowe

Practice with linear filters



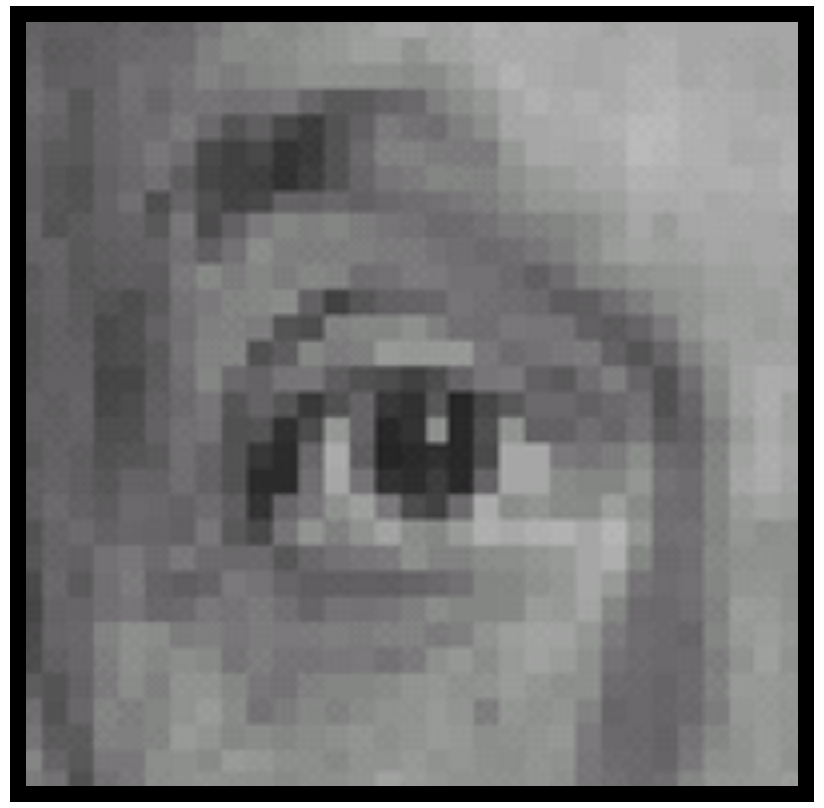
Original

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

?

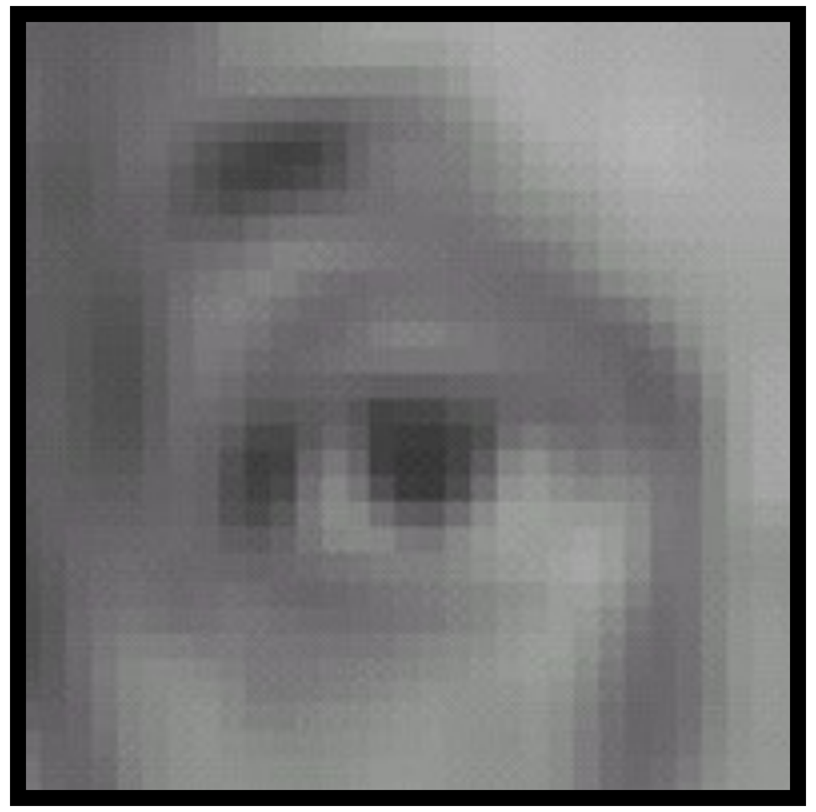
Source: D. Lowe

Practice with linear filters



Original

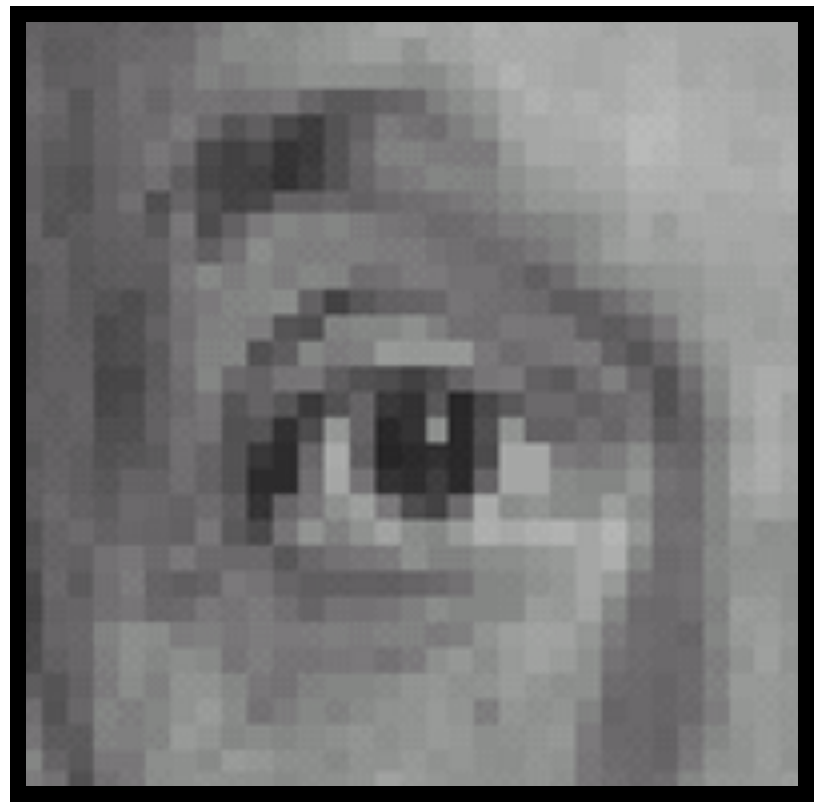
$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$



Blur (with a box filter)

Source: D. Lowe

Practice with linear filters



Original

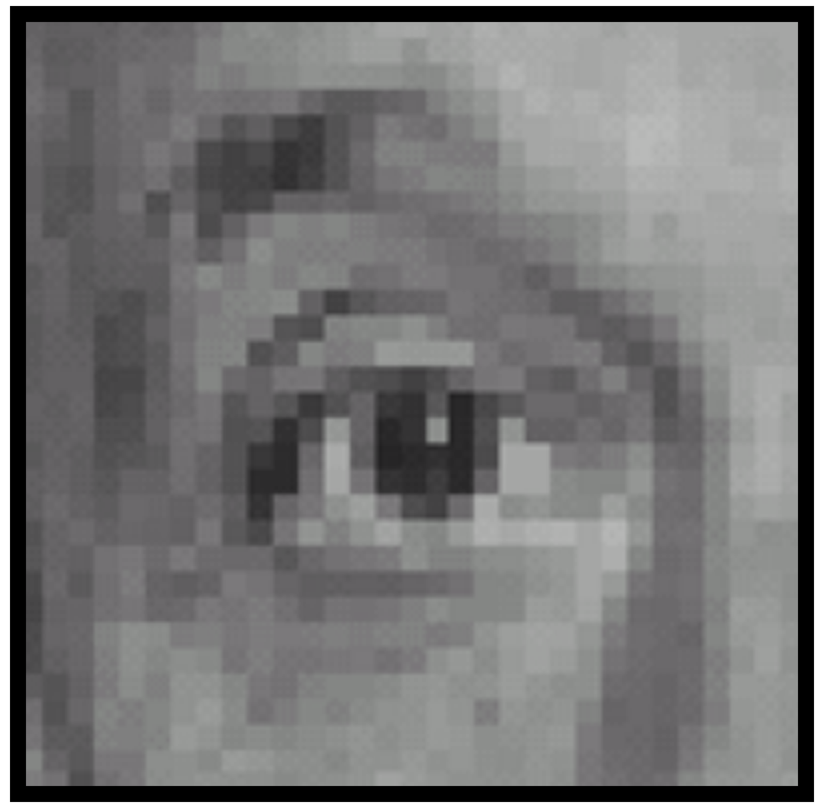
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

?

(Note that filter sums to 1)

Source: D. Lowe

Practice with linear filters



Original

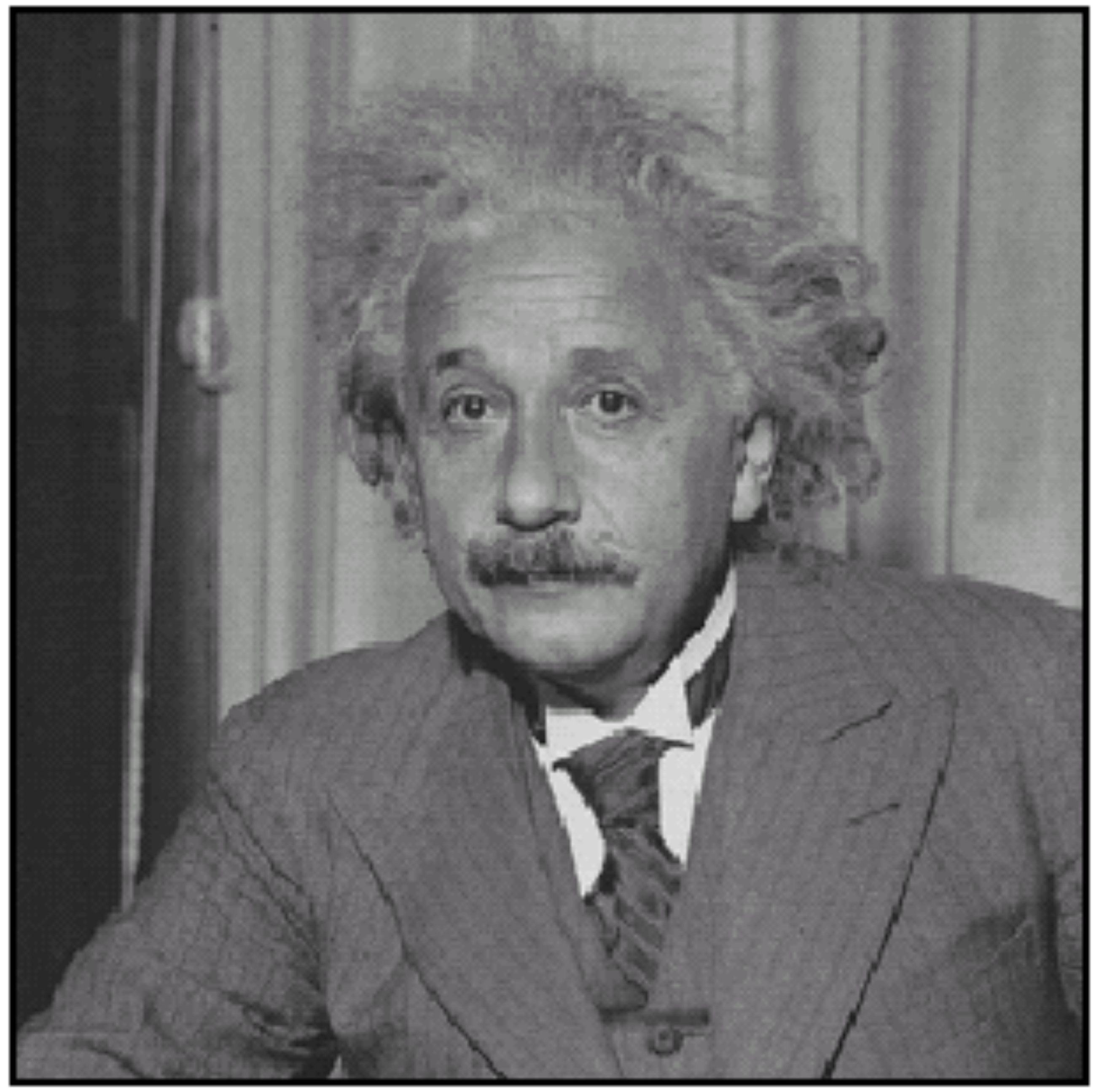
$$\begin{array}{|c|c|c|}\hline 0 & 0 & 0 \\ \hline 0 & 2 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array} \; - \; \frac{1}{9}\begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

Sharpening filter

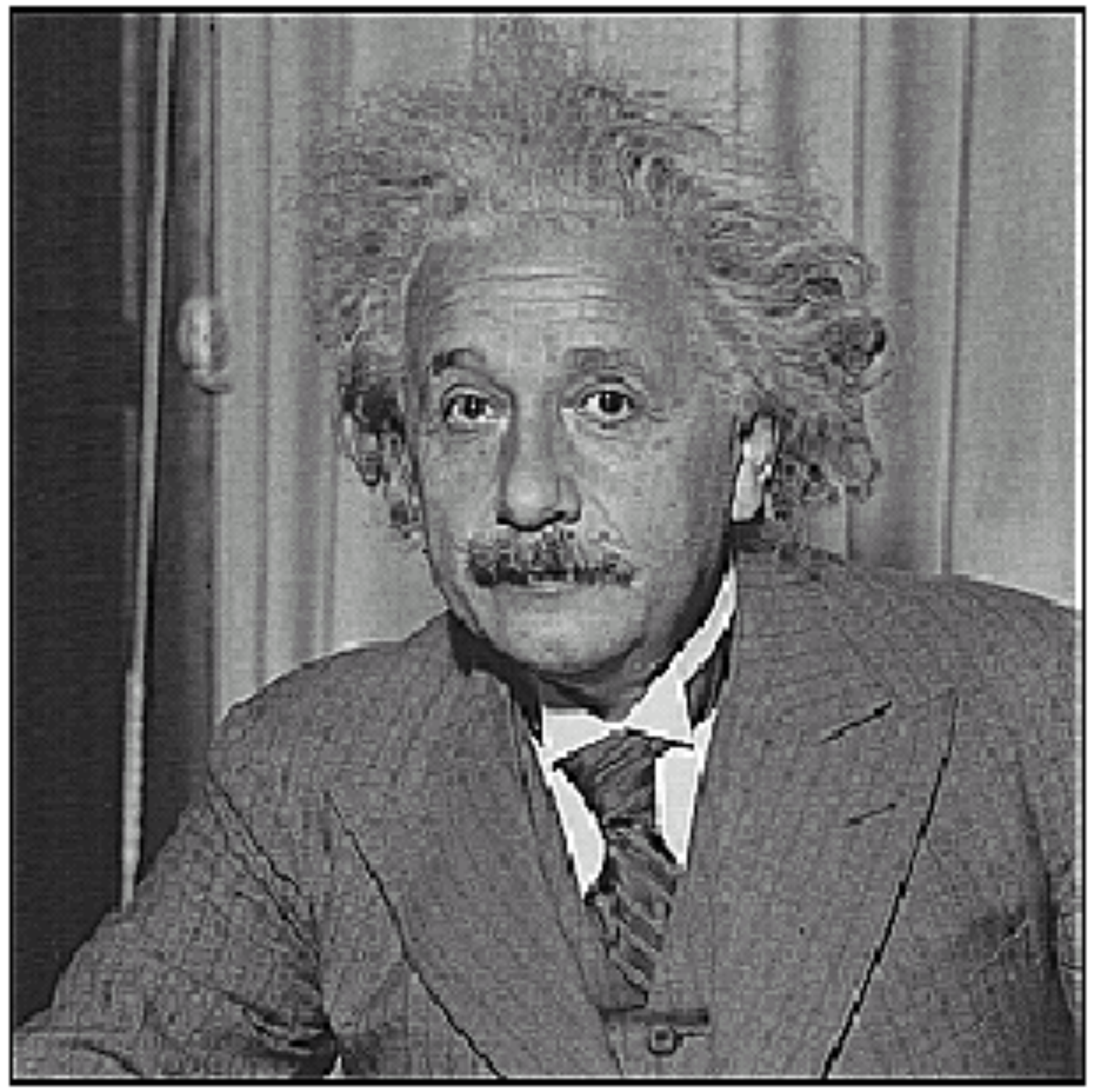
- Accentuates differences with local average

Source: D. Lowe

Sharpening



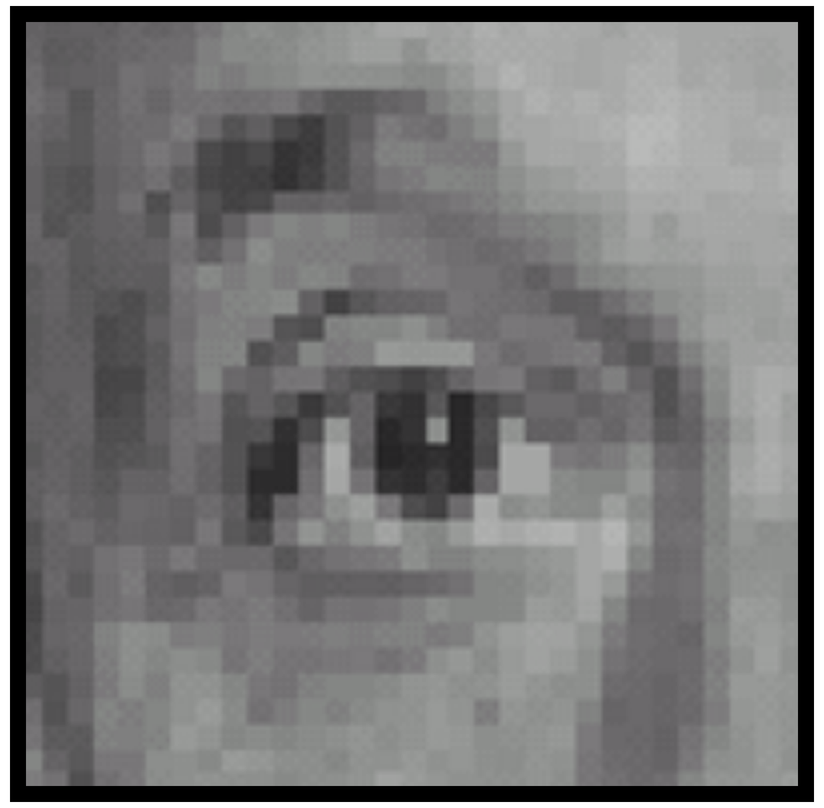
before



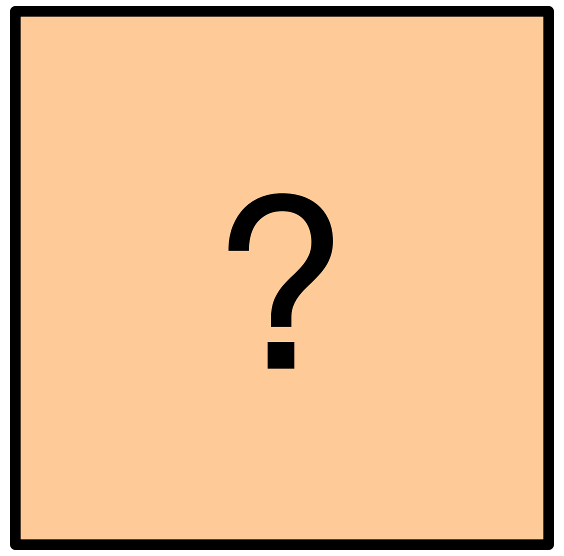
after

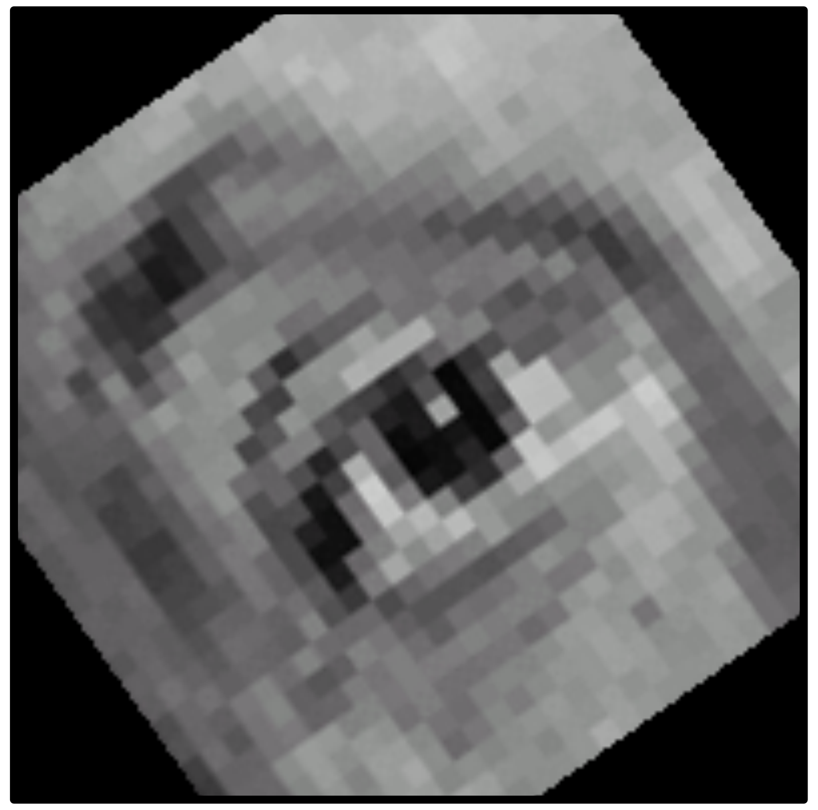
Source: D. Lowe

Practice with linear filters



Original





Can you do this?

Rectangular filter

g[m,n]

$\otimes$

h[m,n]

=

f[m,n]

Source: Torralba, Freeman, Isola

Rectangular filter

g[m,n]

$\otimes$

h[m,n]

=

f[m,n]

Source: Torralba, Freeman, Isola

“Naturally” occurring filters

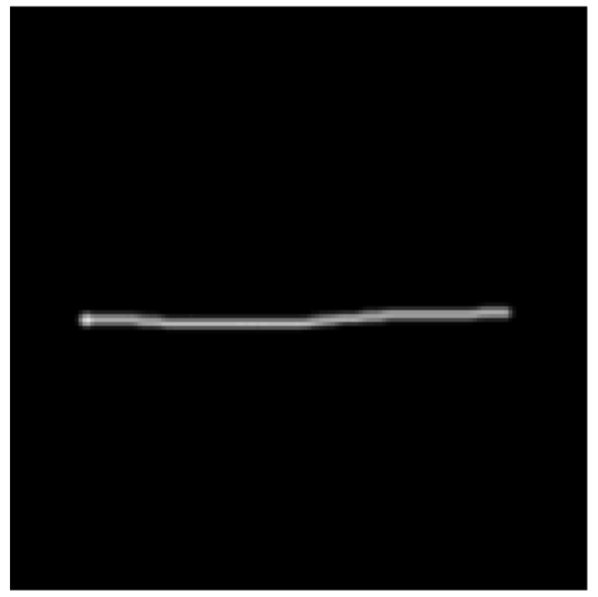
Input image

Motion blur

Source: Torralba, Freeman, Isola

“Naturally” occurring filters

Input image

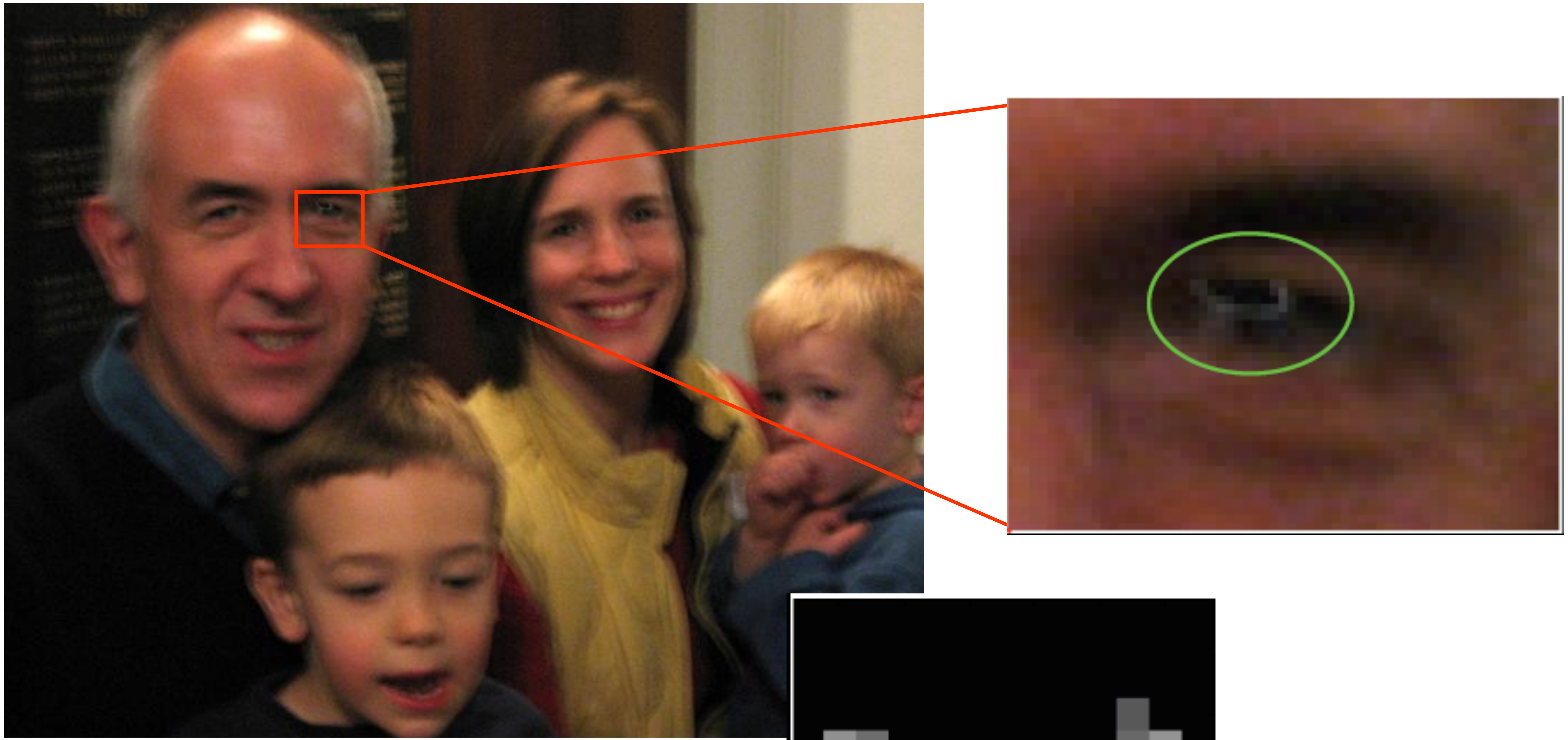


Convolution weights

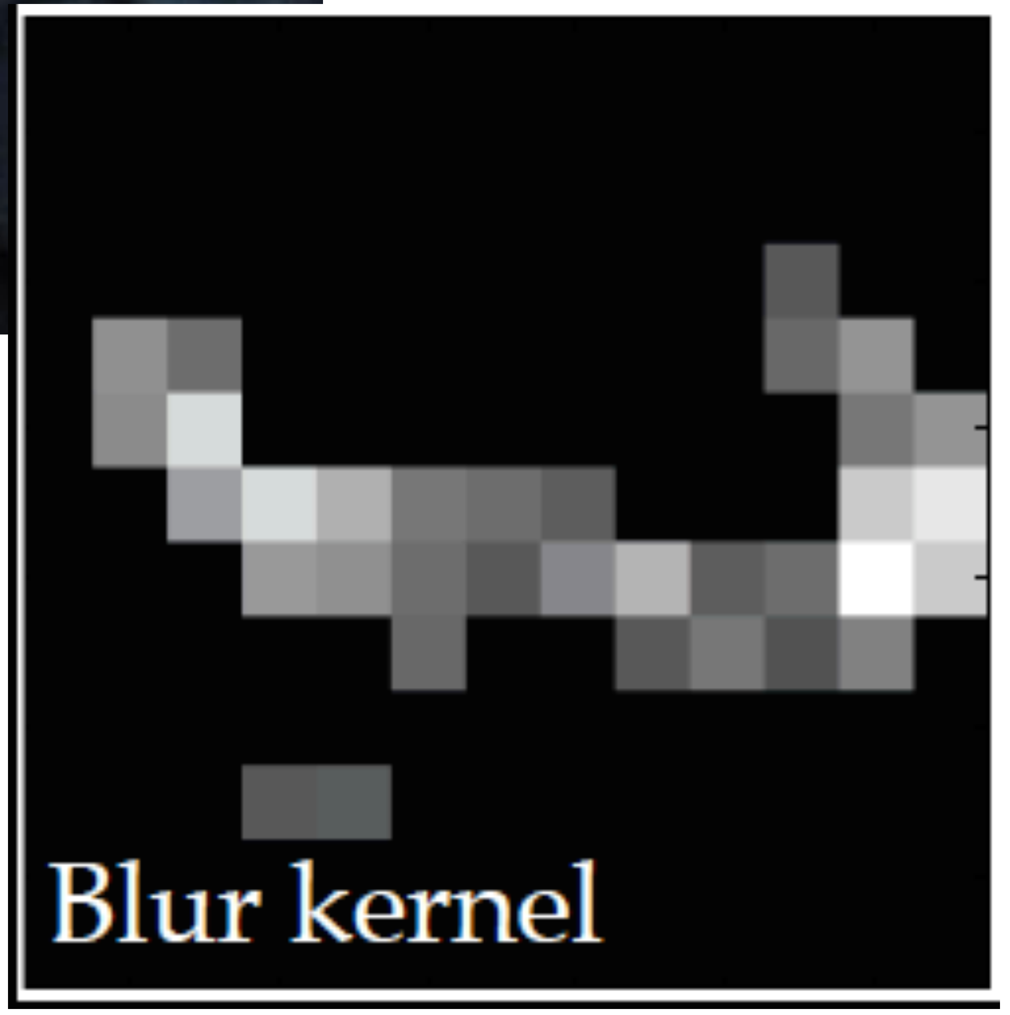
Convolution output

Source: Torralba, Freeman, Isola

Camera shake



(from Fergus et al, 2007)



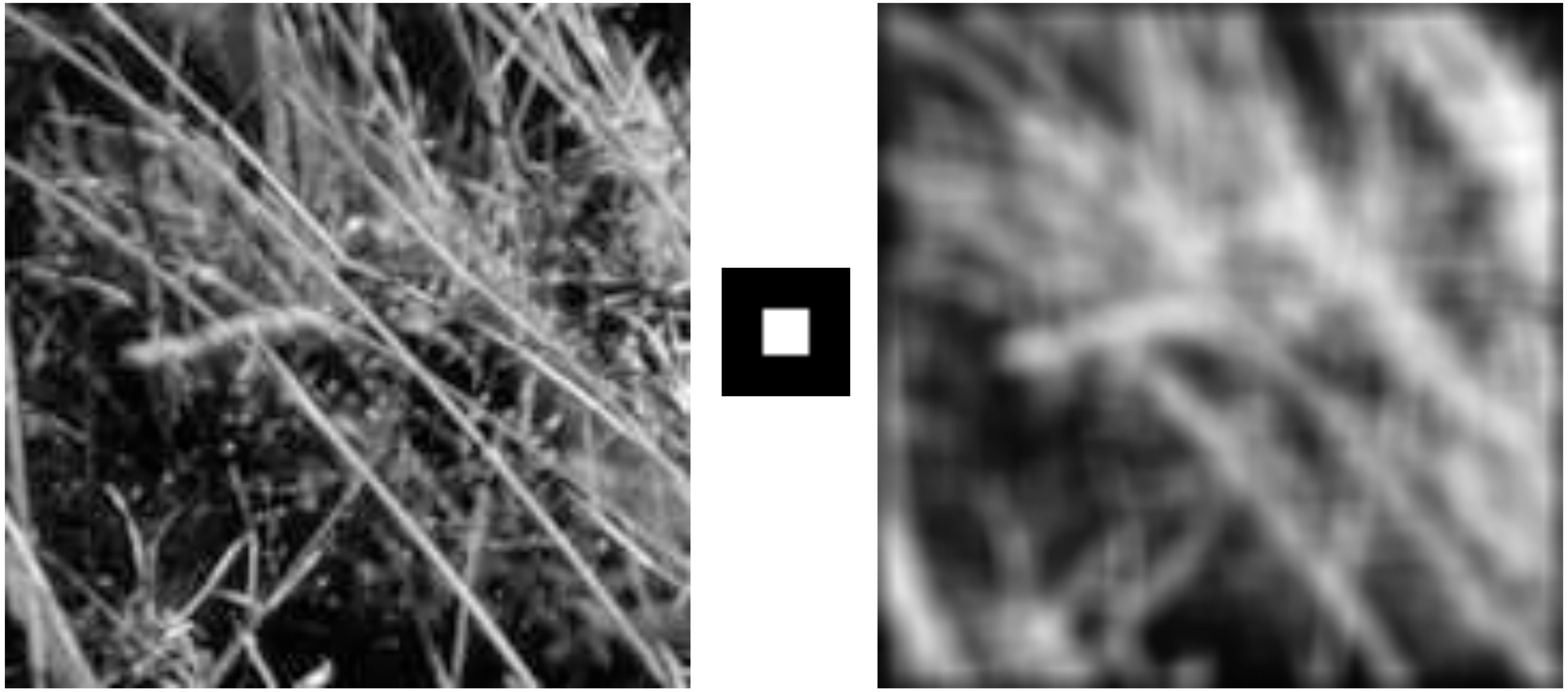
Source: Torralba, Freeman, Isola

Blur occurs in many natural situations

Source: Torralba, Freeman, Isola

Smoothing with box filter revisited

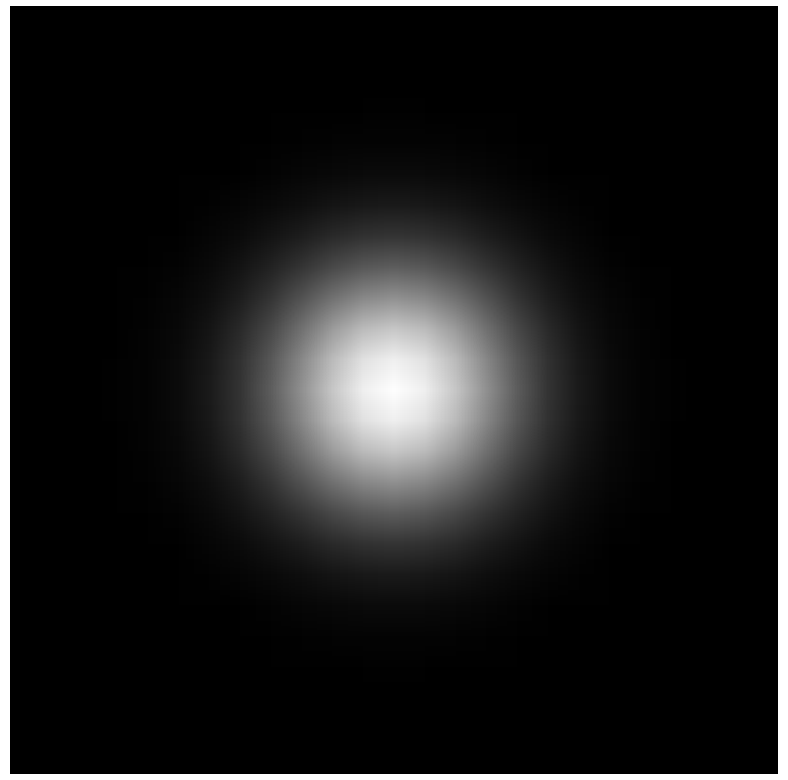
- What’s wrong with this picture?
- What’s the solution?



Source: D. Forsyth

Smoothing with box filter revisited

- What's wrong with this picture?
- What's the solution?
- To eliminate edge effects, weight contribution of neighborhood pixels according to their closeness to the center

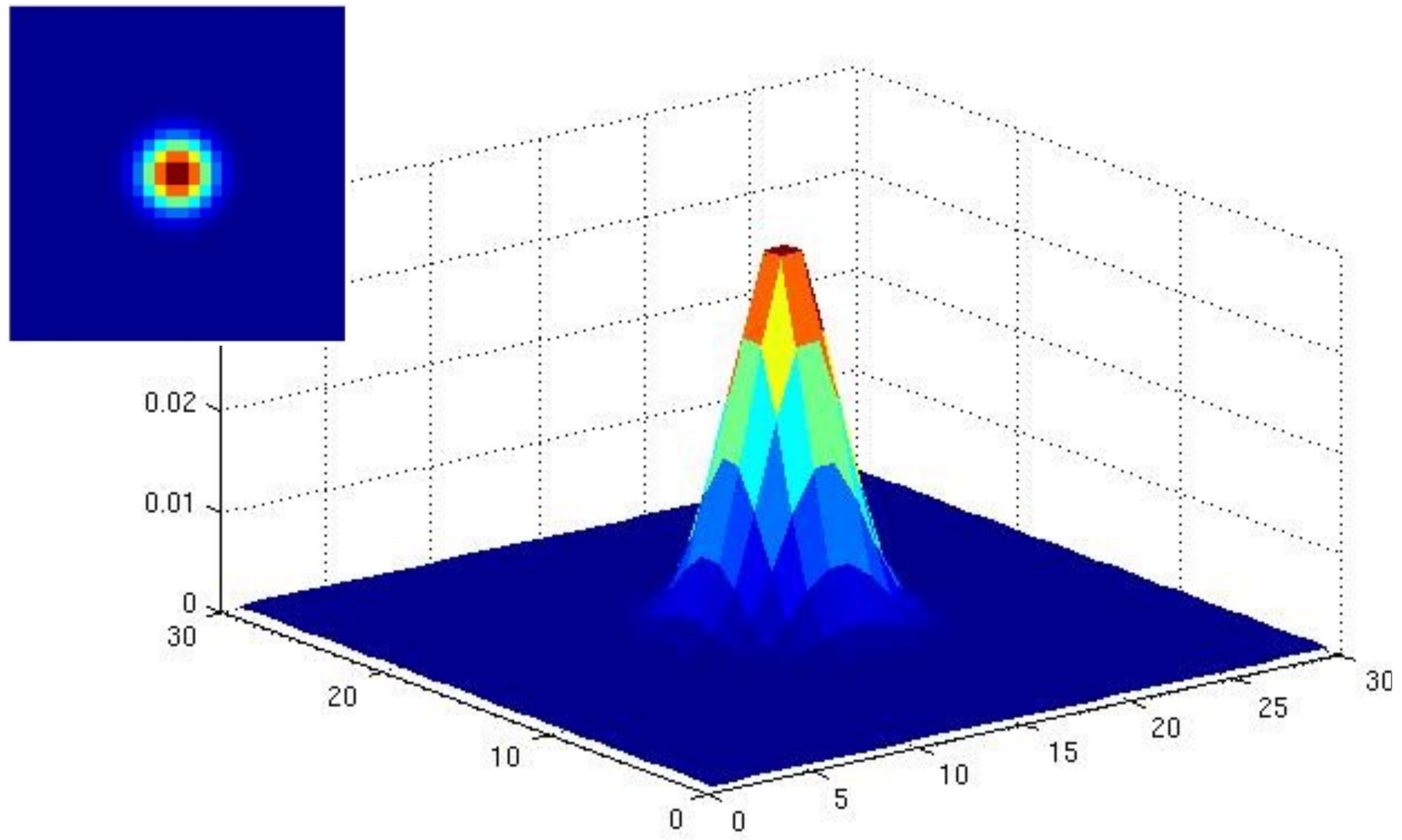


"fuzzy blob"

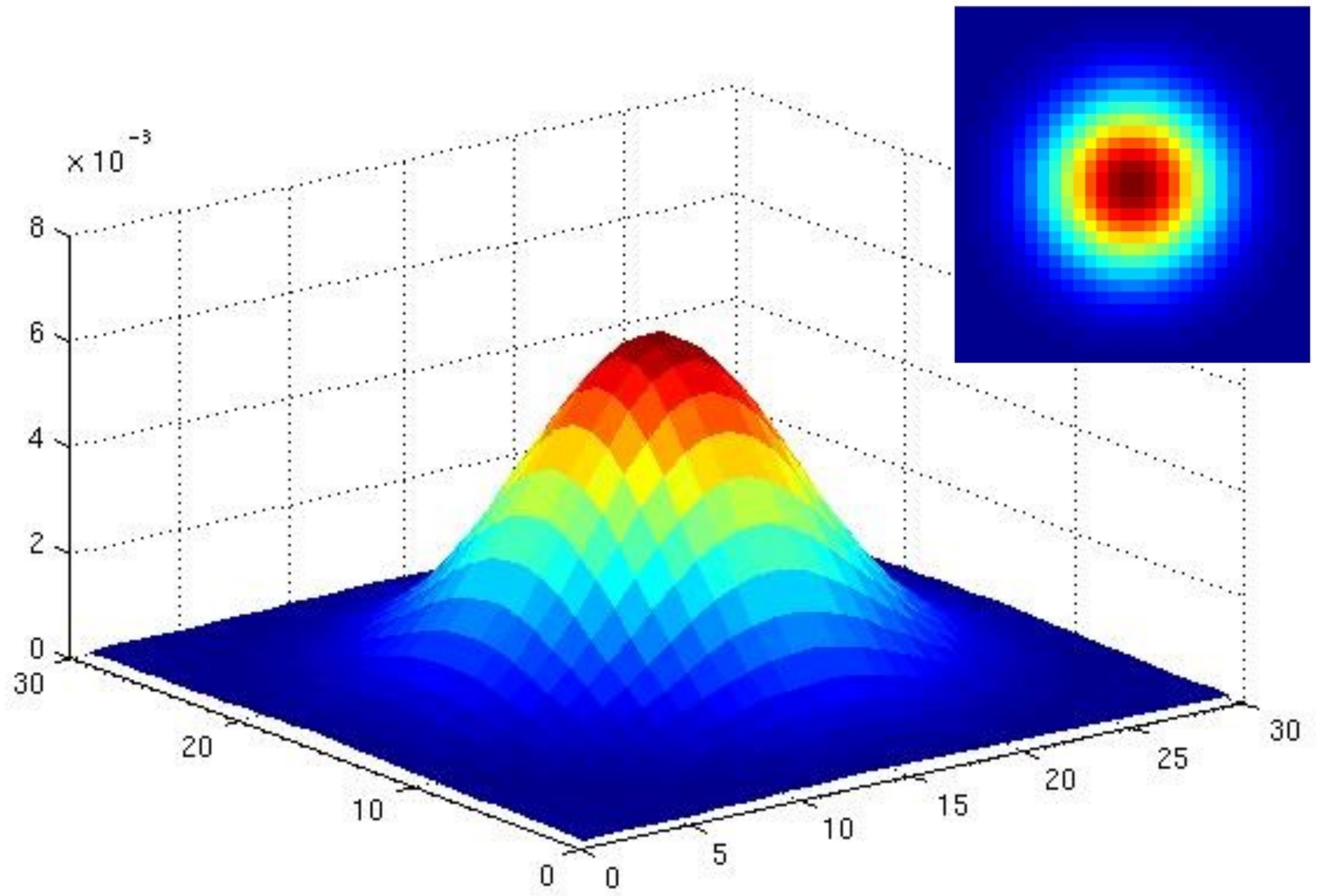
Source: S. Lazebnik

Gaussian kernel

$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$



σ = 2 with 30 x 30 kernel

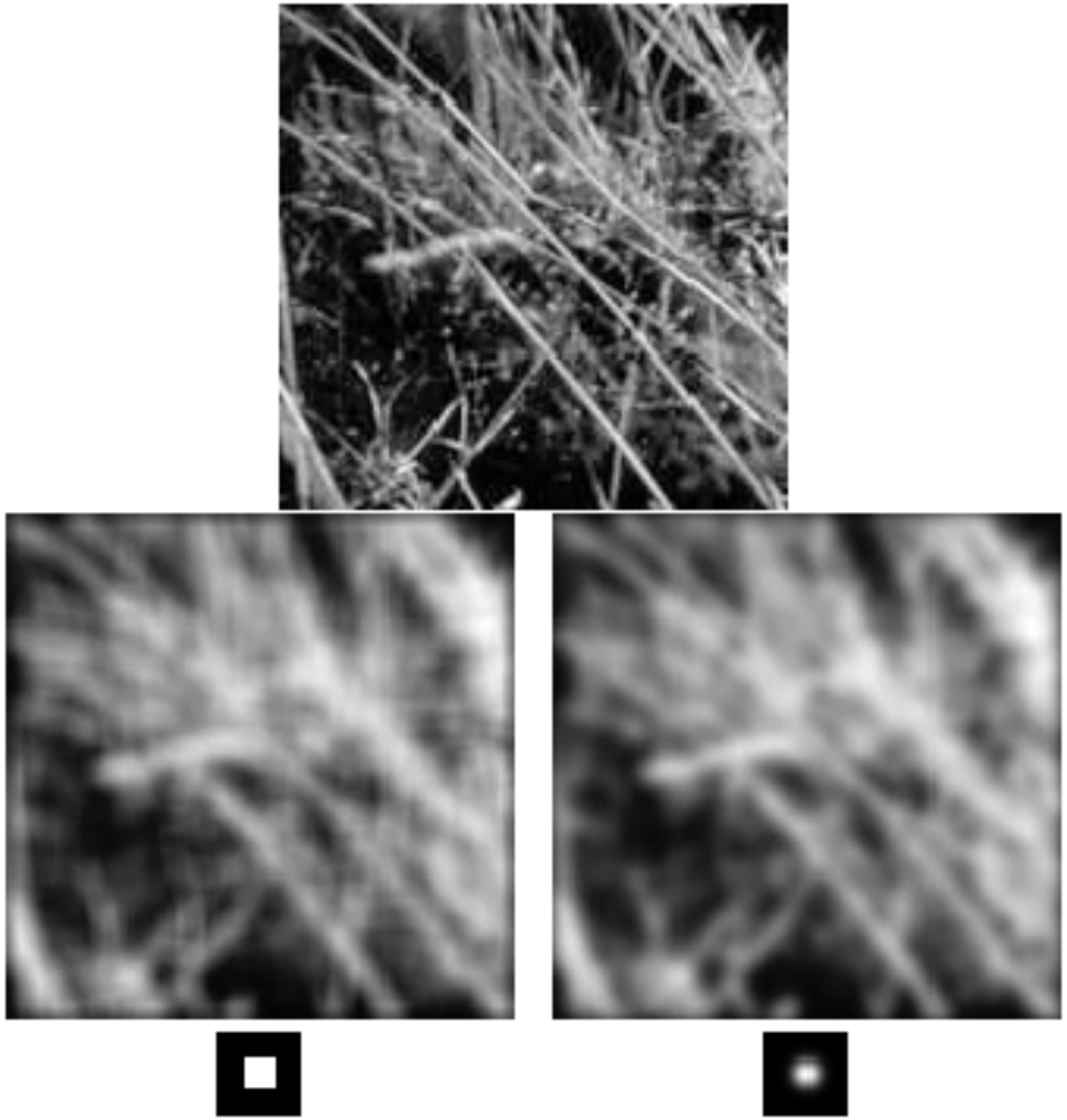


σ = 5 with 30 x 30 kernel

- Constant factor in front makes kernel sum to 1 (can also omit it and just divide by sum of filter weights).

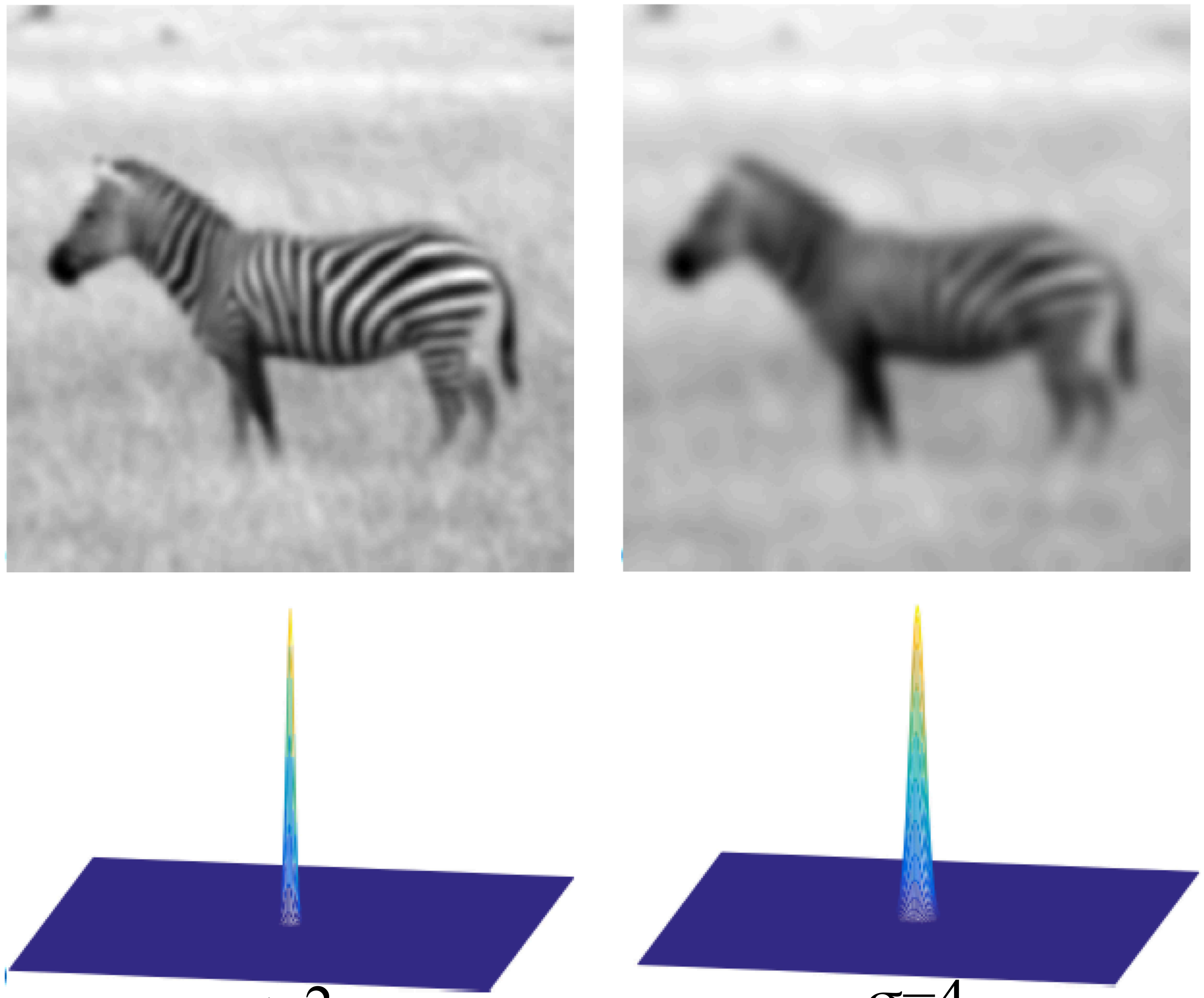
Source: K. Grauman

Gaussian vs. box filtering



Source: S. Lazebnik

Gaussian standard deviation



σ=2 σ=4 σ=8

Source: Torralba, Freeman, Isola

Gaussian filters

- Convolution with self is another Gaussian
  - Can smooth with small-σ kernel, repeat, get same result as larger-σ
  - Convolving two times with Gaussian kernel with std. dev. σ is same as convolving once with kernel with std. dev. $\sigma\sqrt{2}$

I

blur(blur(I))

blur(blur(blur(I)))

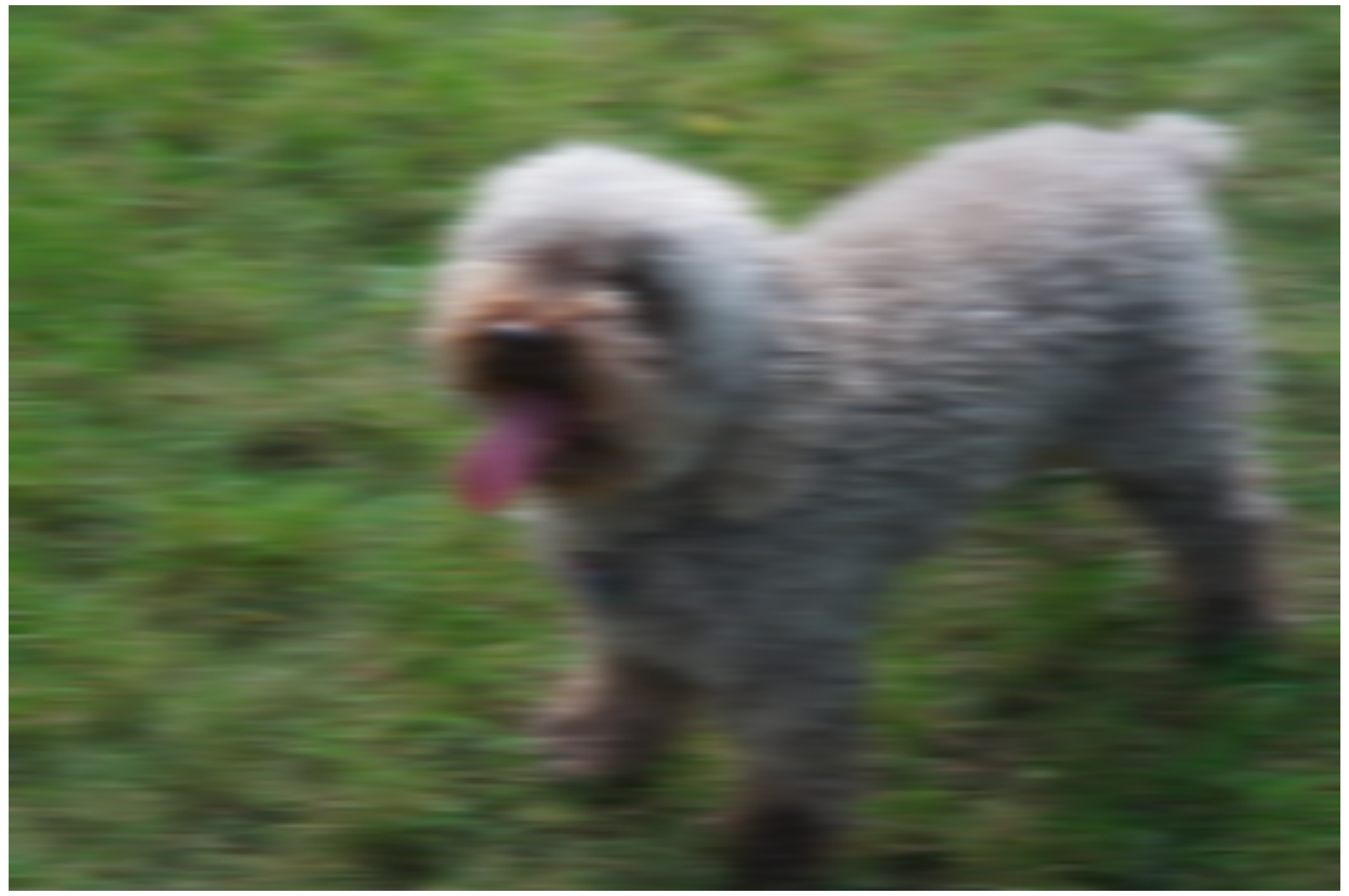
blur(blur(blur(blur(I))))

Source: K. Grauman

Gaussian filters

- It's a **separable kernel**
  - Blur with 1D Gaussian in one direction, then the other.
  - Faster to compute. O(n) time for an n*n kernel instead of O(n^2)
  - Learn more about this in Problem Set 1!

I



$\mathrm{blur}_x(I)$

$\mathrm{blur}_y(\mathrm{blur}_x(I))$

Edges: recall last lecture…

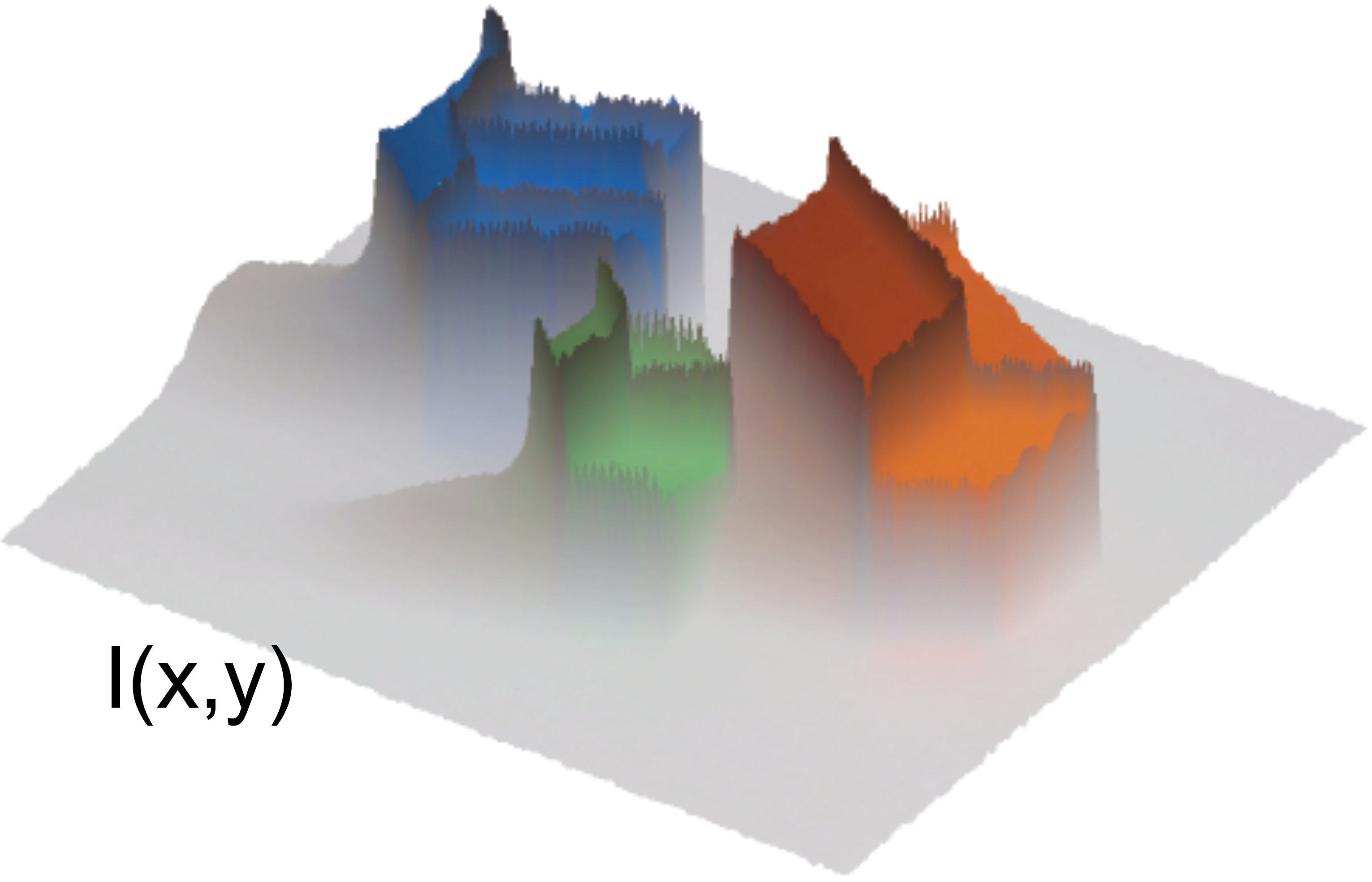


Image gradient:

$$\nabla \mathbf{I} = \left(\frac{\partial \mathbf{I}}{\partial x}, \frac{\partial \mathbf{I}}{\partial y} \right)$$

Approximation image derivative:

$$\frac{\partial \mathbf{I}}{\partial x} \simeq \mathbf{I}(x, y) - \mathbf{I}(x-1, y)$$

Edge strength $E(x, y) = |\nabla \mathbf{I}(x, y)|$

Edge orientation: $\theta(x, y) = \angle \nabla \mathbf{I} = \arctan \frac{\partial \mathbf{I} / \partial y}{\partial \mathbf{I} / \partial x}$

Edge normal: $\mathbf{n} = \frac{\nabla \mathbf{I}}{|\nabla \mathbf{I}|}$

Slide credit: Antonio Torralba

Discrete derivatives

$$d_0 = [1, -1]$$

$$f \circ d_0 = f[n] - f[n-1]$$

$$d_1 = [1, 0, -1]/2$$

$$f \circ d_1 = \frac{f[n+1] - f[n-1]}{2}$$

Source: Torralba, Freeman, Isola

[-1 1]

g[m,n]

○ [-1, 1]

h[m,n]

=

f[m,n]

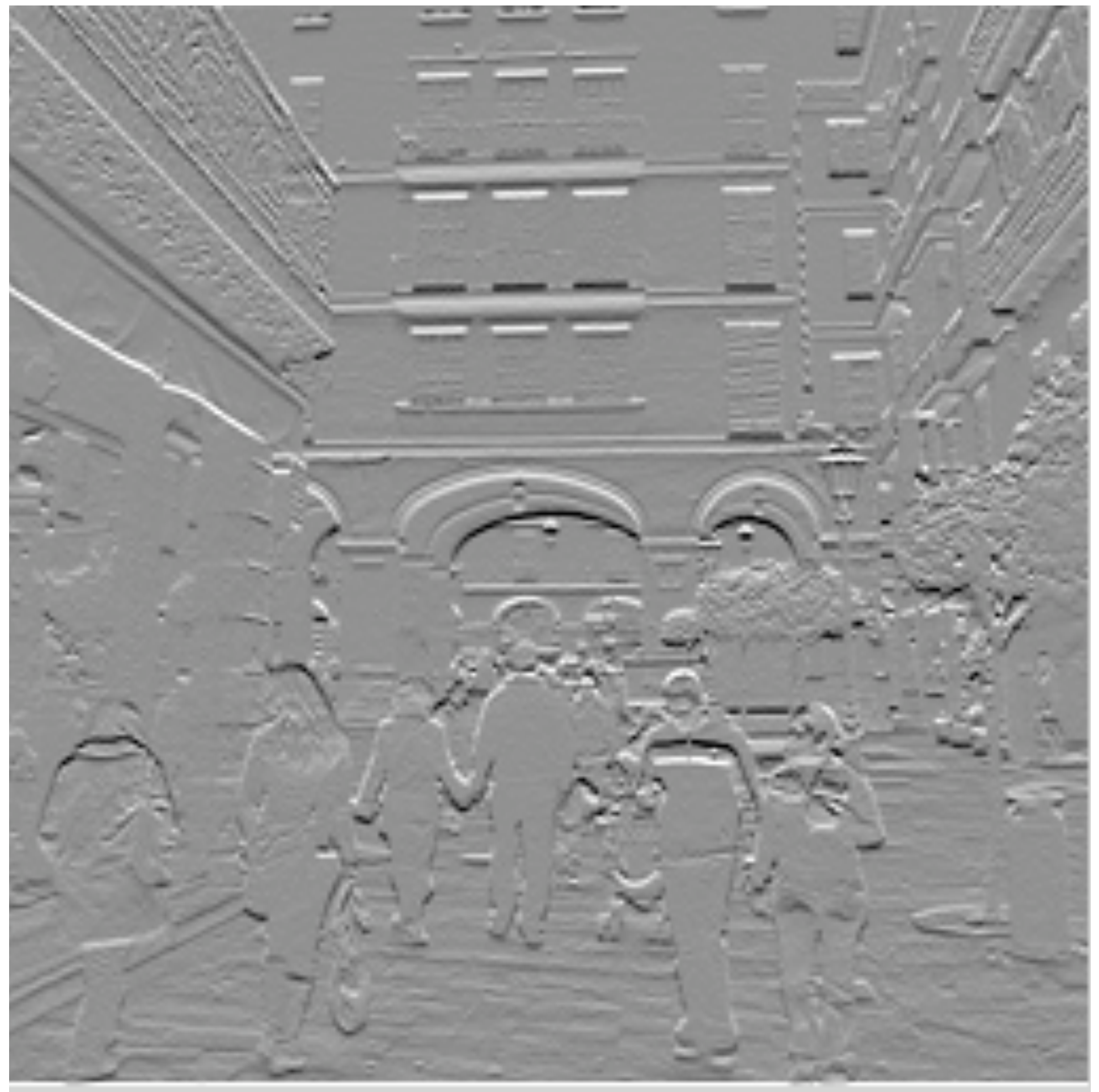
Source: Torralba, Freeman, Isola

$[-1\ 1]^T$

g[m,n]

$\circ$ $[-1, 1]^T$ =

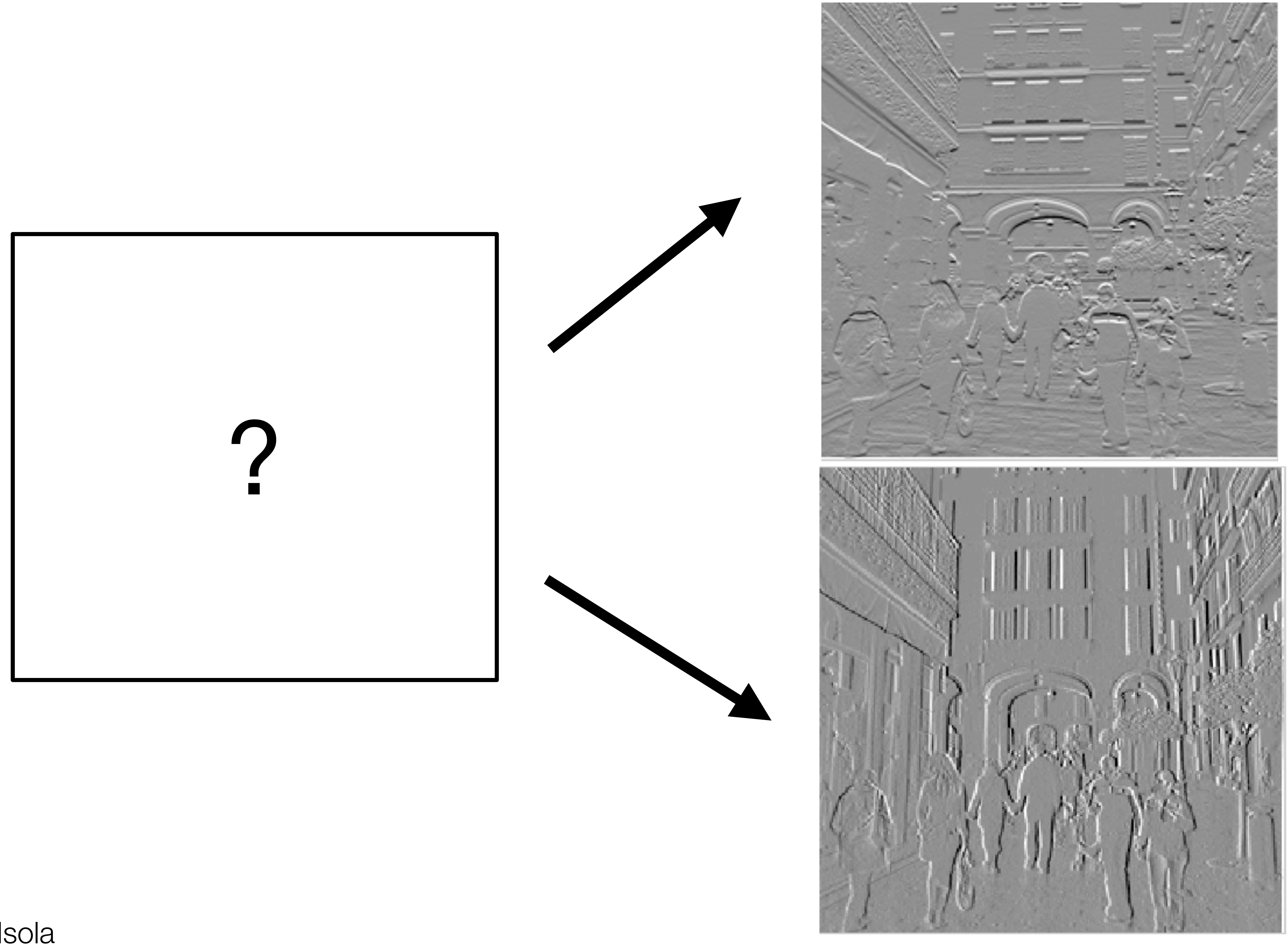
h[m,n]



f[m,n]

Source: Torralba, Freeman, Isola

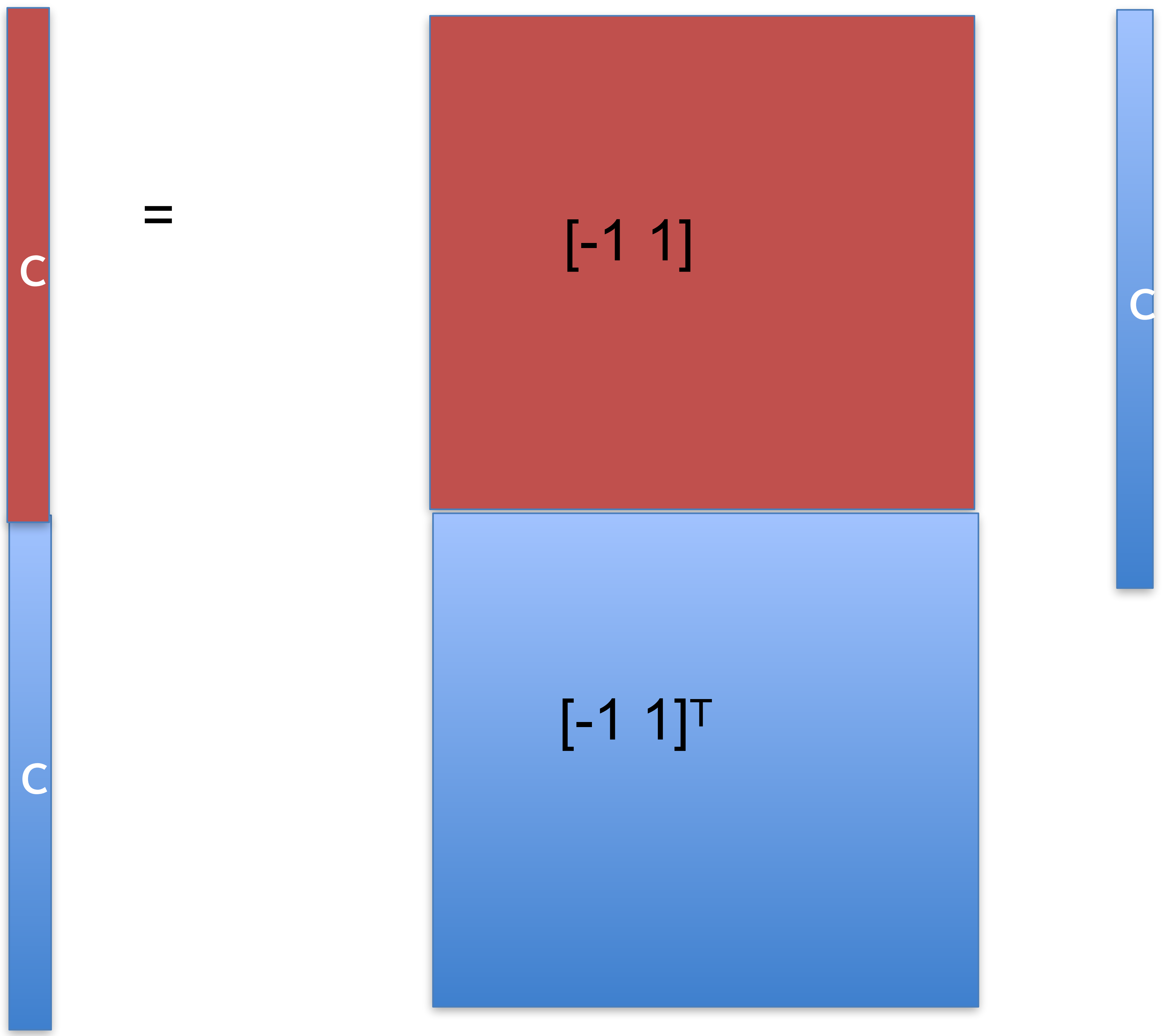
Can we recover the image?



Source: Torralba, Freeman, Isola

Reconstruction from 2D derivatives

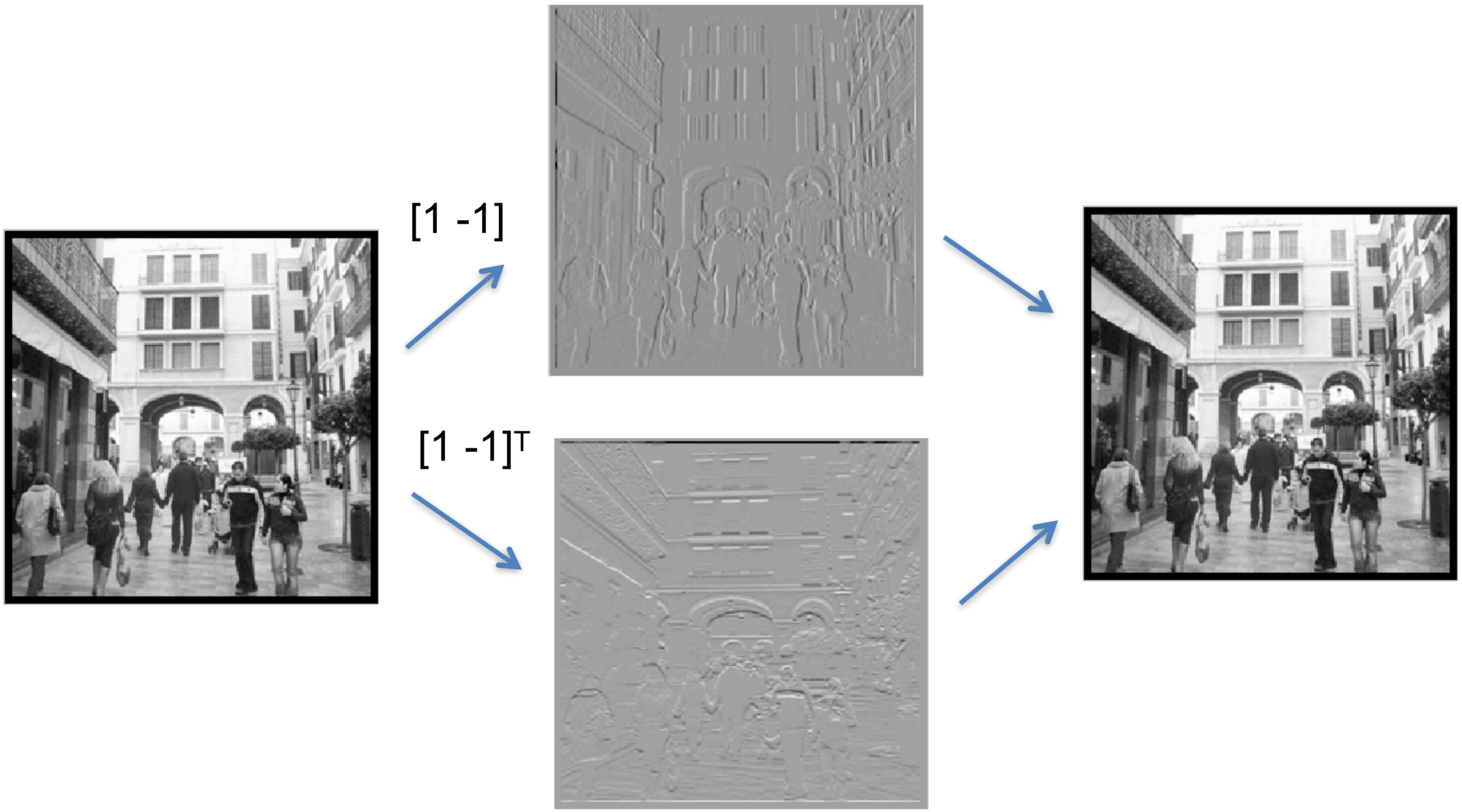
In 2D, we have multiple derivatives (along *n* and *m*)



and we compute the pseudo-inverse of the full matrix.

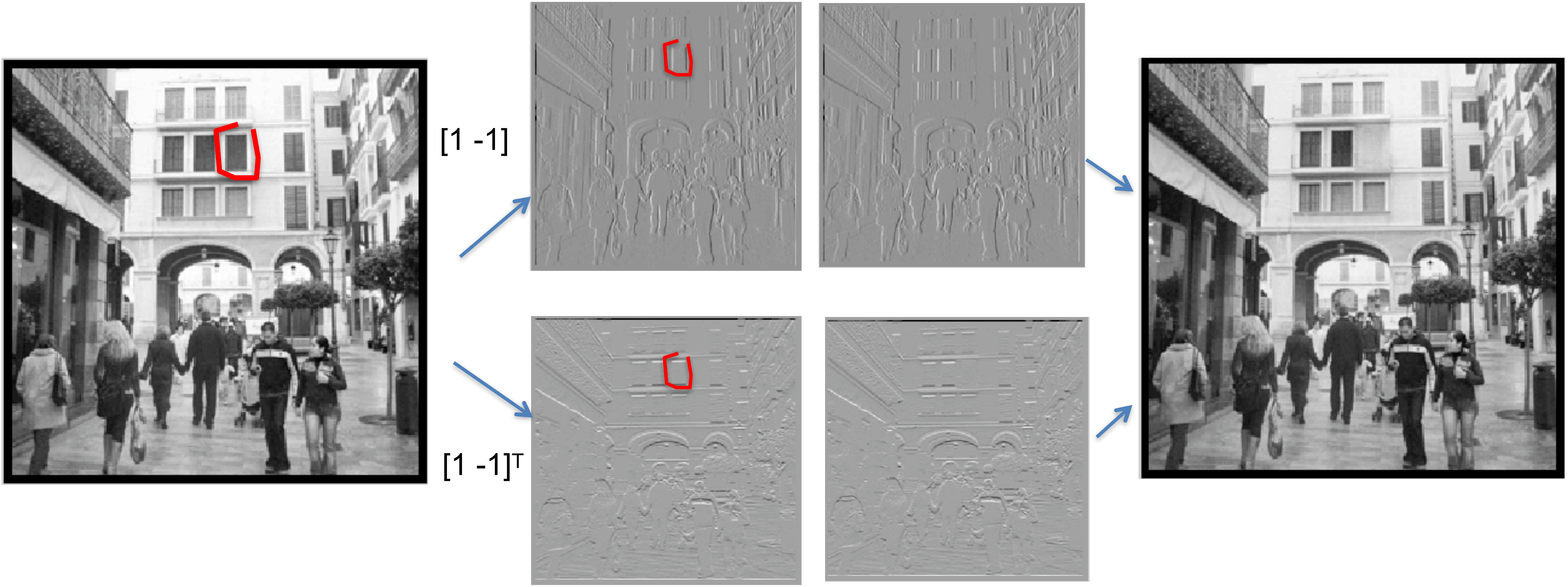
Source: Torralba, Freeman, Isola

Reconstruction from 2D derivatives



Source: Torralba, Freeman, Isola

Editing the edge image



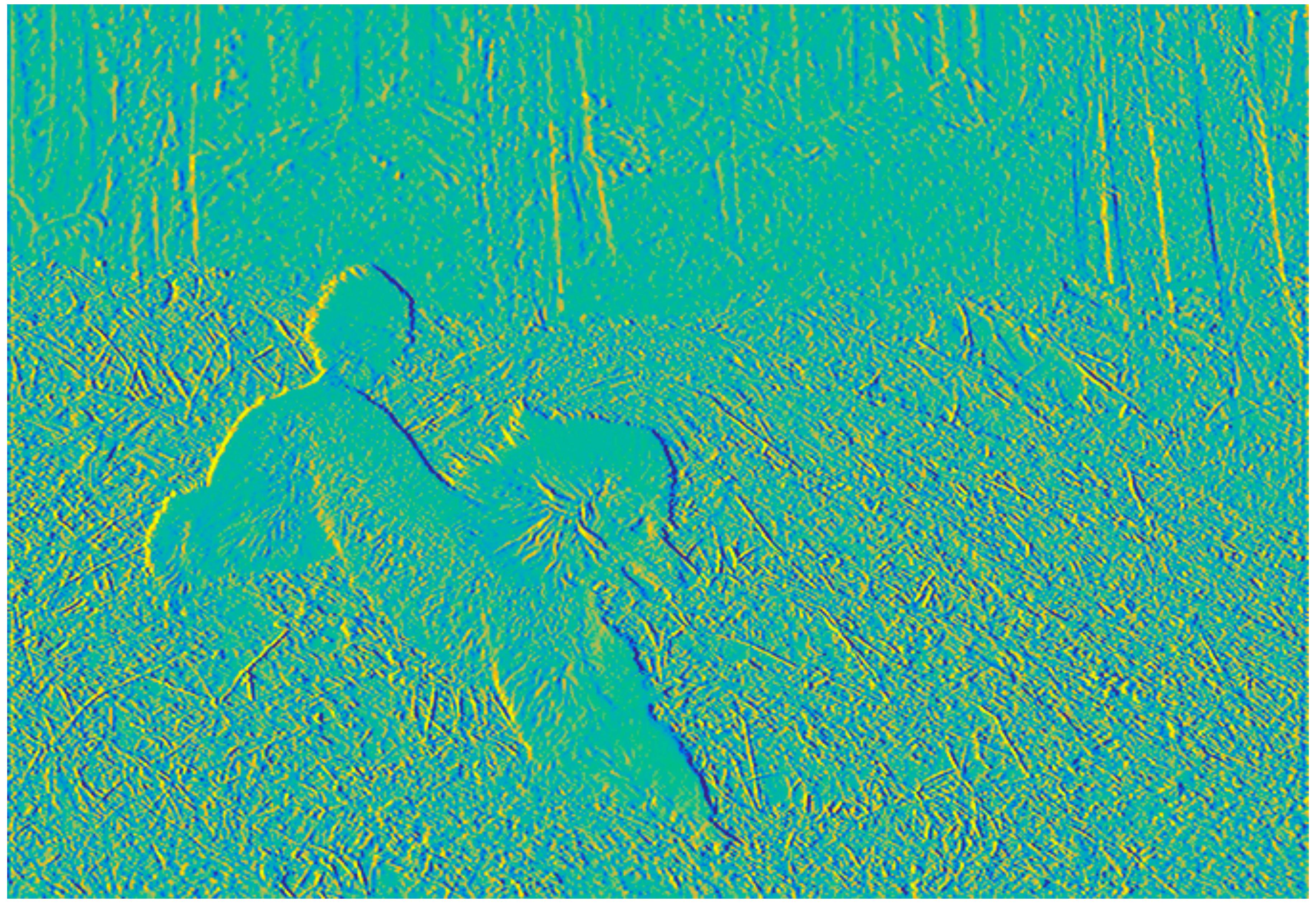
Source: Torralba, Freeman, Isola

Thresholding edges

Source: Torralba, Freeman, Isola

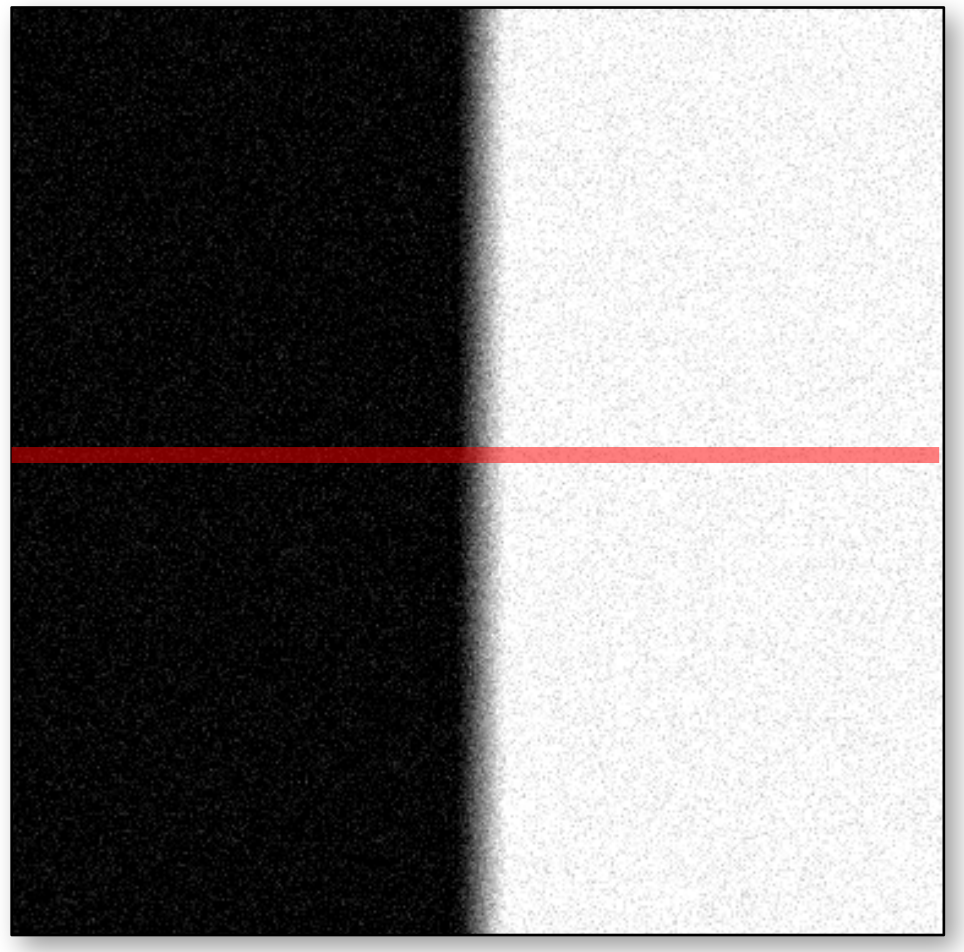
Issues with derivative filters

[1 -1]



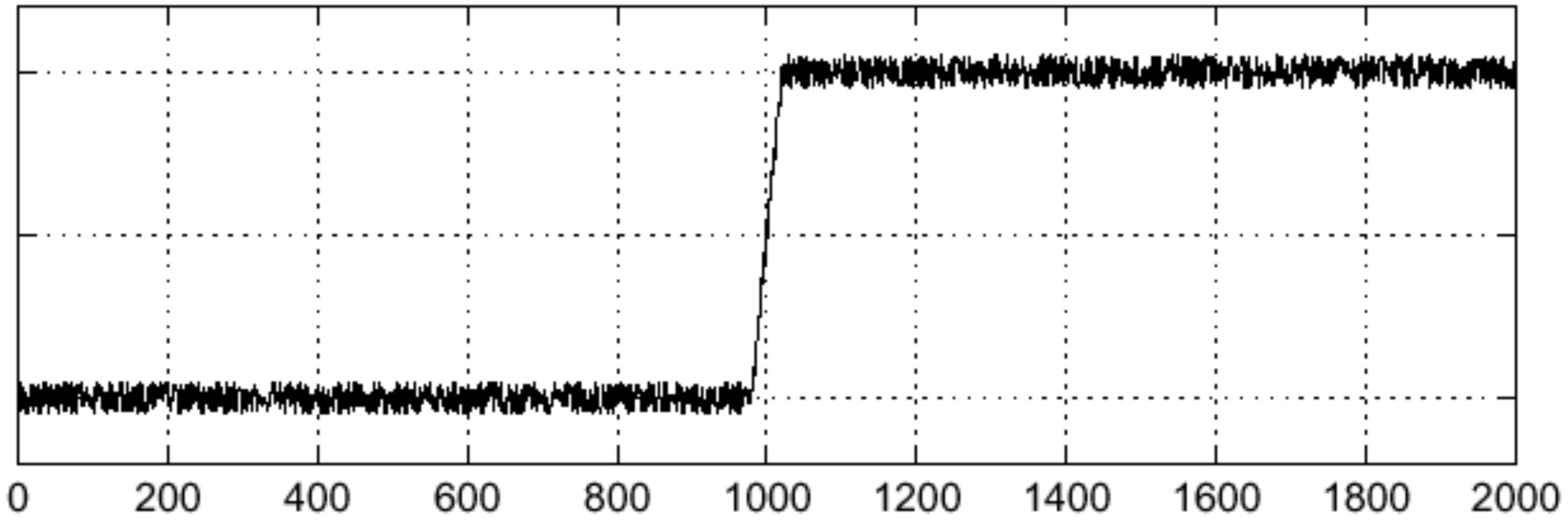
- Sensitive to edges at small spatial scales
- Also sensitive to noise
- You'll see this in Problem Set 1

Why is this happening?

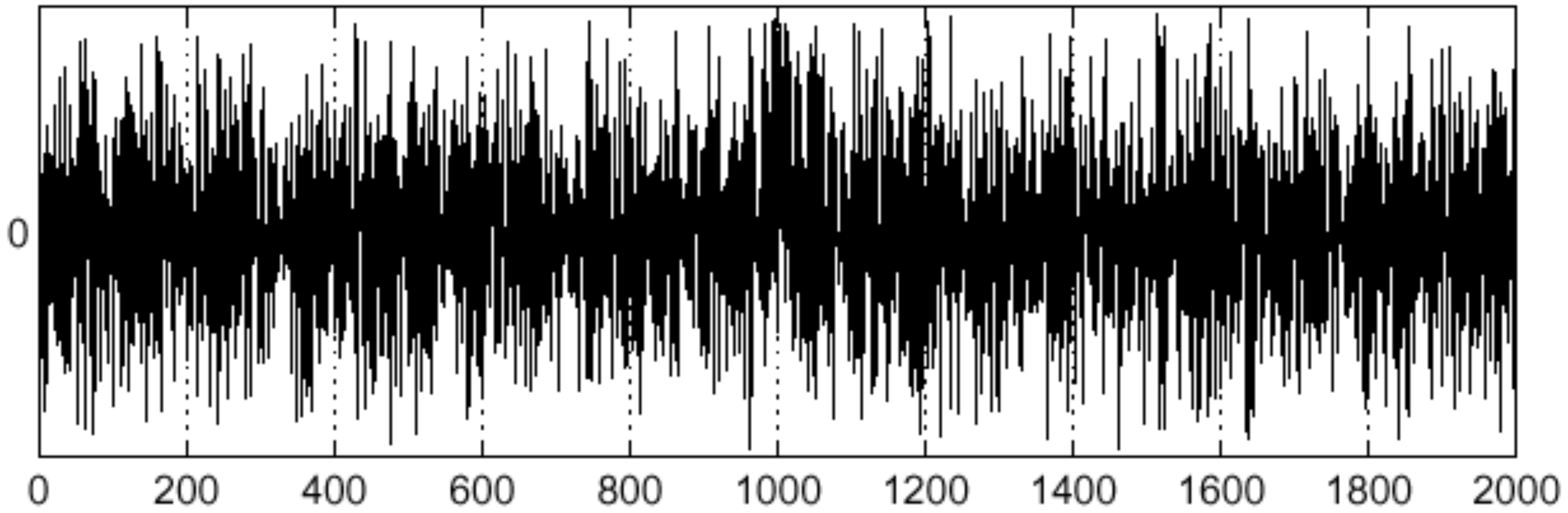


Noisy input image

$f(x)$



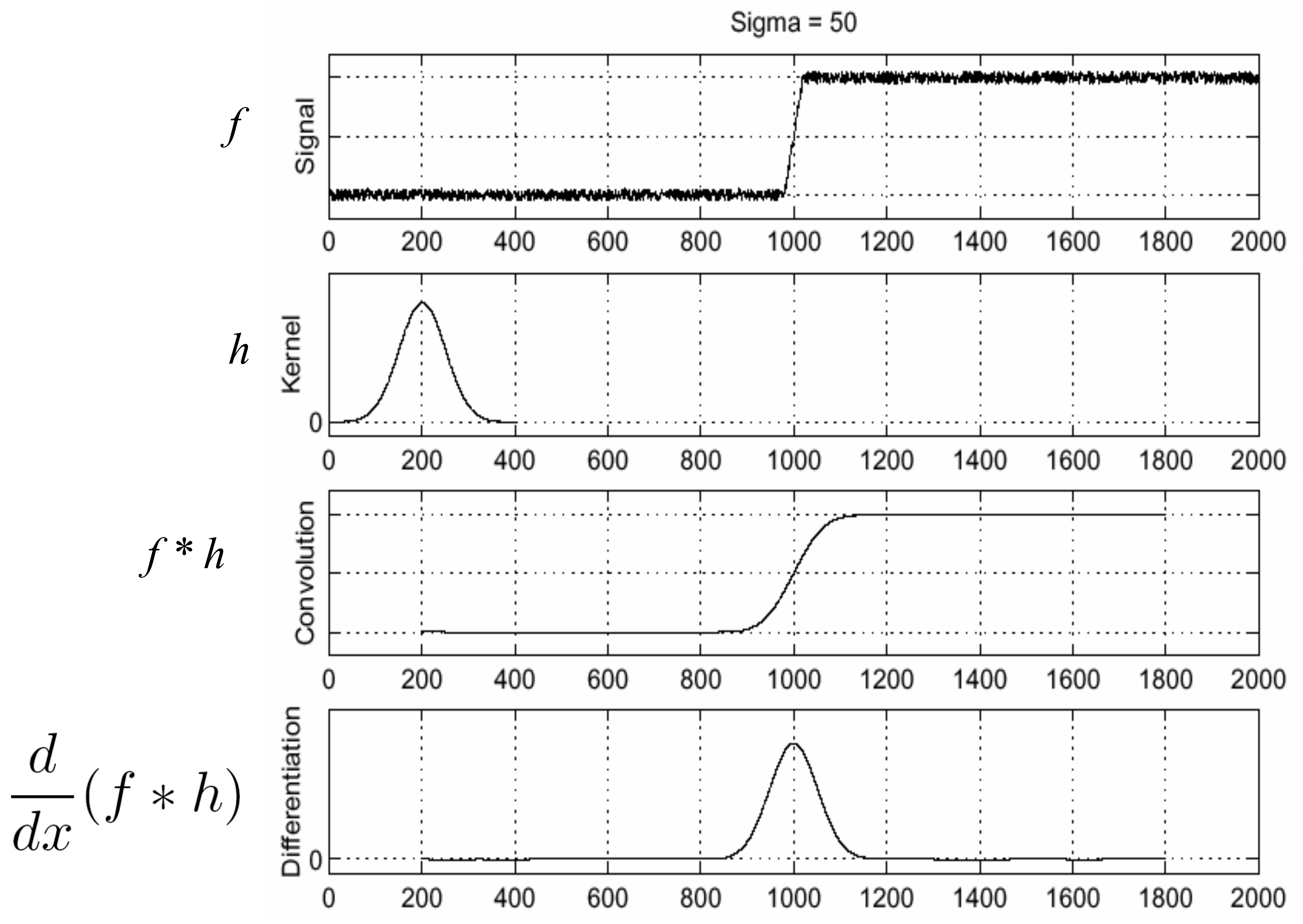
$\frac{d}{dx}f(x)$



Where is the edge?

Source: S. Seitz

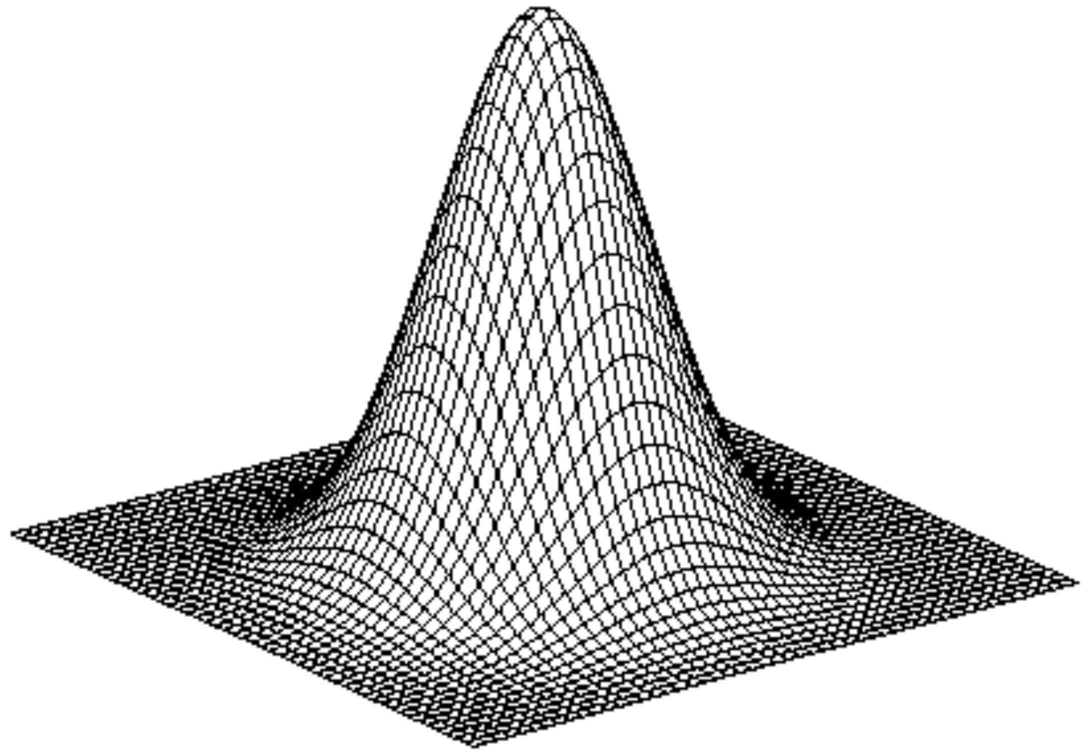
Solution: smooth first



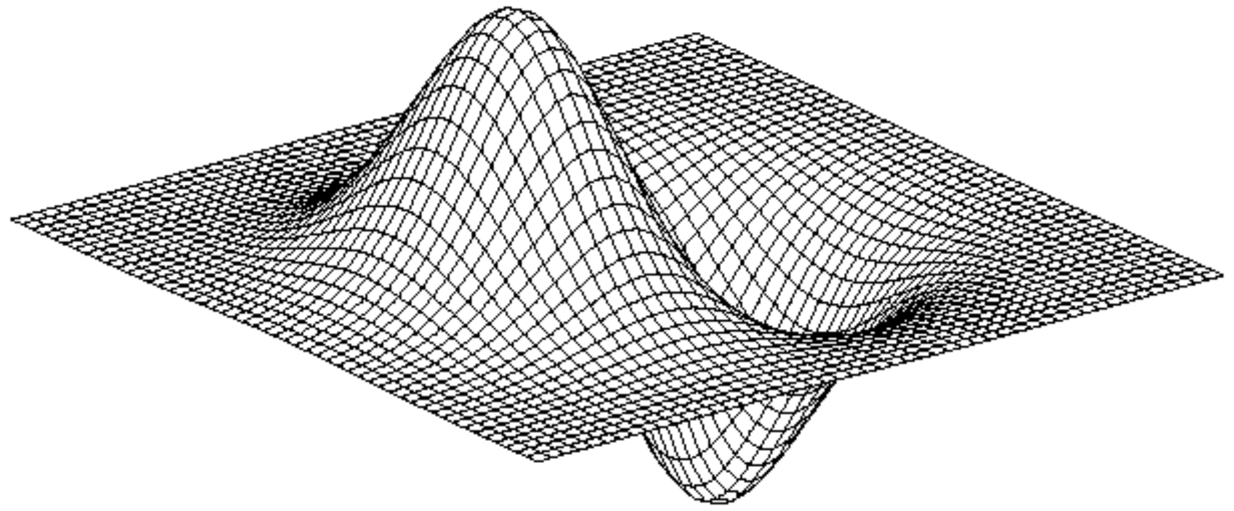
To find edges, look for peaks in $\frac{d}{dx}(f * h)$

Source: S. Seitz

Derivative of Gaussian filter



Gaussian



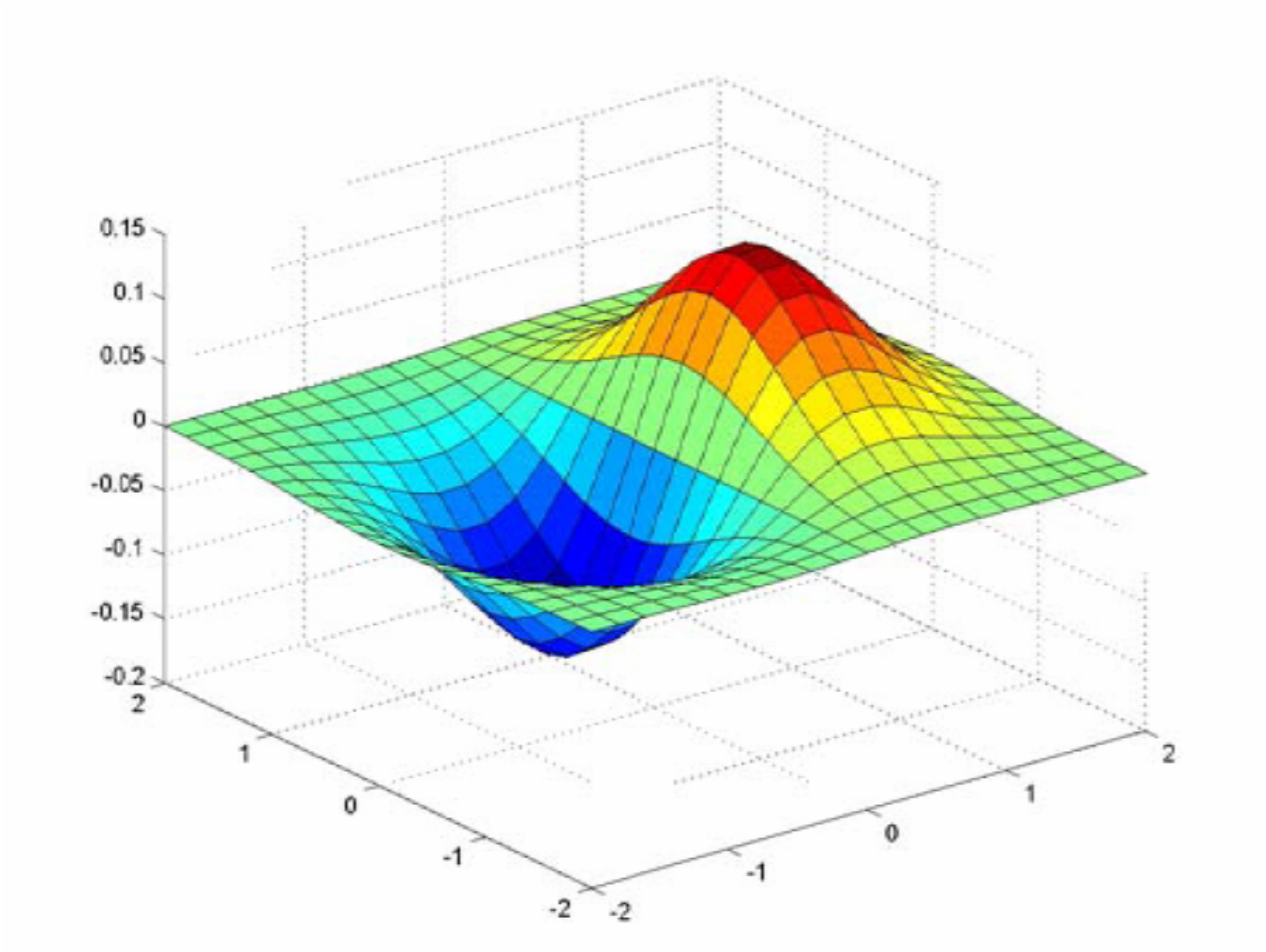
Derivative of Gaussian (x)

$$h_\sigma(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$

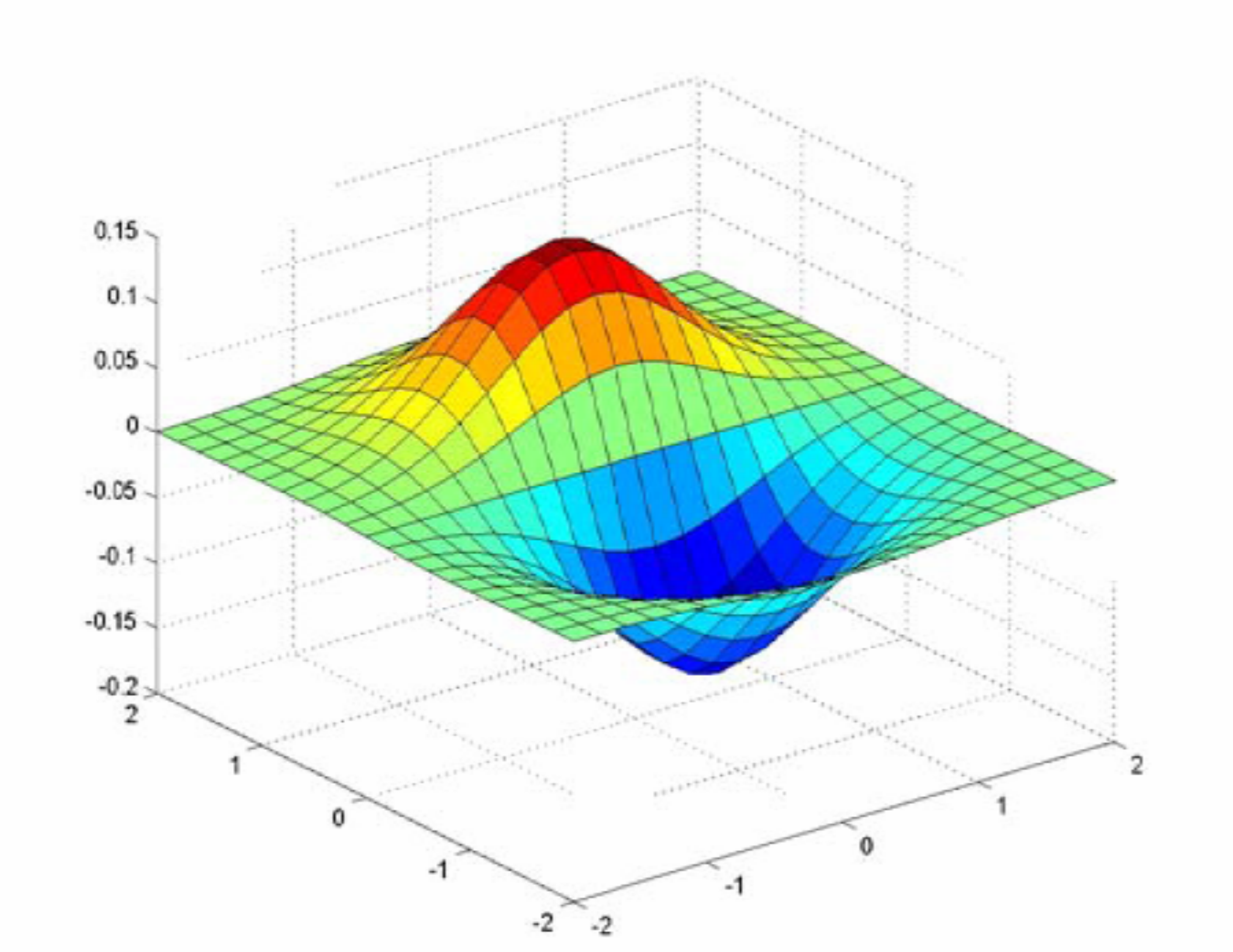
$$\frac{\partial}{\partial x} h_\sigma(u,v)$$

Source: N. Snavely

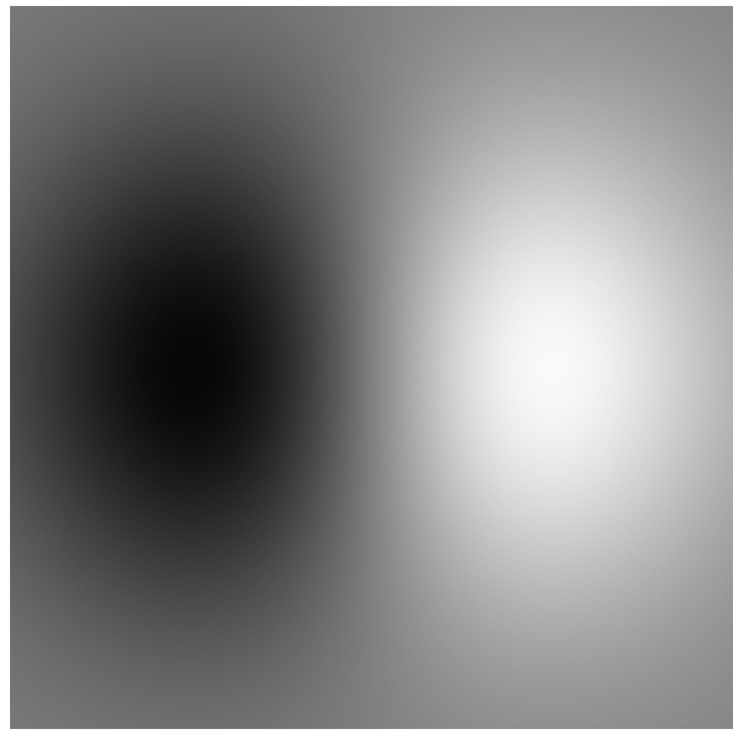
Derivative of Gaussian filter

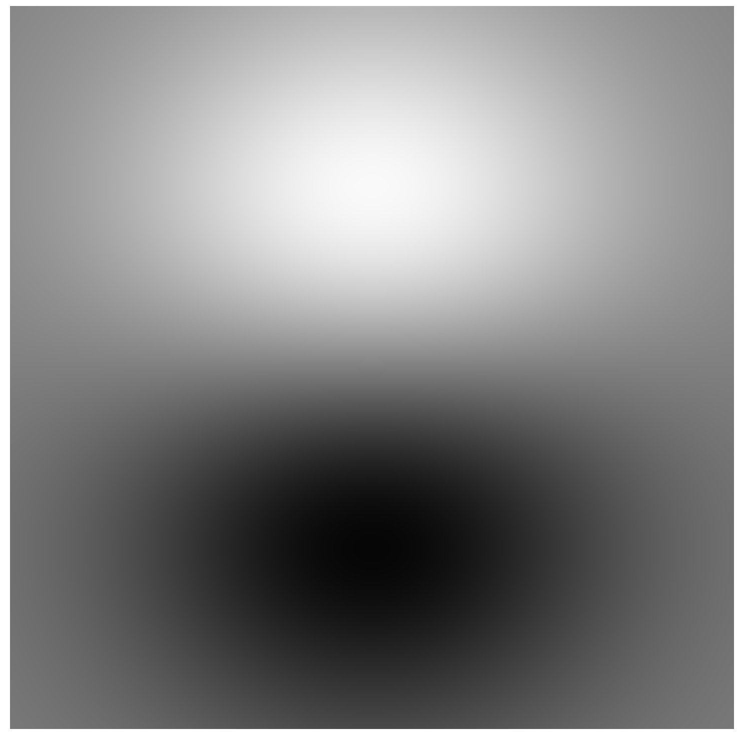


x-direction



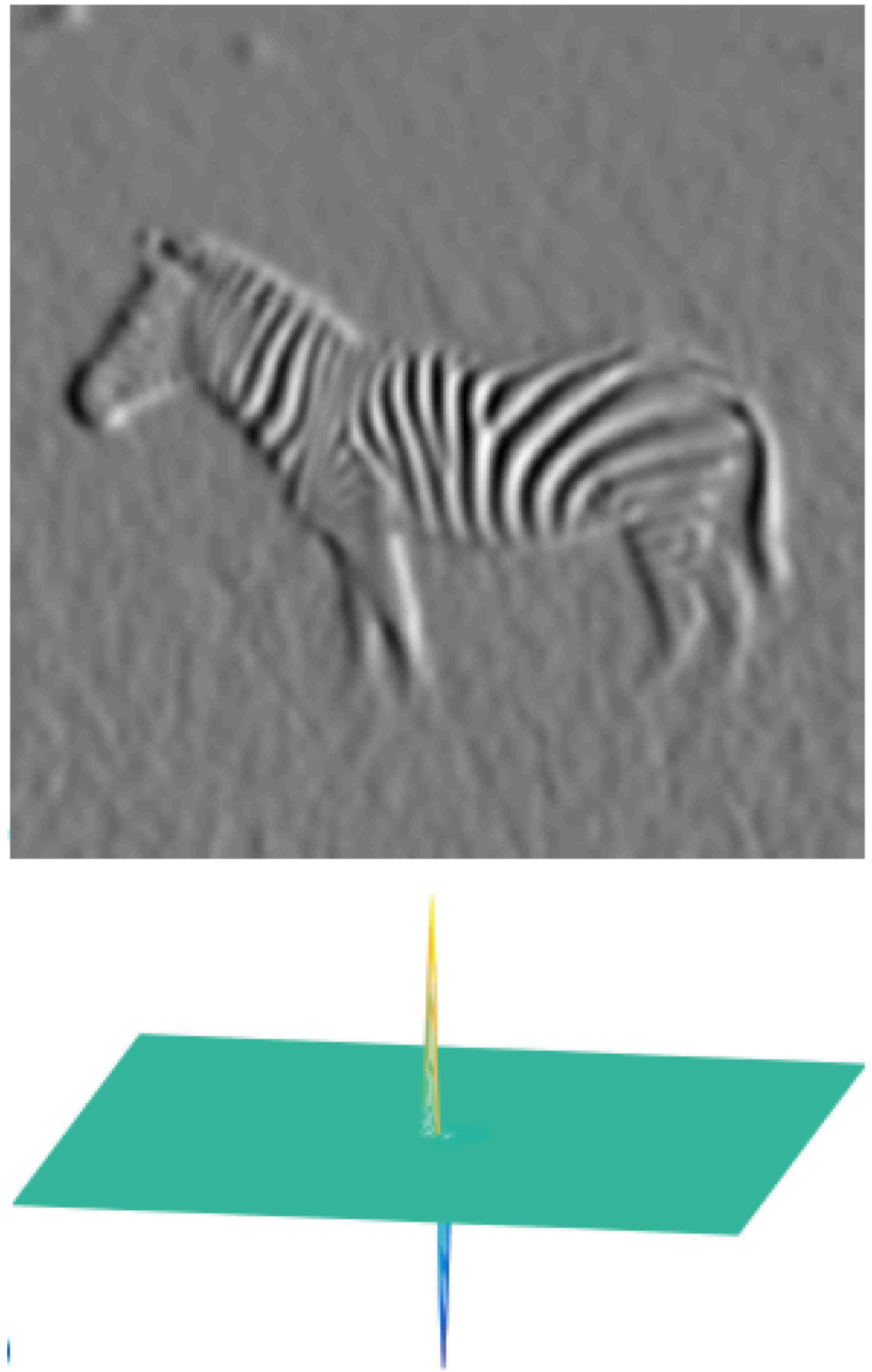
y-direction



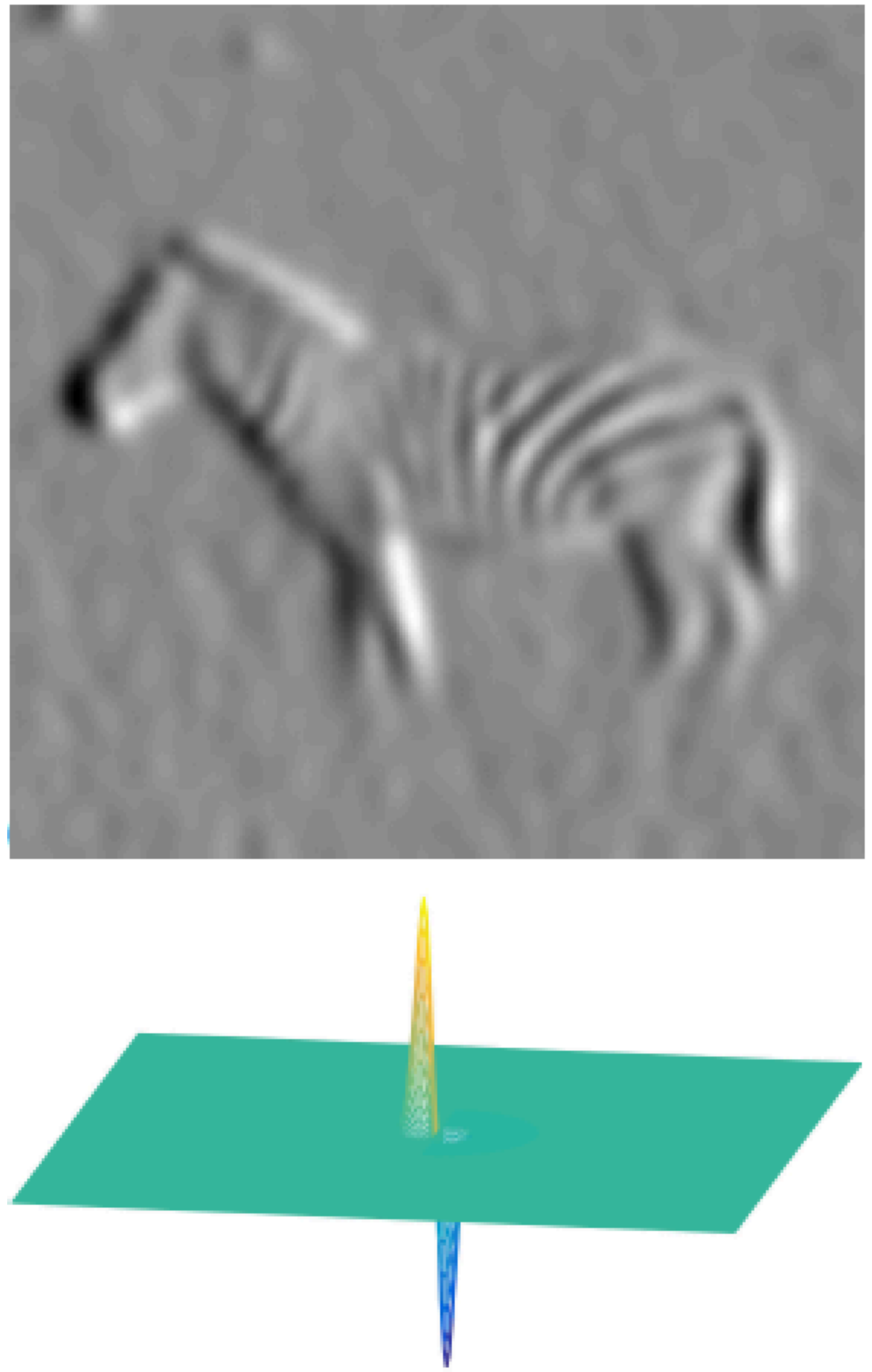


Source: N. Snavely

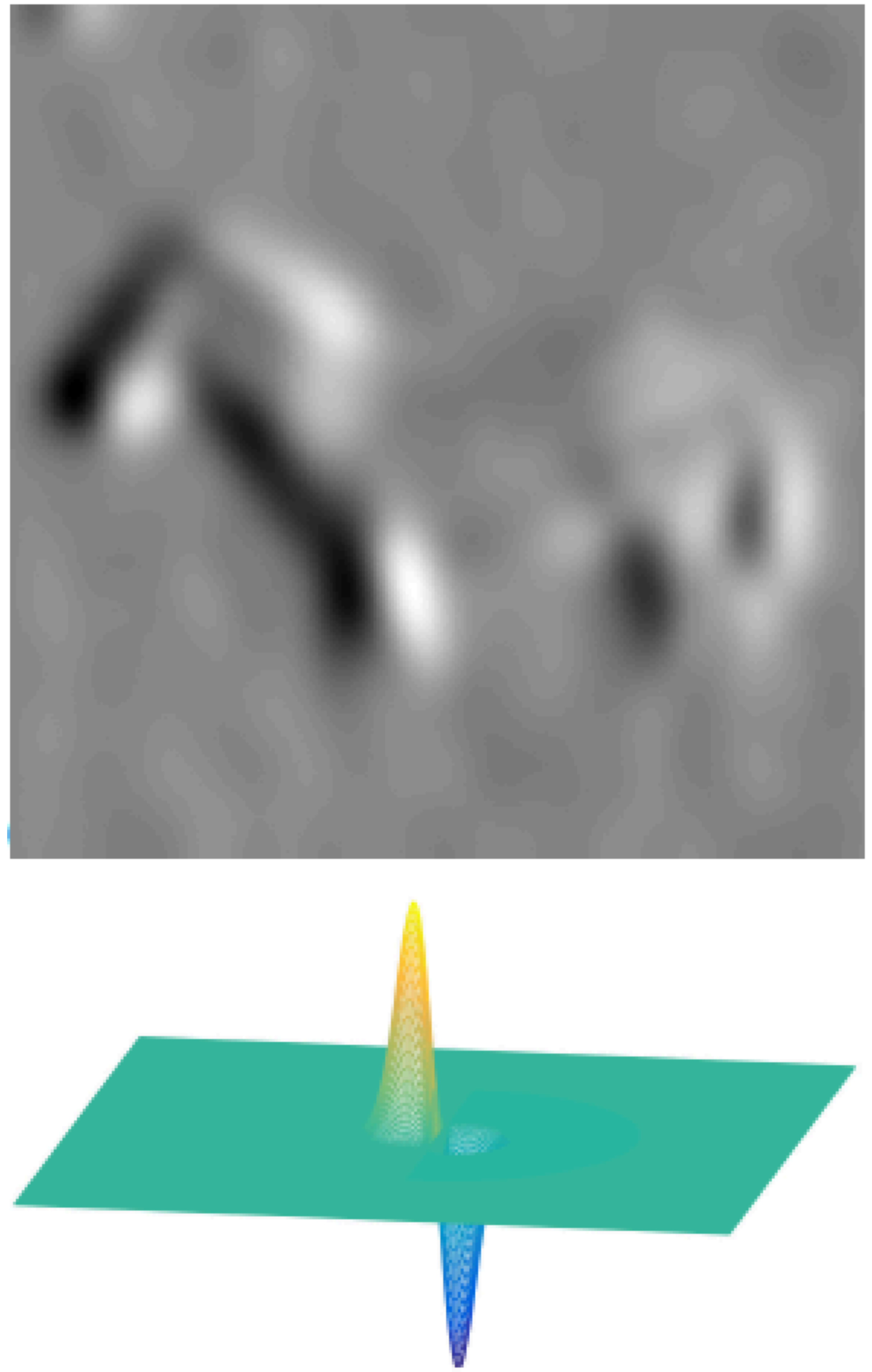
Derivatives of Gaussians: Scale



σ=2



σ=4

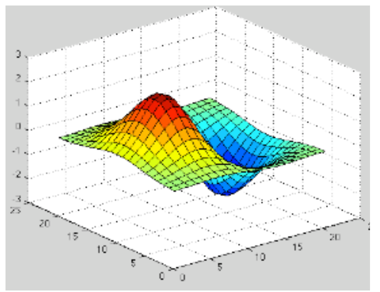


σ=8

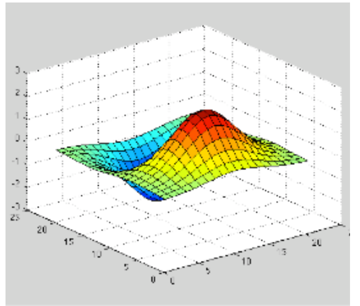
Source: Torralba, Freeman, Isola

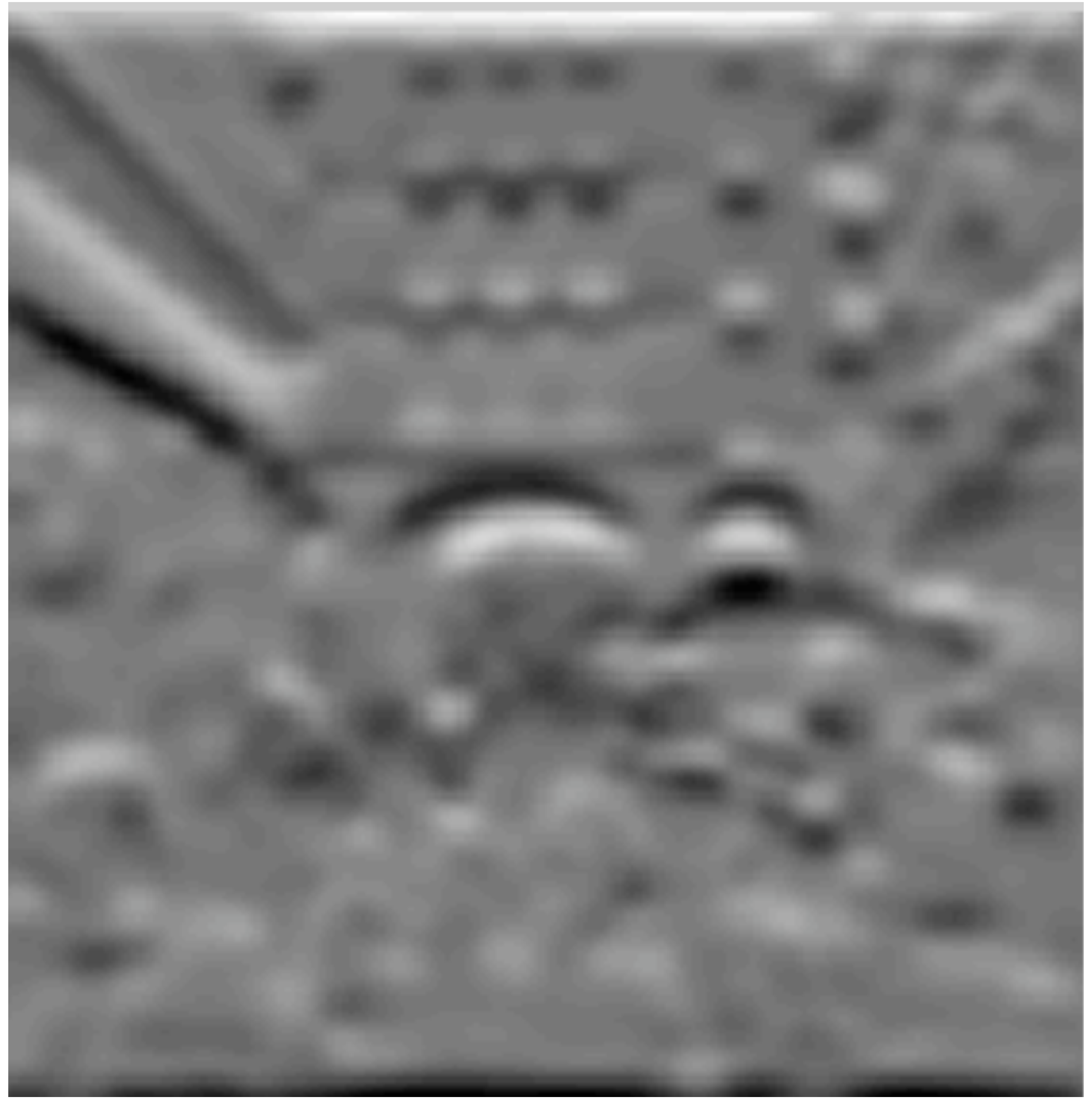
Picks up larger-scale edges

$$g_x(x,y) = \frac{\partial g(x,y)}{\partial x} = \frac{-x}{2\pi\sigma^4} e^{-\frac{x^2+y^2}{2\sigma^2}}$$



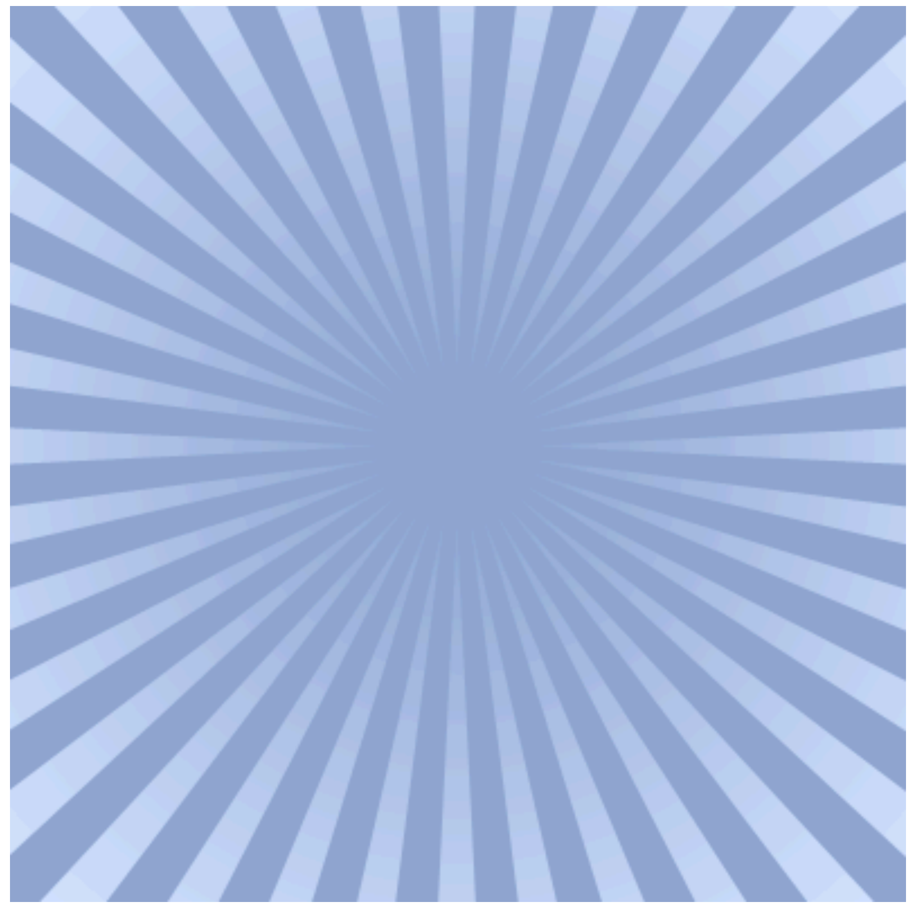
$$g_y(x,y) = \frac{\partial g(x,y)}{\partial y} = \frac{-y}{2\pi\sigma^4} e^{-\frac{x^2+y^2}{2\sigma^2}}$$



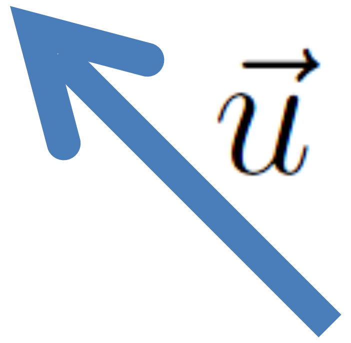


Source: Torralba, Freeman, Isola

Computing a directional derivative



f



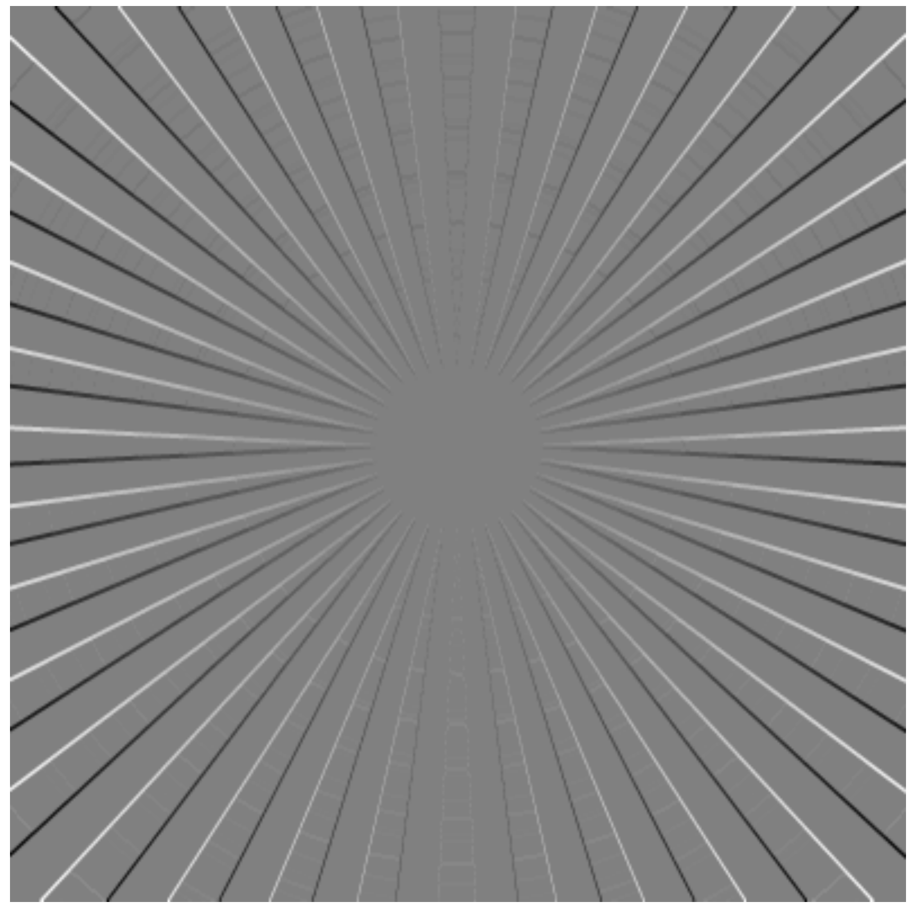
$\vec{u}$

(From multivariable calculus)

$$\nabla_{\vec{u}} f(\vec{x}) = \nabla f(\vec{x}) \cdot \vec{u}$$

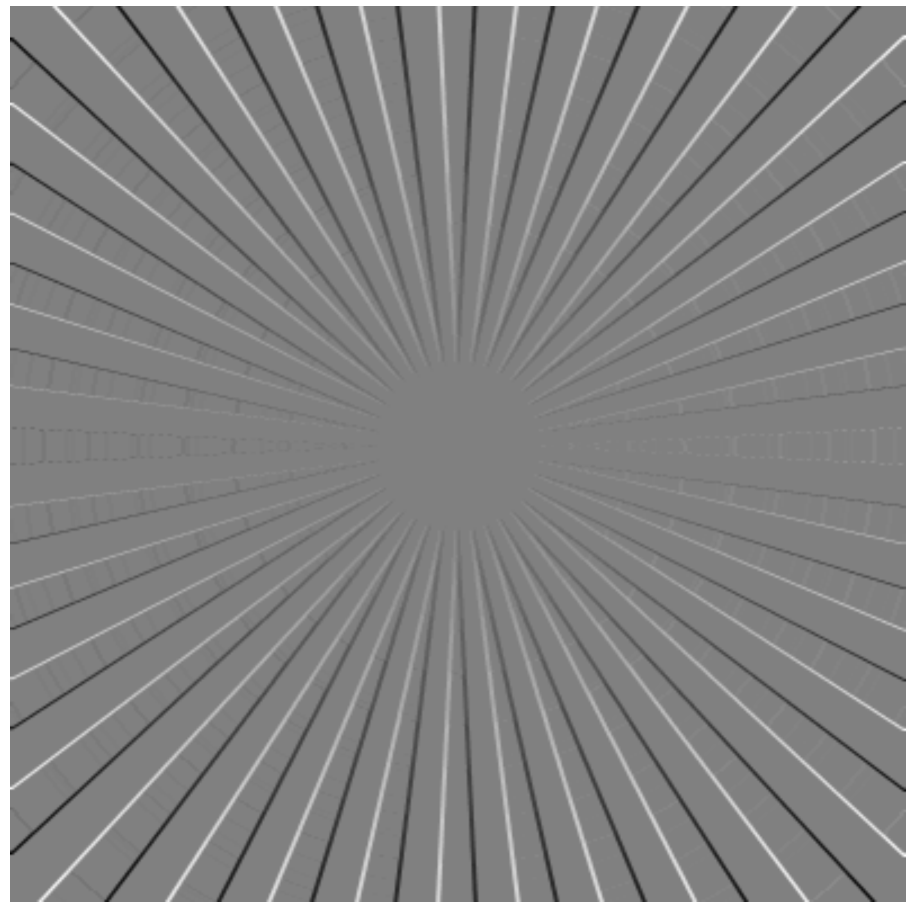
$\nabla_{\vec{u}} f$ = ?

Directional derivative is a linear combination of partial derivatives



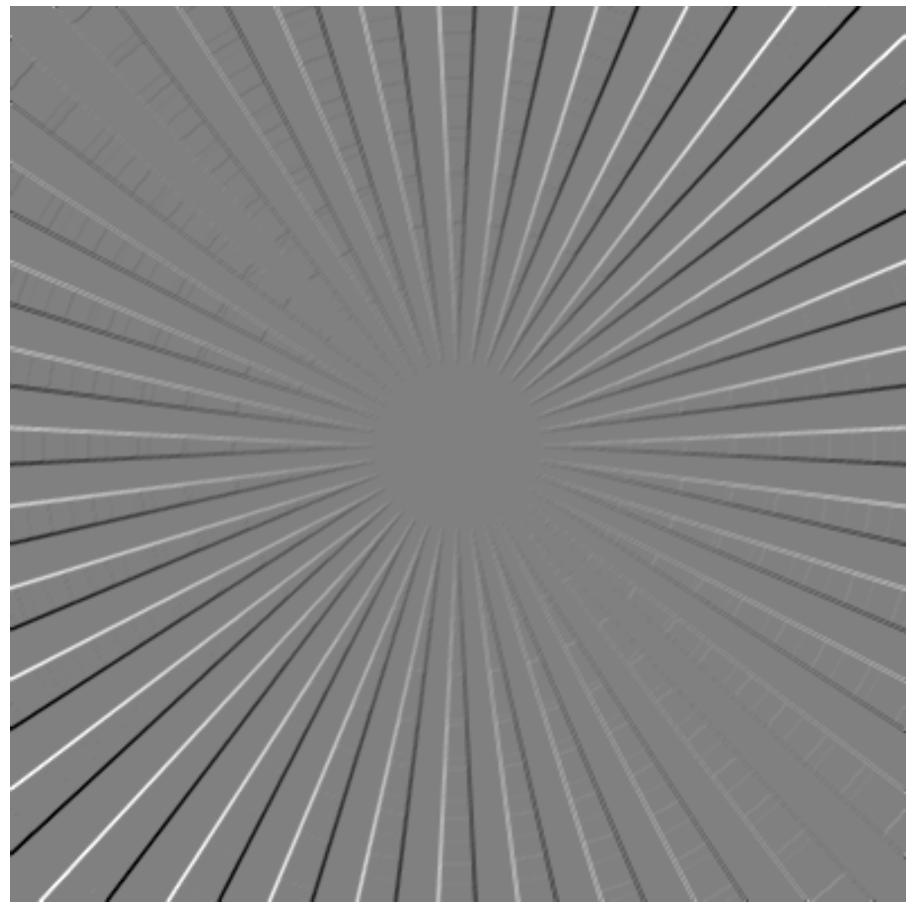
$\frac{\partial f}{\partial x} \cdot u_x$

+



$\frac{\partial f}{\partial y} \cdot u_y$

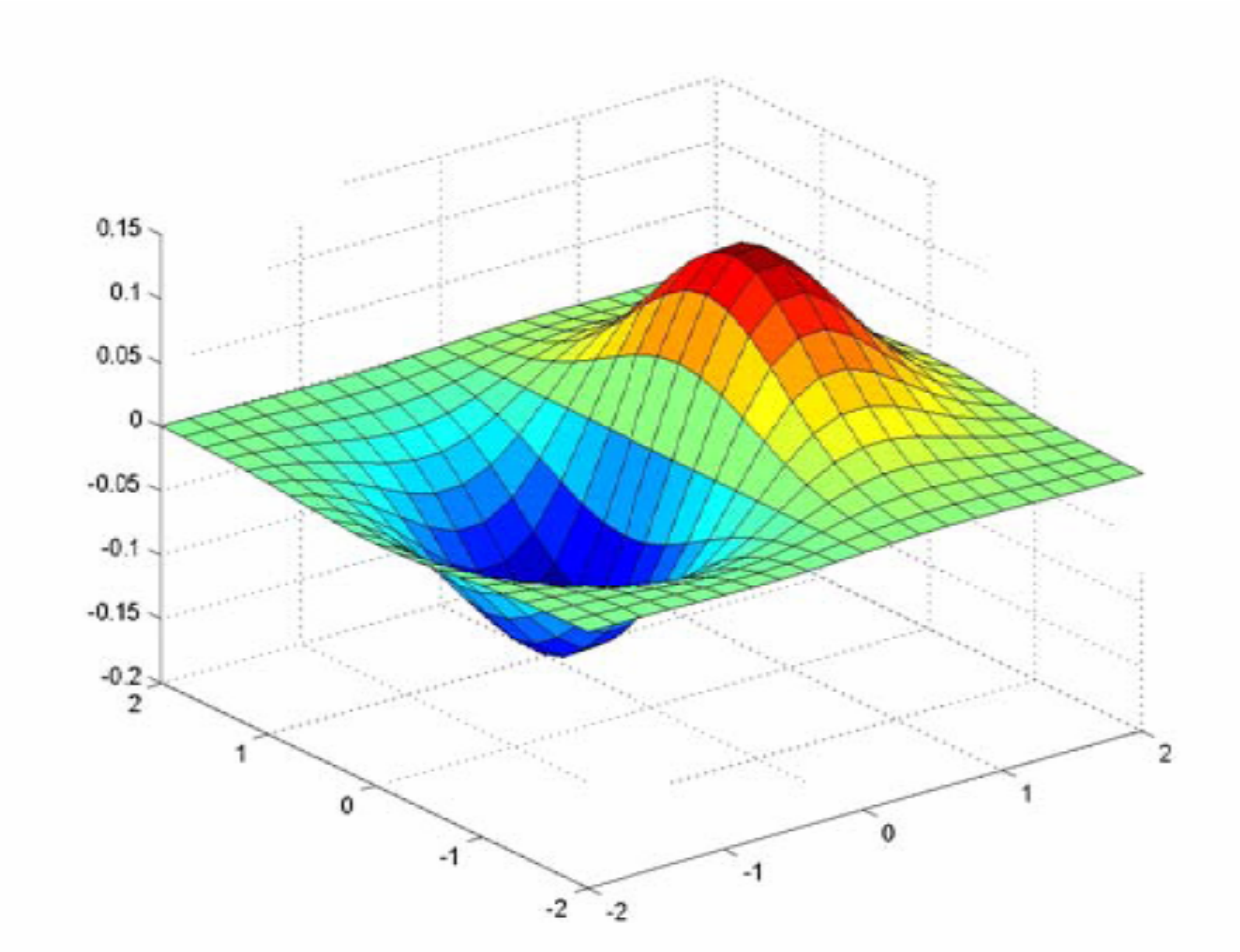
=



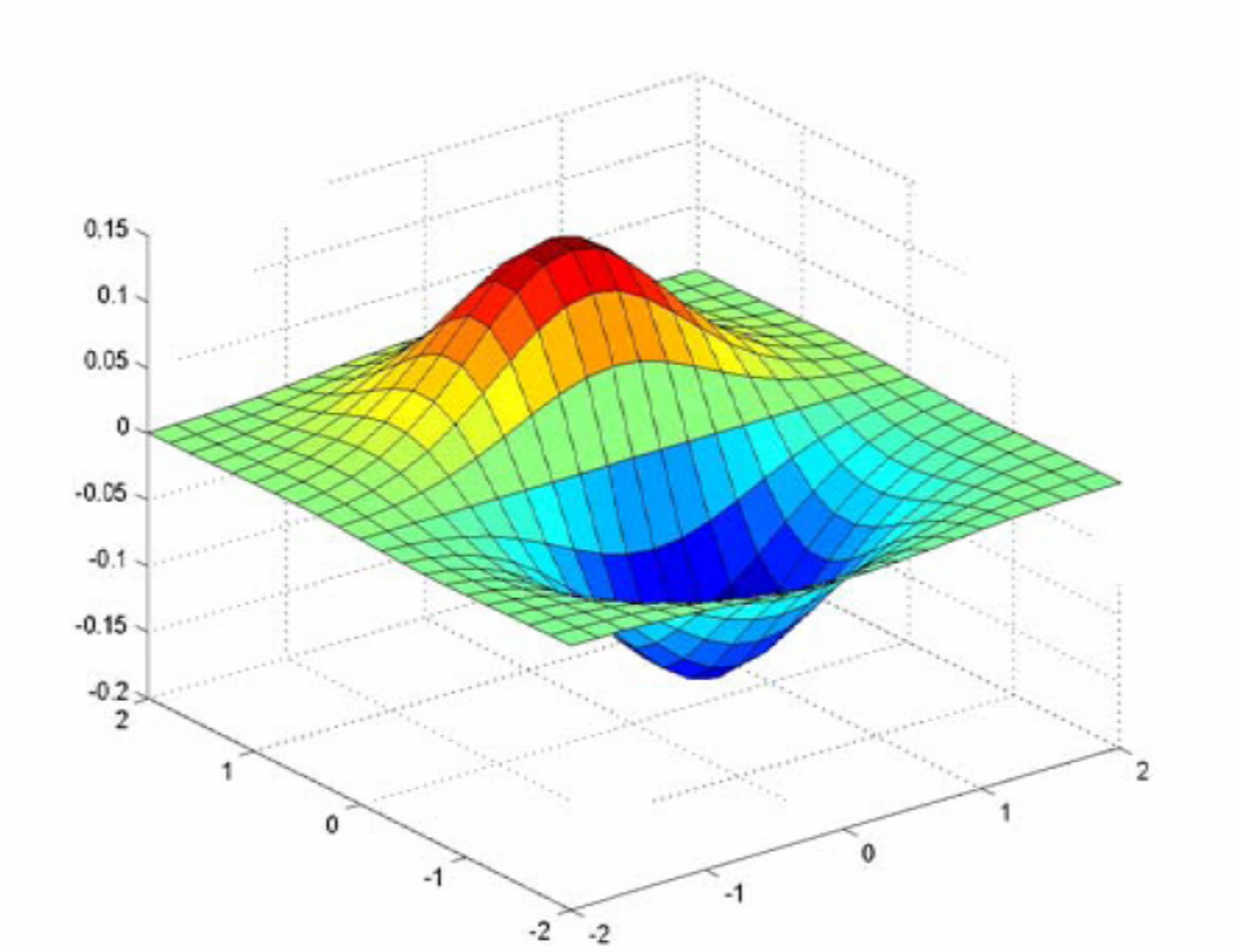
$\nabla_{\vec{u}} f$

Source: N. Snavely

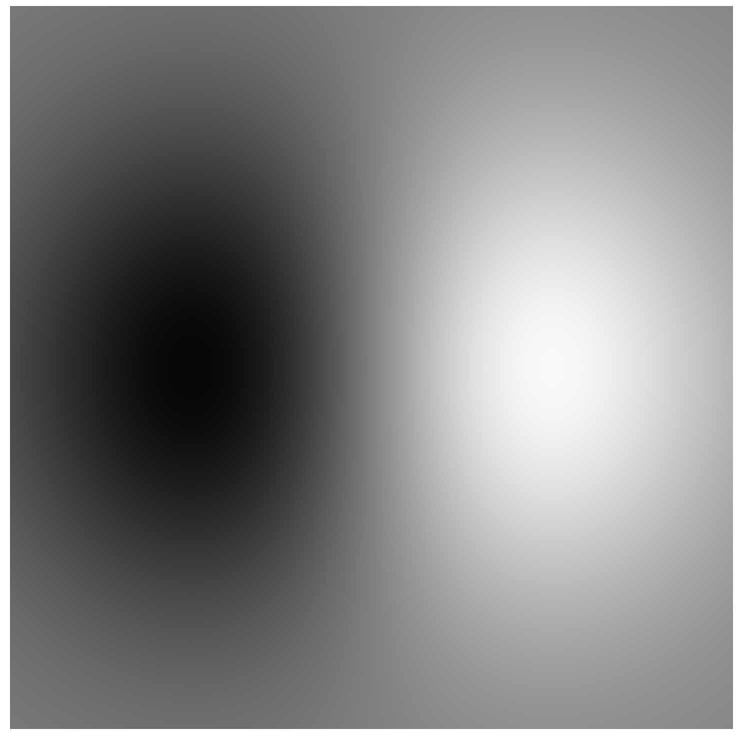
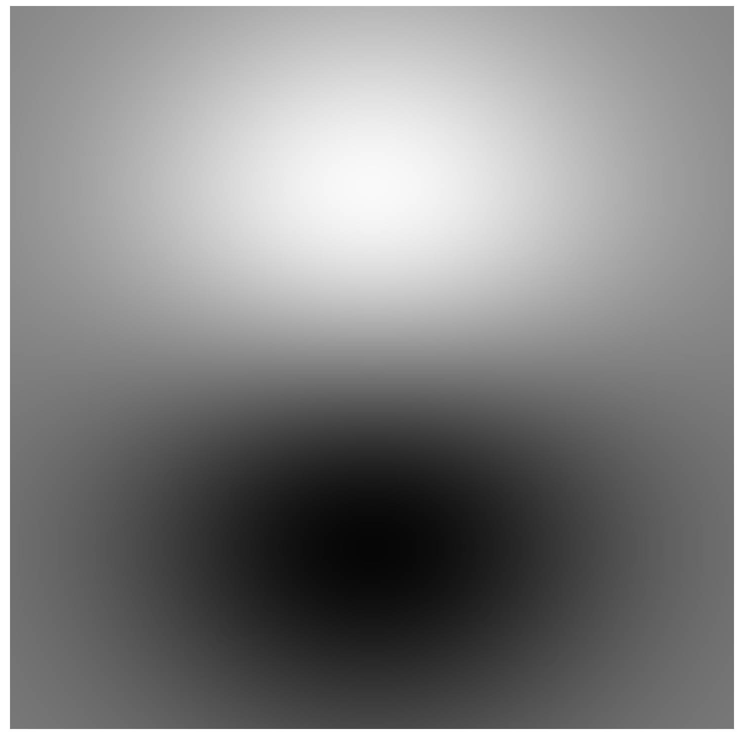
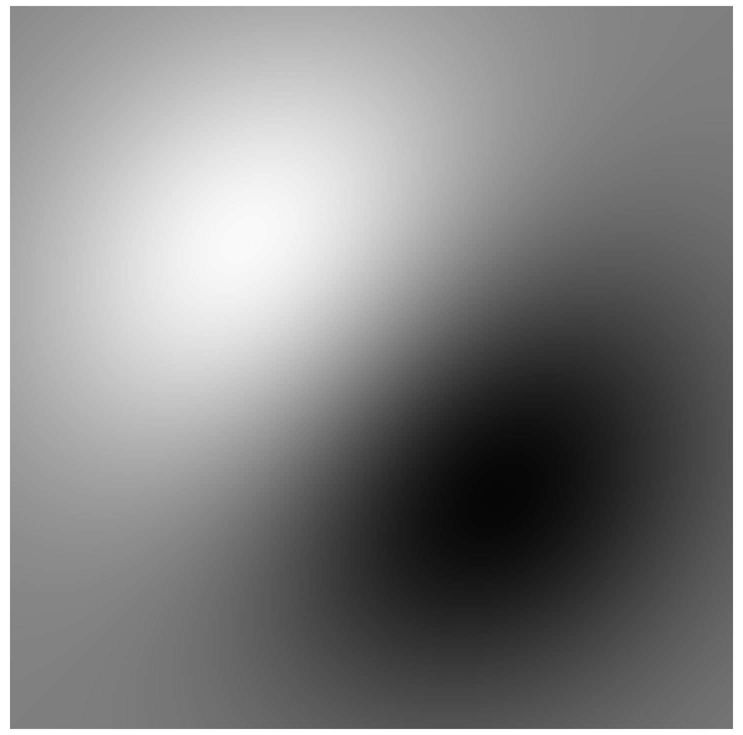
Derivative of Gaussian filter



x-direction



y-direction

$\cos(\theta)$  $+ \sin(\theta)$  $=$ 

Source: N. Snavely

The Sobel operator

- Common approximation to derivative of Gaussian
- Where does this come from?

$$\frac{1}{8}\begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline \end{array} \qquad \frac{1}{8}\begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -2 & -1 \\ \hline \end{array}$$

$$s_x \qquad\qquad s_y$$

Source: N. Snavely

An approximation to the Gaussian

- Apply filter to itself repeatedly.
- Converges to Gaussian, due to Central Limit Theorem

$$b_1 = [1 \;\; 1]$$

$$b_2 = [1\ 1] \circ [1\ 1] = [1\ 2\ 1]$$

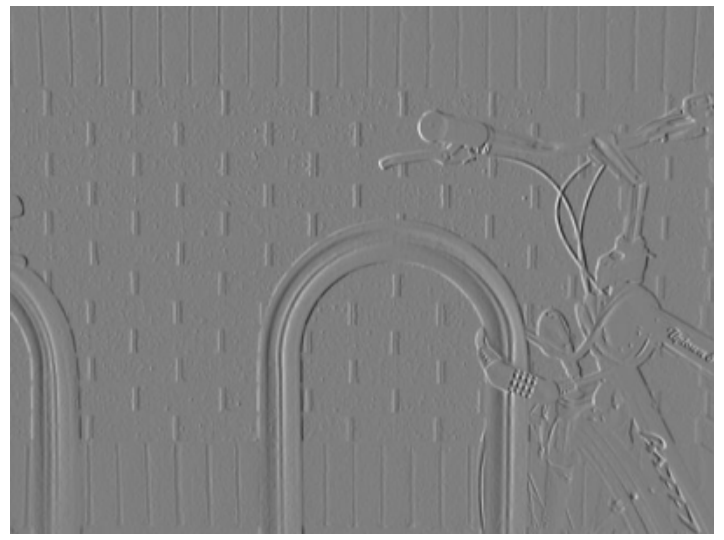
$$b_3 = [1\ 1] \circ [1\ 1] \circ [1\ 1] = [1\ 3\ 3\ 1]$$

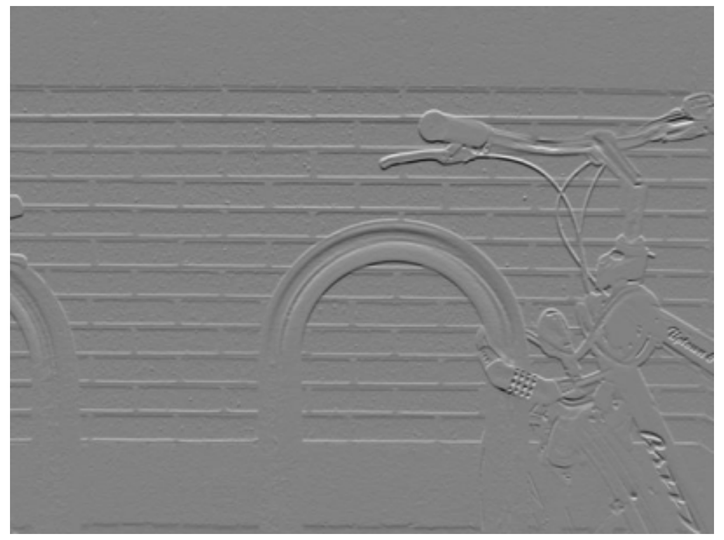
Source: Torralba, Freeman, Isola

Binomial filter

b_1					1 1					$\sigma_1^2 = 1/4$
b_2					$\boxed{1 \quad 2 \quad 1}$					$\sigma_2^2 = 1/2$
b_3					1 3 3 1					$\sigma_3^2 = 3/4$
b_4					1 4 6 4 1					$\sigma_4^2 = 1$
b_5					1 5 10 10 5 1					$\sigma_5^2 = 5/4$
b_6					1 6 15 20 15 6 1					$\sigma_6^2 = 3/2$
b_7					1 7 21 35 35 21 7 1					$\sigma_7^2 = 7/4$
b_8					1 8 28 56 70 56 28 8 1					$\sigma_8^2 = 2$

Sobel operator: example





Source: N. Snavely / Wikipedia

Next class: frequency