

Image Filtering

Module 2

What is a Feature?

In computer vision, a "feature" refers to a distinctive piece of information in an image that is useful for specific computational tasks such as recognition, detection, or tracking. Features can range from simple elements like corners and edges to more complex structures like color regions, or texture patterns.

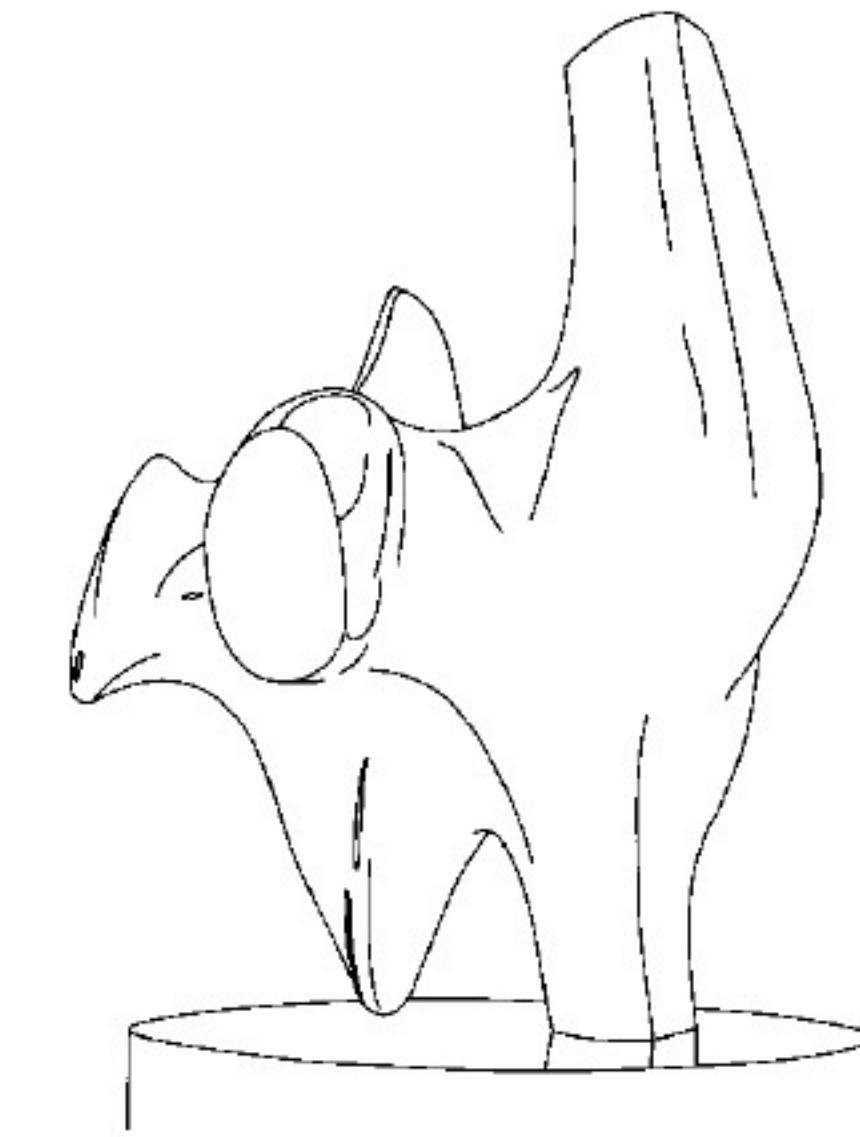
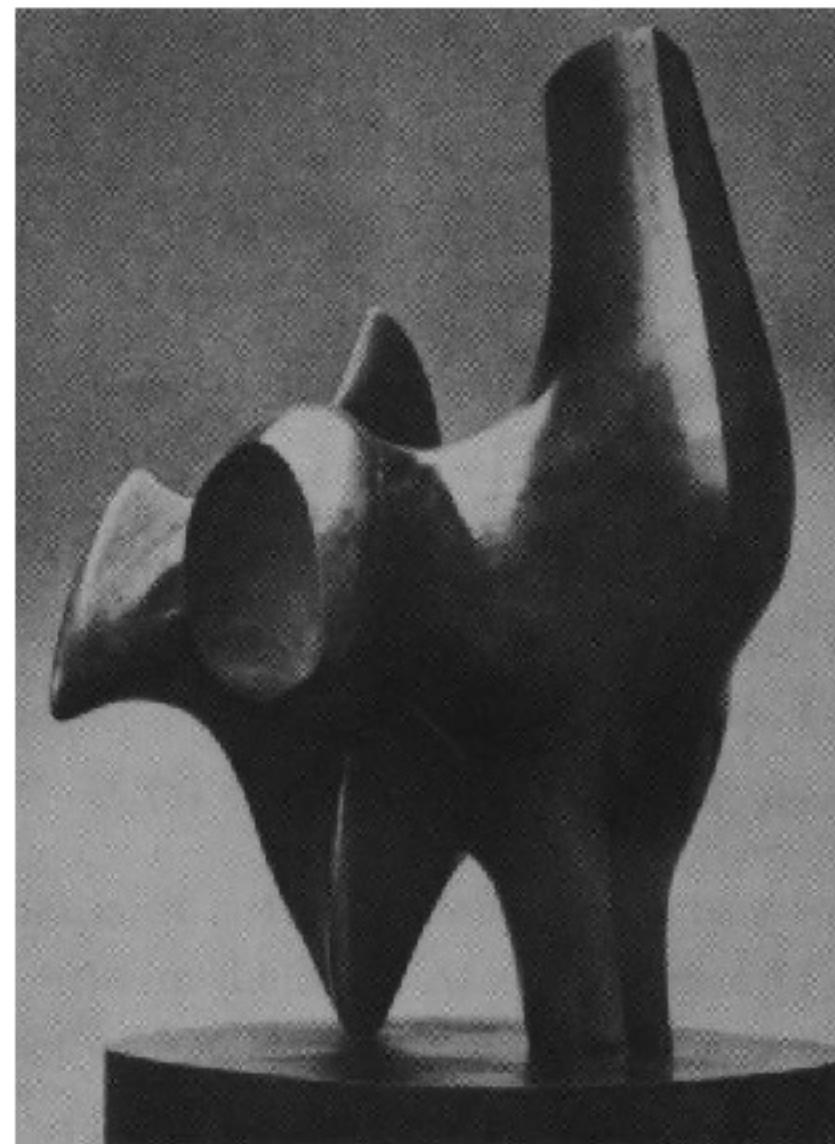


How do we understand the different parts of this image?



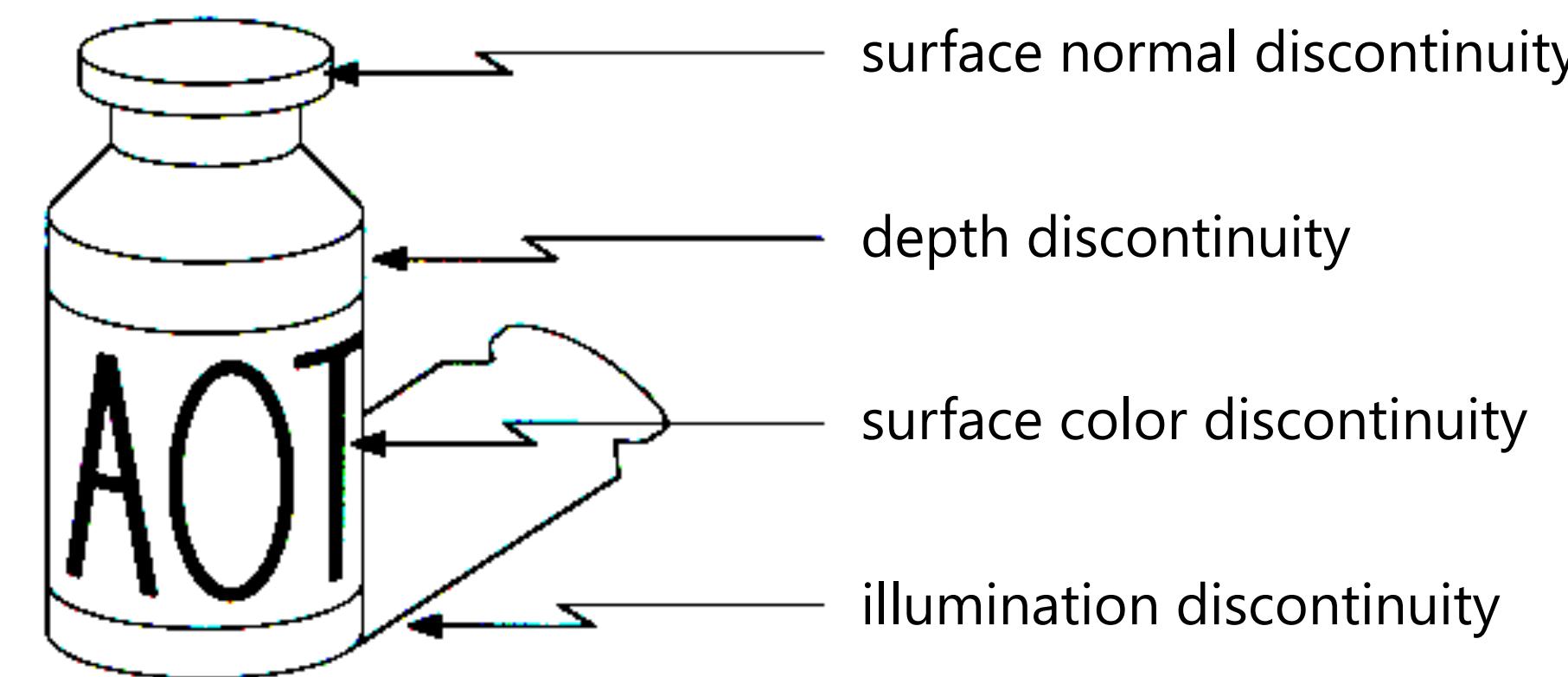
**We already discussed color but what about edges?
How do we define an edge?**

Edge detection



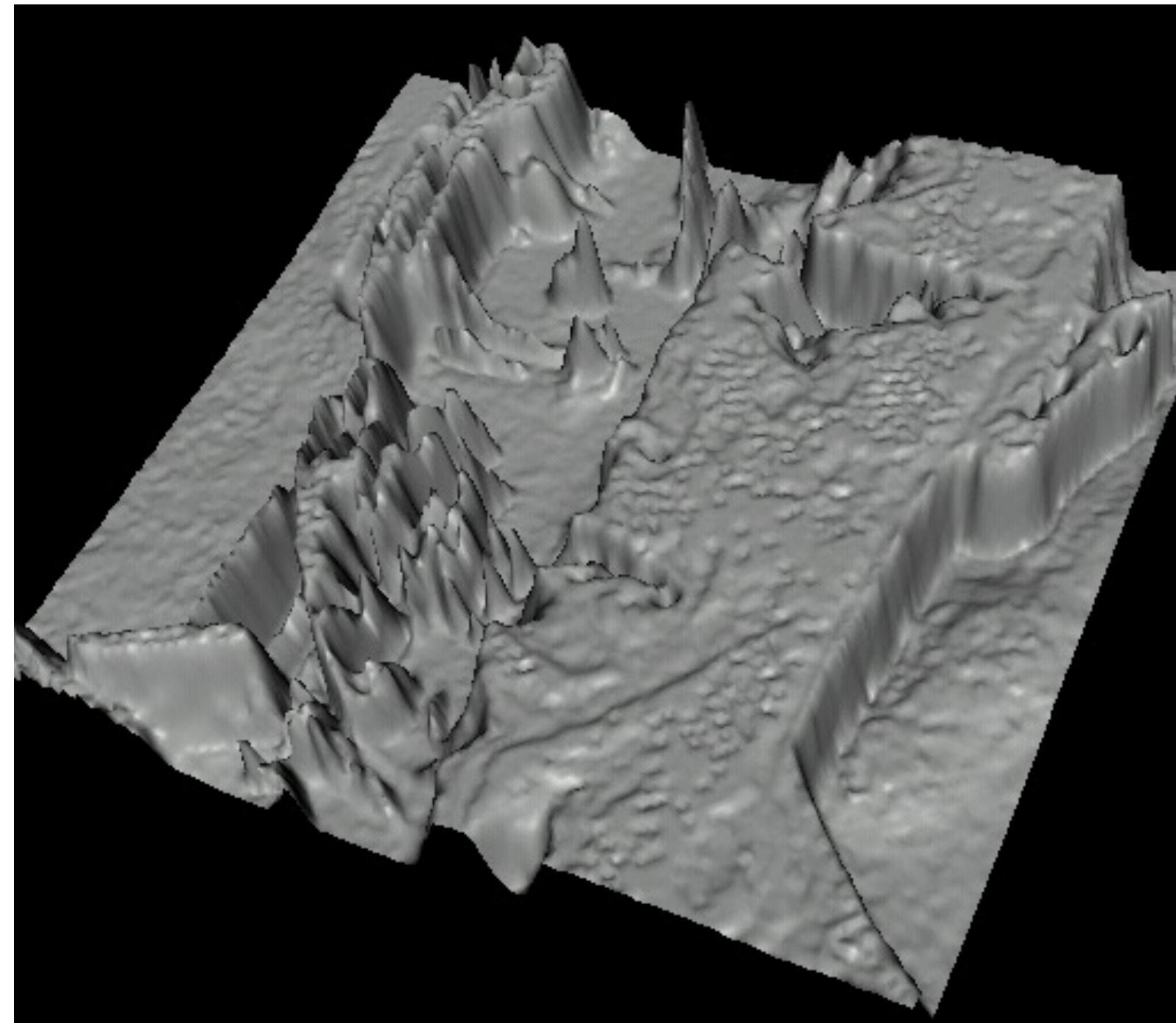
- Convert a 2D image into a set of curves
 - Extracts salient features of the scene
 - More compact than pixels

Origin of edges



- Edges are caused by a variety of factors

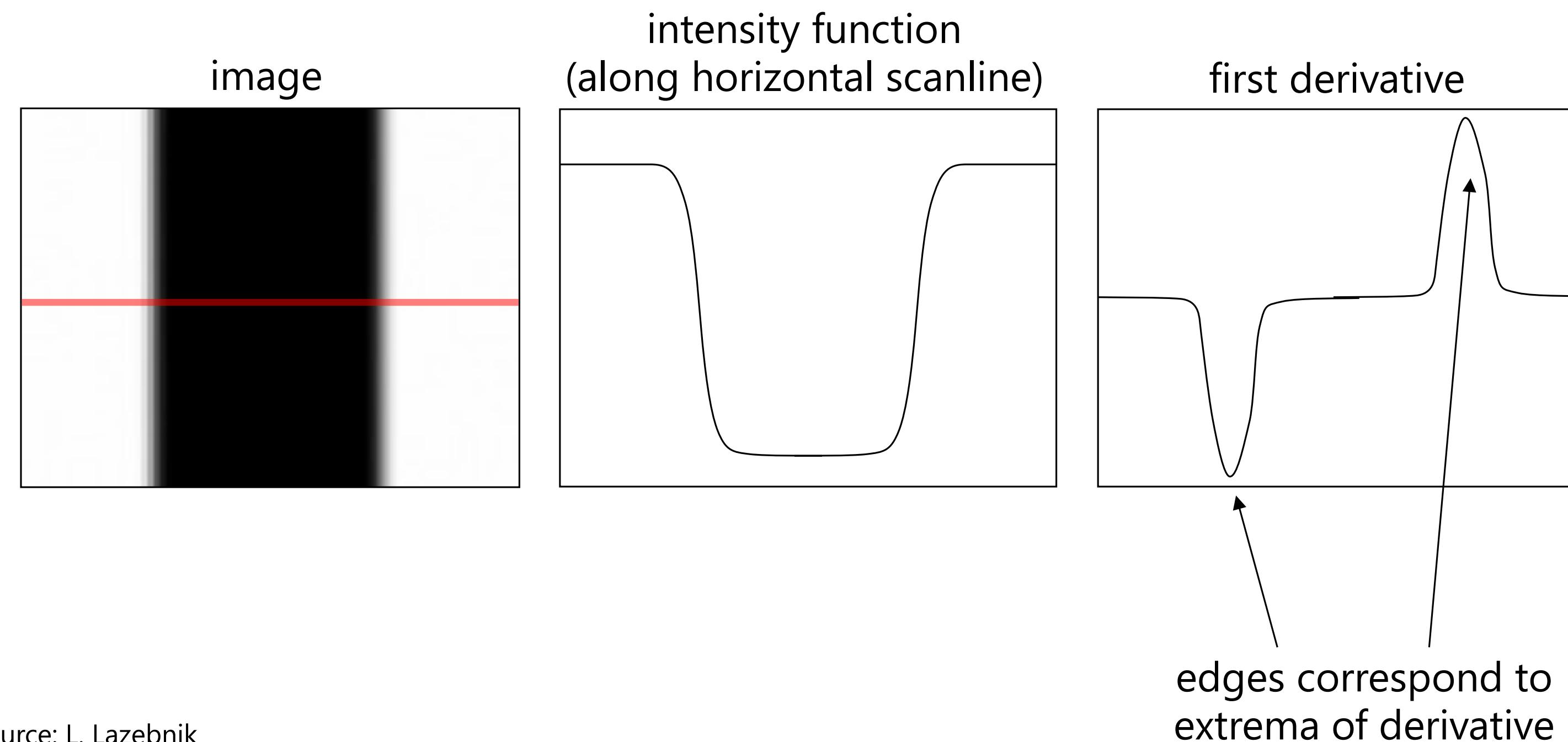
Images as functions...



- Edges look like steep cliffs

Characterizing edges

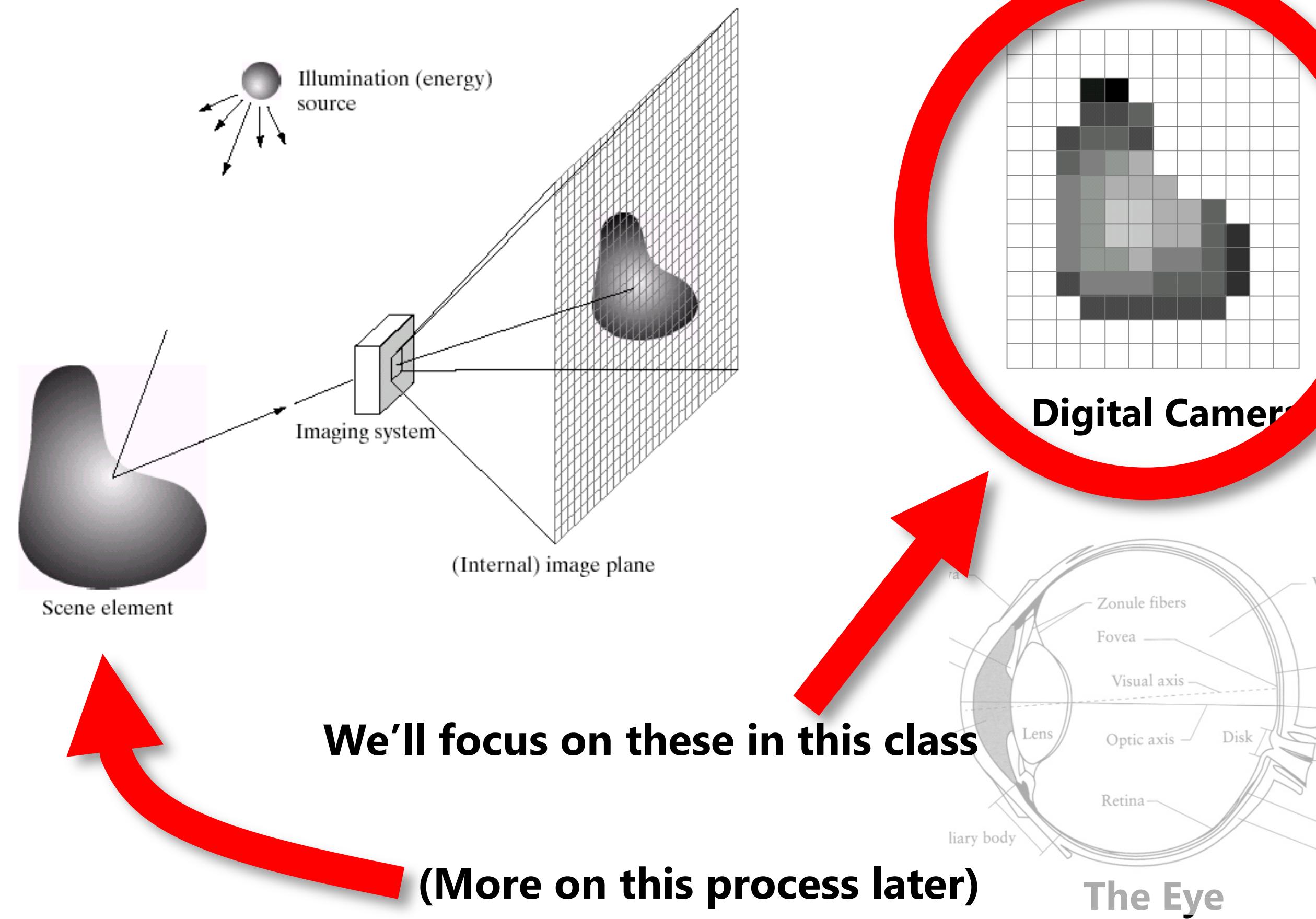
- An edge is a place of *rapid change* in the image intensity function



Source: L. Lazebnik

How do we discover edge regions within our image?

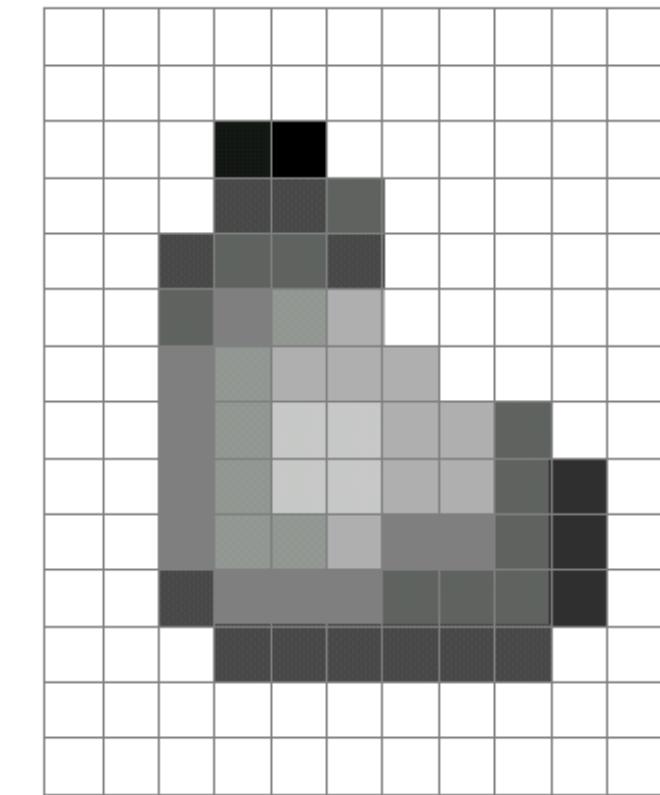
What is an image?



Source: A. Efros

What is an image?

- A grid (matrix) of intensity values



=

255	255	255	255	255	255	255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255	255	255	255	255	255	255	255	255
255	255	255	20	0	255	255	255	255	255	255	255	255	255	255	255
255	255	255	75	75	75	255	255	255	255	255	255	255	255	255	255
255	255	75	95	95	75	255	255	255	255	255	255	255	255	255	255
255	255	96	127	145	175	255	255	255	255	255	255	255	255	255	255
255	255	127	145	175	175	175	255	255	255	255	255	255	255	255	255
255	255	127	145	200	200	175	175	95	255	255	255	255	255	255	255
255	255	127	145	200	200	175	175	95	47	255	255	255	255	255	255
255	255	127	145	145	175	127	127	95	47	255	255	255	255	255	255
255	255	74	127	127	127	95	95	95	47	255	255	255	255	255	255
255	255	255	74	74	74	74	74	74	74	255	255	255	255	255	255
255	255	255	255	255	255	255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255	255	255	255	255	255	255	255	255

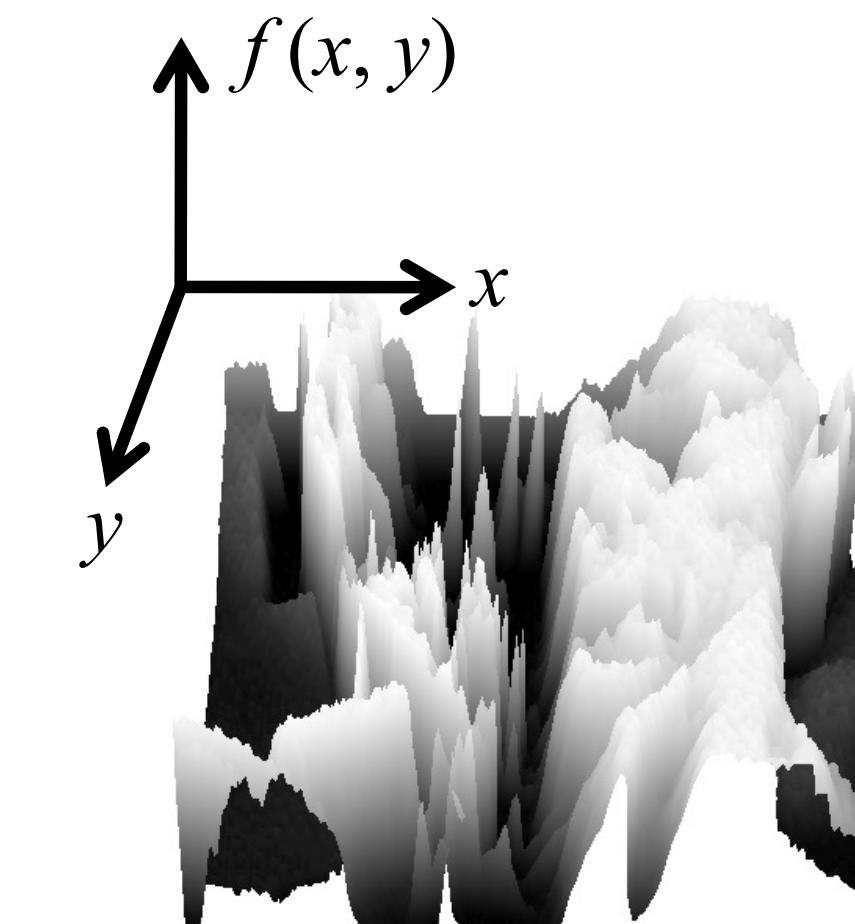
(common to use one byte per value: 0 = black, 255 = white)

What is an image?

- Can think of a (grayscale) image as a **function** f from \mathbb{R}^2 to \mathbb{R} :
 - $f(x,y)$ gives the **intensity** at position (x,y)



[snoop](#)

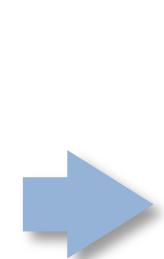
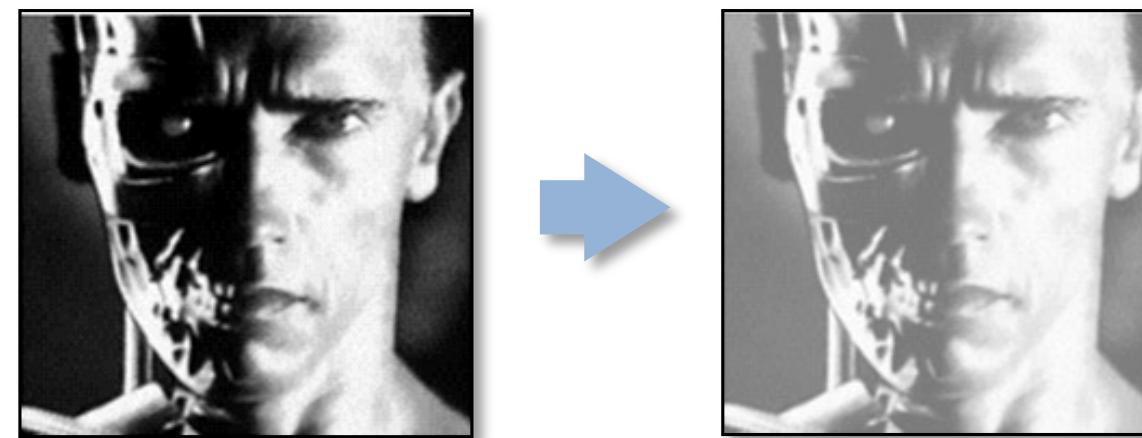


[3D view](#)

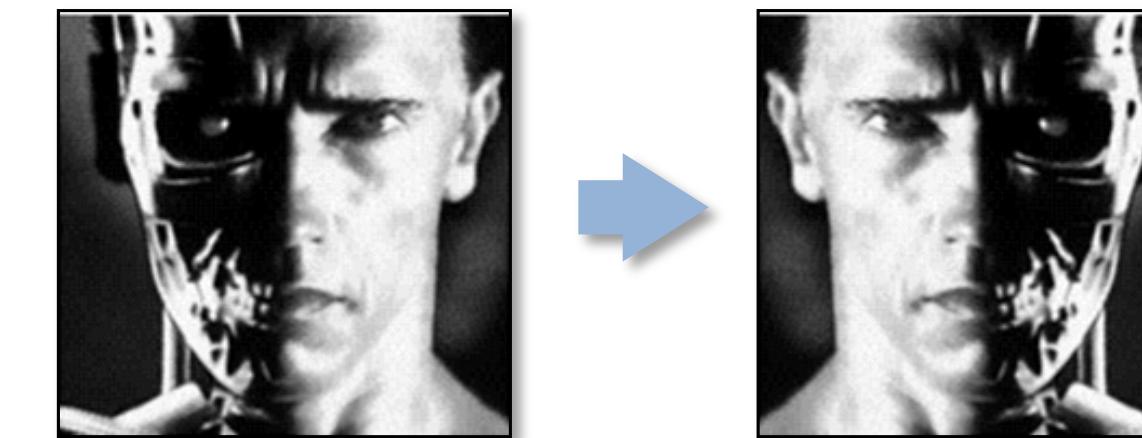
- A **digital** image is a discrete (**sampled, quantized**) version of this function

Image transformations

- As with any function, we can apply operators to an image



$$g(x,y) = f(x,y) + 20$$



$$g(x,y) = f(-x,y)$$

- Today we'll talk about a special kind of operator, *convolution* (linear filtering)

Filters

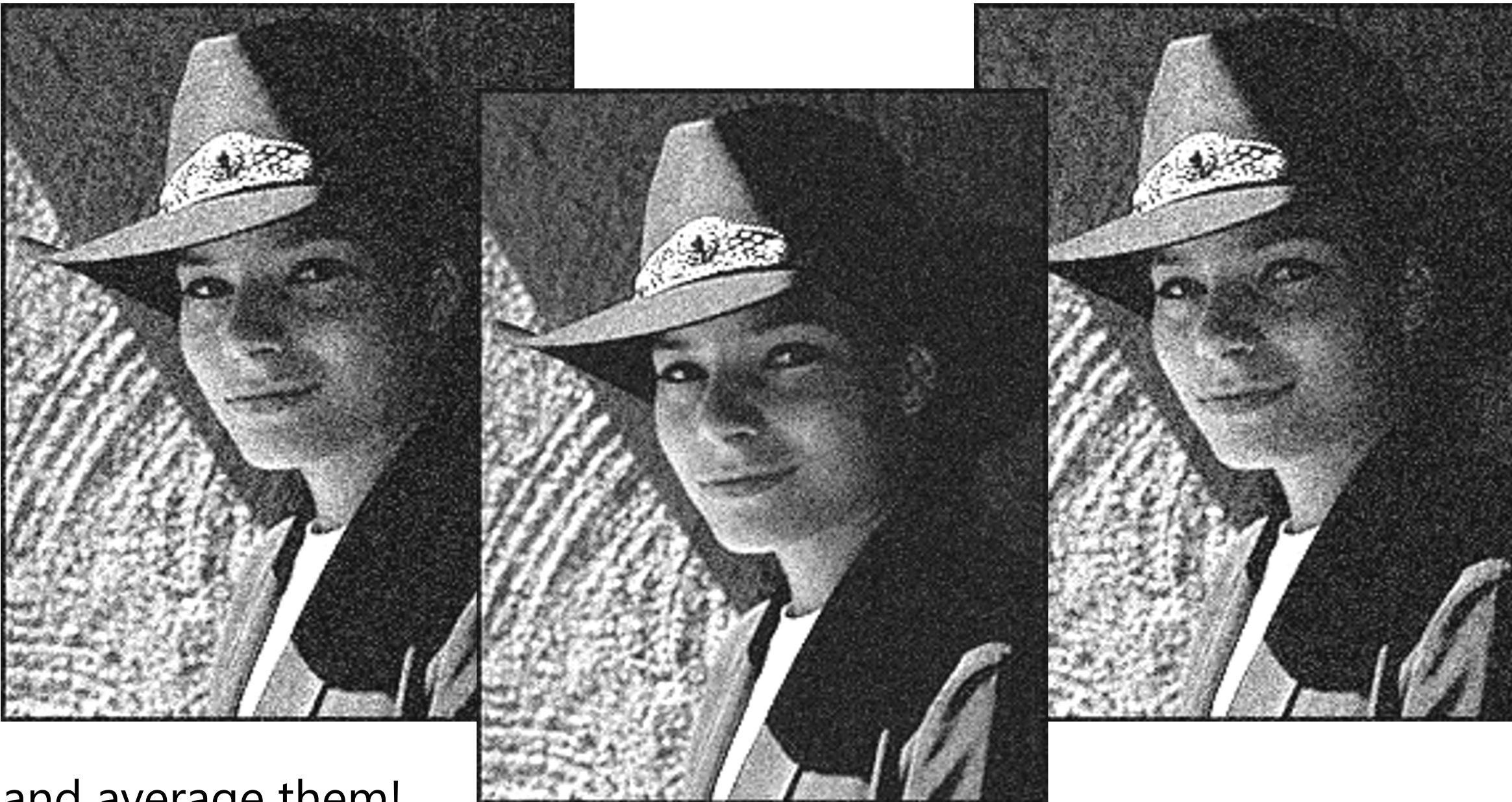
- Filtering
 - Form a new image whose pixel values are a combination of the original pixel values
- Why?
 - To get useful information from images
 - E.g., extract edges or contours (to understand shape)
 - To enhance the image
 - E.g., to remove noise
 - E.g., to sharpen and “enhance image” a la CSI
 - A key operator in Convolutional Neural Networks

Image Processing Problems

- Image Restoration
 - denoising
 - deblurring
- Image Compression
 - JPEG, HEIF, MPEG, ...
- Locating Structural Features
 - corners
 - edges

Question: Noise reduction

- Given a camera and a still scene, how can you reduce noise?



Take lots of images and average them!

What's the next best thing?

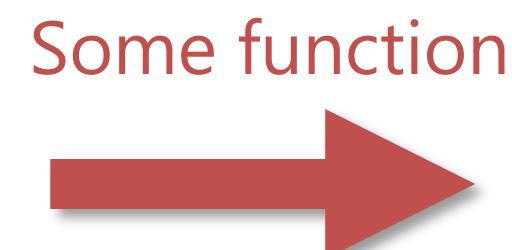
Source: S. Seitz

Image filtering

- Modify the pixels in an image based on some function of a local neighborhood of each pixel

10	5	3
4	5	1
1	1	7

Local image data

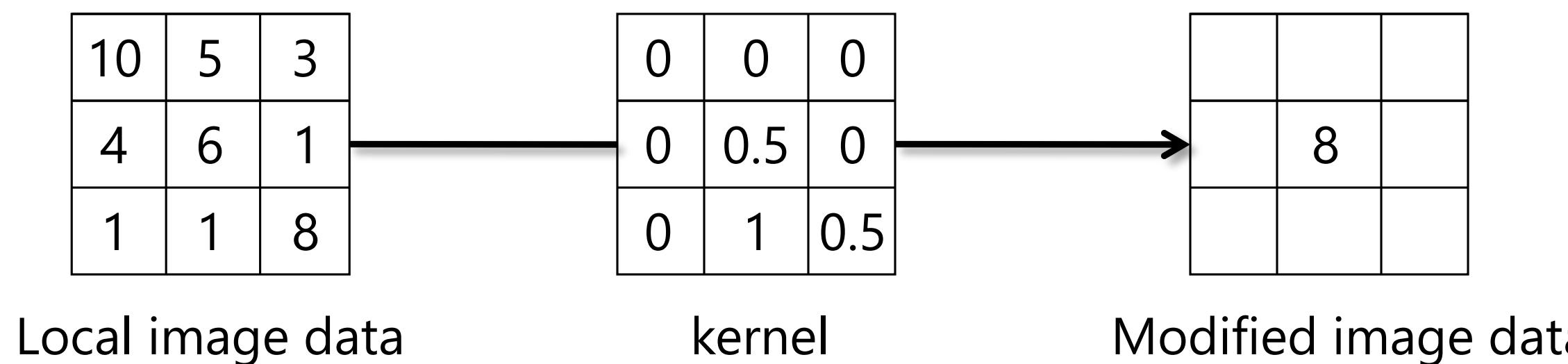


	7	

Modified image data

Linear filtering

- One simple version of filtering: linear filtering (cross-correlation, convolution)
 - Replace each pixel by a linear combination (a weighted sum) of its neighbors
- The prescription for the linear combination is called the “kernel” (or “mask”, “filter”)



Source: L. Zhang

Convolution

- Let F be the image and H be the Kernel ($2k + 1 \times 2k + 1$) and G will be the output image

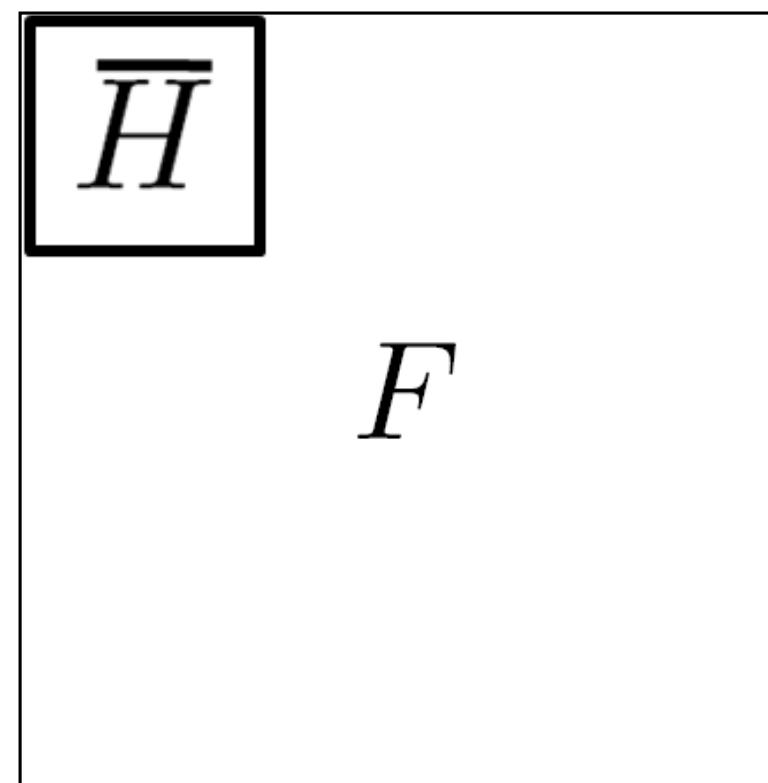
$$G[i, j] = \sum_{u=-k}^k \sum_{v=-k}^k H[u, v]F[i - u, j - v]$$

This is called a **convolution** operation:

$$G = H * F$$

- Convolution is **commutative** and **associative**

Convolution



Adapted from F. Durand

Mean filtering

$$\begin{array}{c|c} \begin{matrix} & & \\ & & \\ & & \end{matrix} & * \\ H & \end{array} \quad \begin{array}{c|c} \begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 90 & 90 & 90 & 90 & 90 & 0 & 0 \\ 0 & 0 & 0 & 90 & 90 & 90 & 90 & 90 & 0 & 0 \\ 0 & 0 & 0 & 90 & 90 & 90 & 90 & 90 & 0 & 0 \\ 0 & 0 & 0 & 90 & 90 & 90 & 90 & 90 & 0 & 0 \\ 0 & 0 & 0 & 90 & 0 & 90 & 90 & 90 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 90 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{matrix} & = \\ F & \end{array} \quad \begin{array}{c|c} \begin{matrix} & & & & & & & \\ 0 & 10 & 20 & 30 & 30 & 30 & 20 & 10 \\ 0 & 20 & 40 & 60 & 60 & 60 & 40 & 20 \\ 0 & 30 & 60 & 90 & 90 & 90 & 60 & 30 \\ 0 & 30 & 50 & 80 & 80 & 90 & 60 & 30 \\ 0 & 30 & 50 & 80 & 80 & 90 & 60 & 30 \\ 0 & 20 & 30 & 50 & 50 & 60 & 40 & 20 \\ 10 & 20 & 30 & 30 & 30 & 30 & 20 & 10 \\ 10 & 10 & 10 & 0 & 0 & 0 & 0 & 0 \end{matrix} & \\ G & \end{array}$$

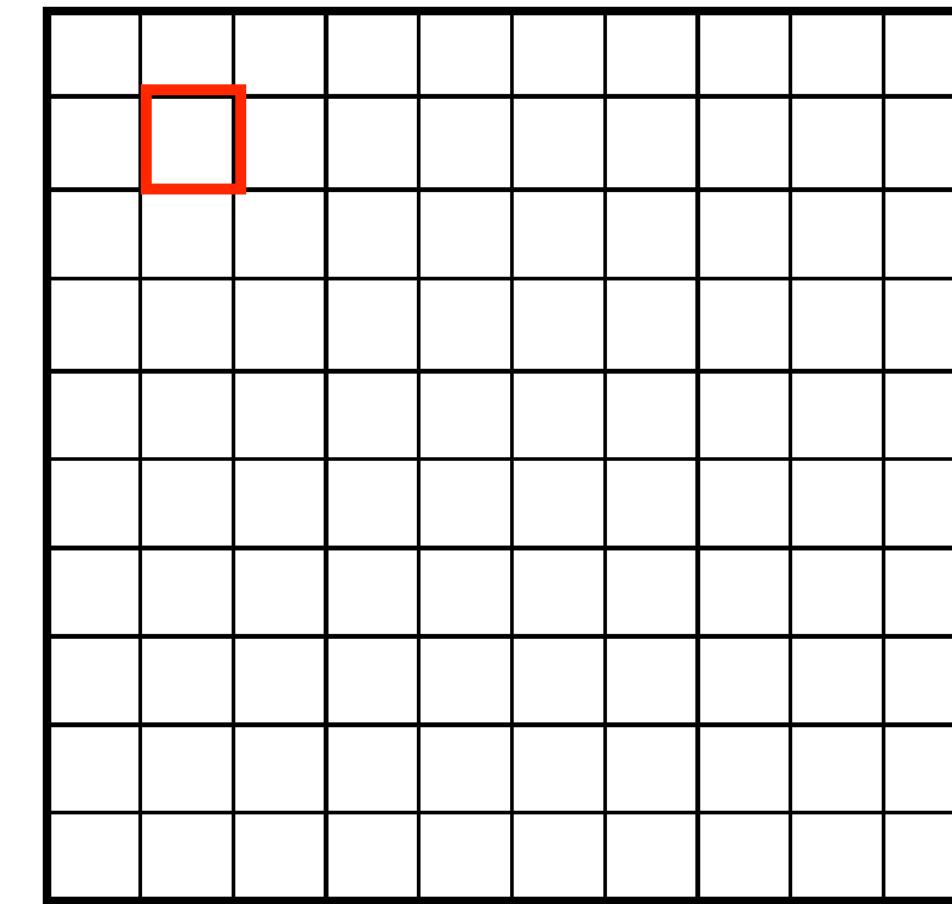
The diagram illustrates the mean filtering process. It shows three matrices: H , F , and G . Matrix H is a 3x3 kernel with all elements set to 0. Matrix F is a 10x10 input image with values ranging from 0 to 90. A 3x3 submatrix in F at indices (7,7) to (9,9) is highlighted with a red border. Matrix G is the result of applying the H filter to F , resulting in a 10x10 output matrix where the central element is 20, indicating the average of the 3x3 kernel applied to the highlighted region in F .

Mean filtering/Moving average

$F[x, y]$

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

$G[x, y]$

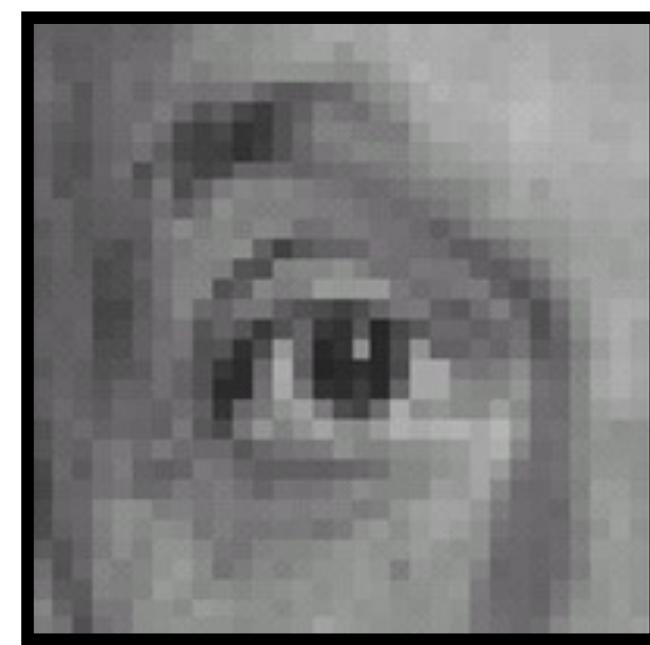


Mean filtering/Moving average

$$F[x, y]$$

$$G[x, y]$$

Linear filters: examples



Original

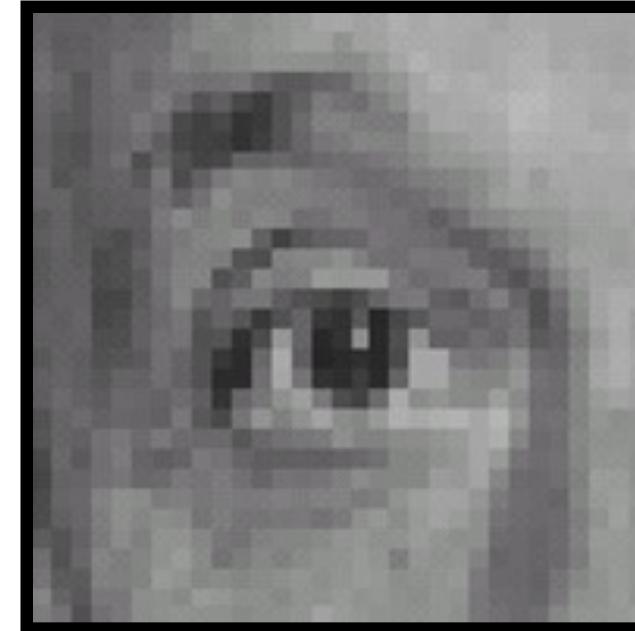
*

0	0	0
0	1	0
0	0	0

Source: D. Lowe

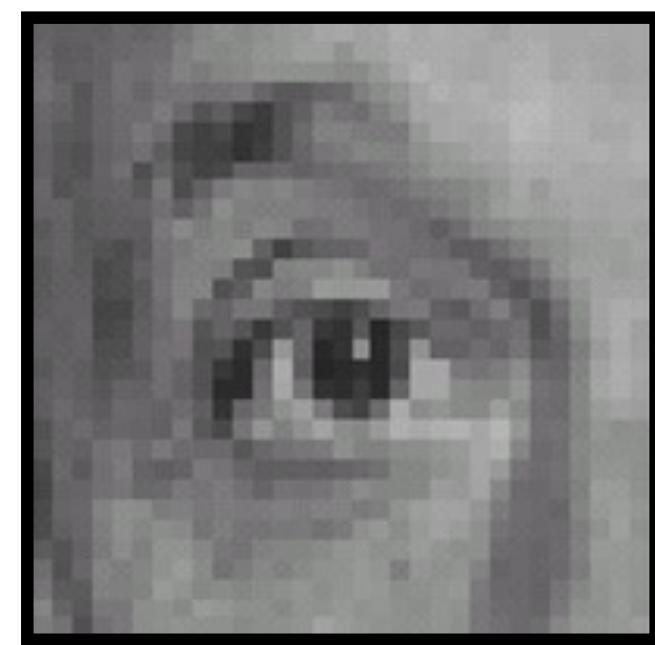
What image operation does filtering with this kernel perform?
([0 0 0; 0 1 0; 0 0 0])

Linear filters: examples


$$\text{Original} \quad * \quad \begin{array}{|c|c|c|} \hline 0 & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array} \quad = \quad \text{Identical image}$$


Source: D. Lowe

Linear filters: examples



Original

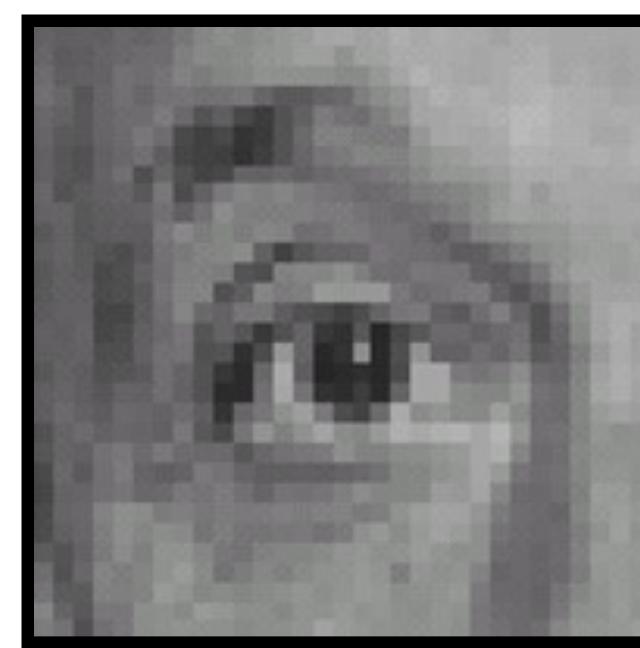
*

0	0	0
1	0	0
0	0	0

Source: D. Lowe

What image operation does filtering with this kernel perform?
([0 0 0; 1 0 0; 0 0 0])

Linear filters: examples



Original

*

$$\begin{array}{|c|c|c|} \hline 0 & 0 & 0 \\ \hline 1 & 0 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array}$$

=



Shifted left by 1 pixel

Source: D. Lowe

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Identity kernel

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Edge detection

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Sharpen kernel

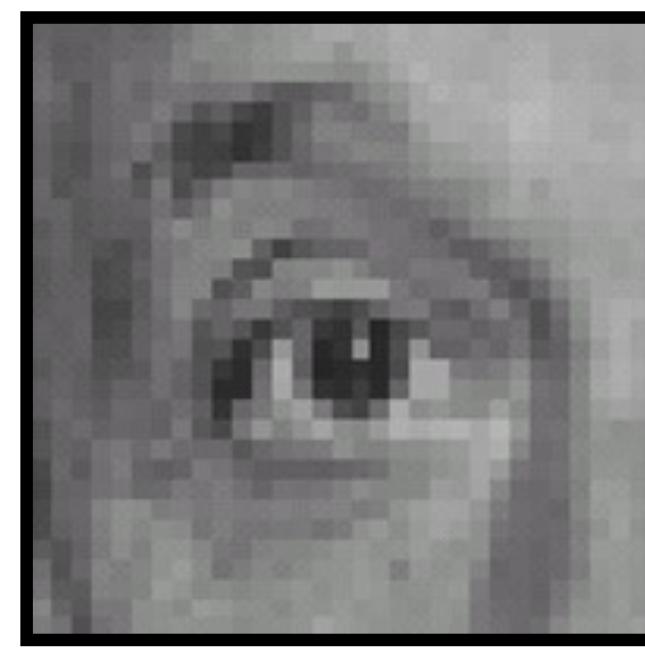
$$\frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Box blur

$$\frac{1}{256} \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

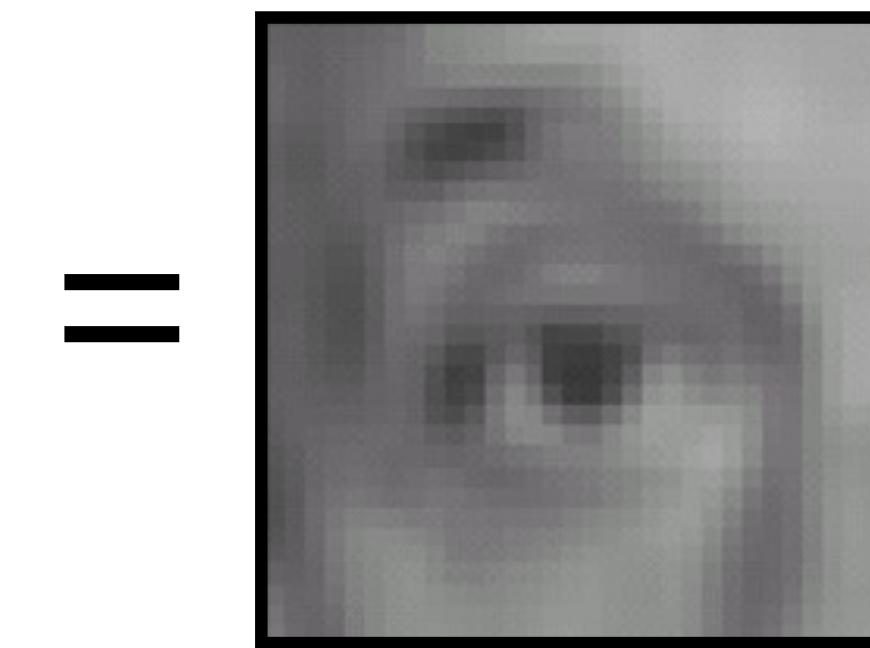
Gaussian blurr kernel

Linear filters: examples



Original

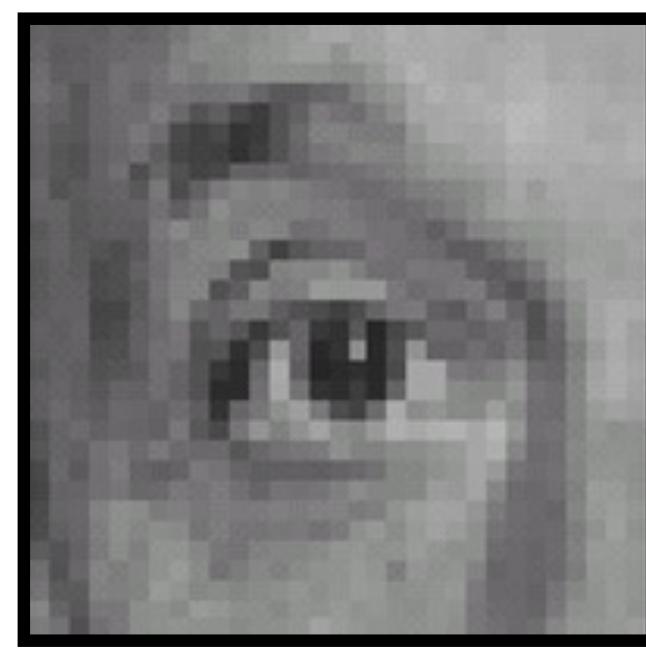
$$\ast \frac{1}{9} \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$



Blur (with a mean filter)

Source: D. Lowe

Linear filters: examples



Original

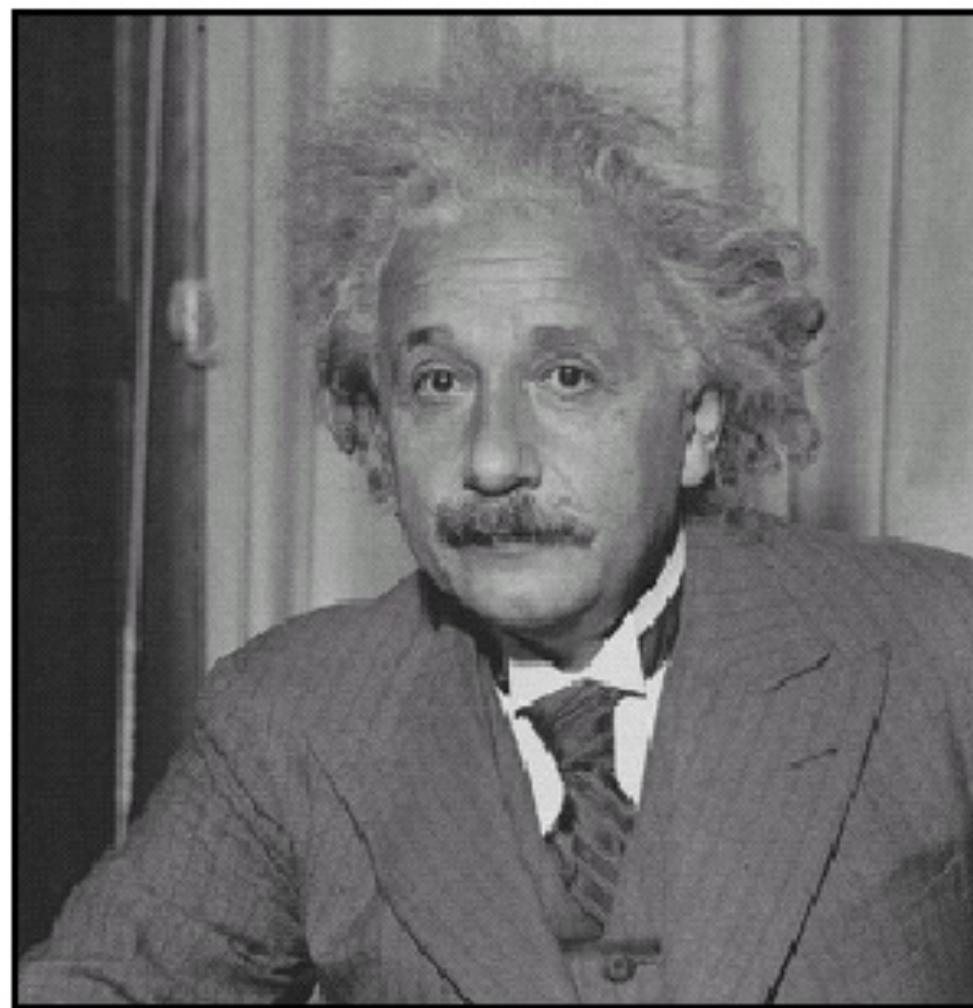
$$\ast \left(\begin{array}{|ccc|} \hline 0 & 0 & 0 \\ \hline 0 & 2 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array} - \frac{1}{9} \begin{array}{|ccc|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array} \right) =$$



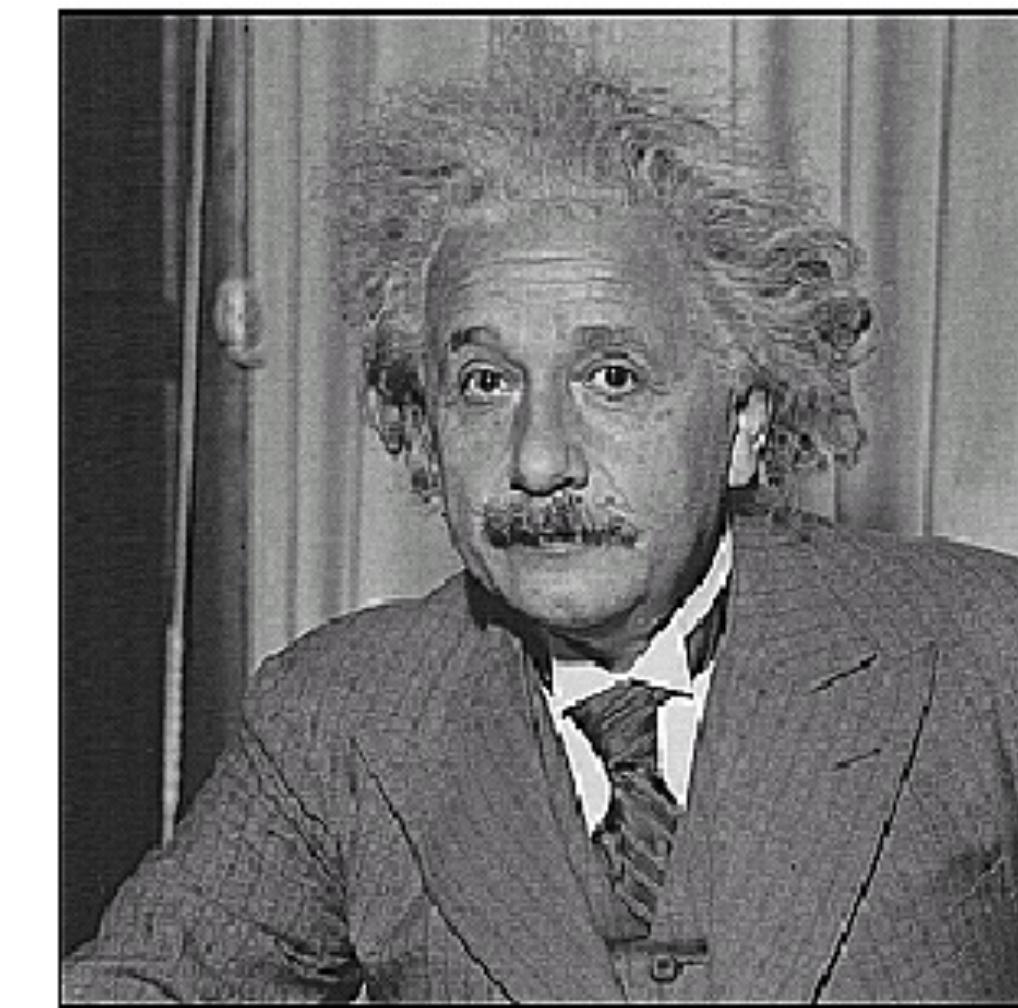
Sharpening filter
(accentuates edges)

Source: D. Lowe

Sharpening



before



after

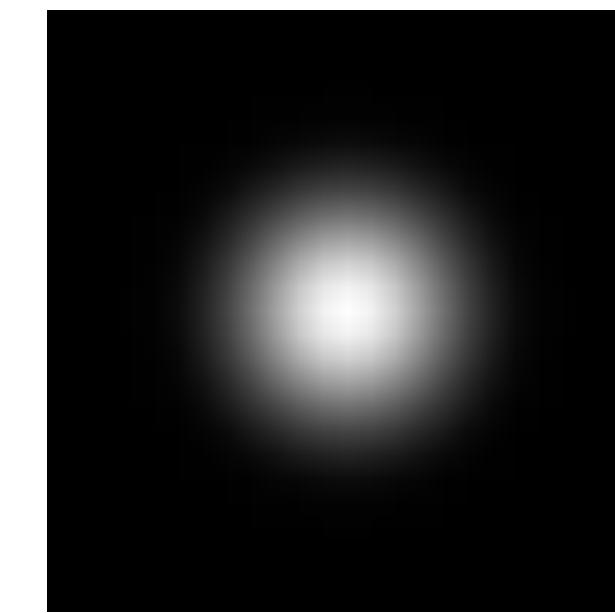
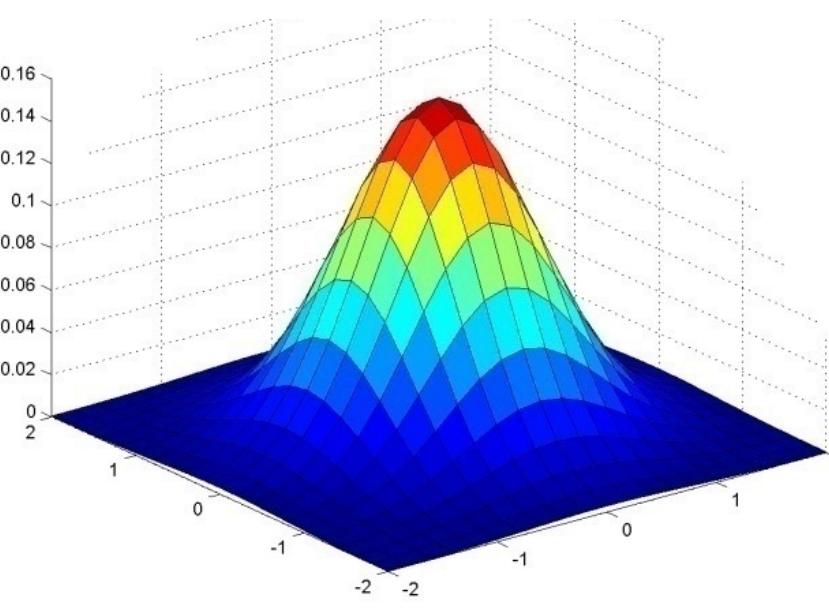
Source: D. Lowe

Smoothing with box filter revisited



Source: D. Forsyth

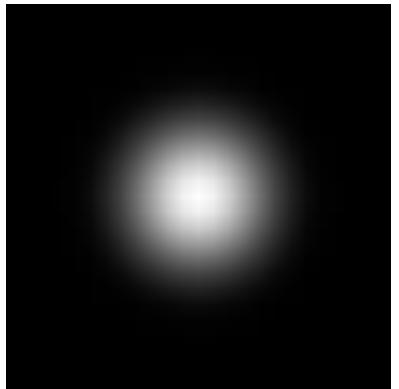
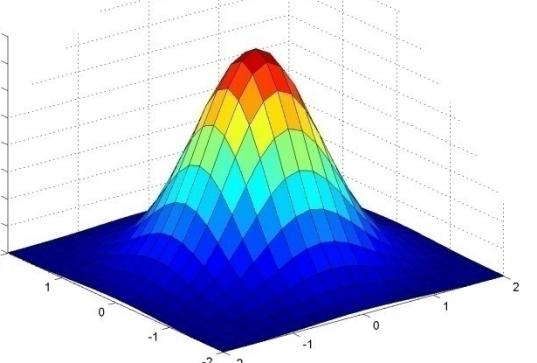
Gaussian kernel



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

Rasmussen

Gaussian kernel



Here is an example of a simple 3x3 Gaussian matrix:

0.075 0.124 0.075
0.124 0.204 0.124
0.075 0.124 0.075

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

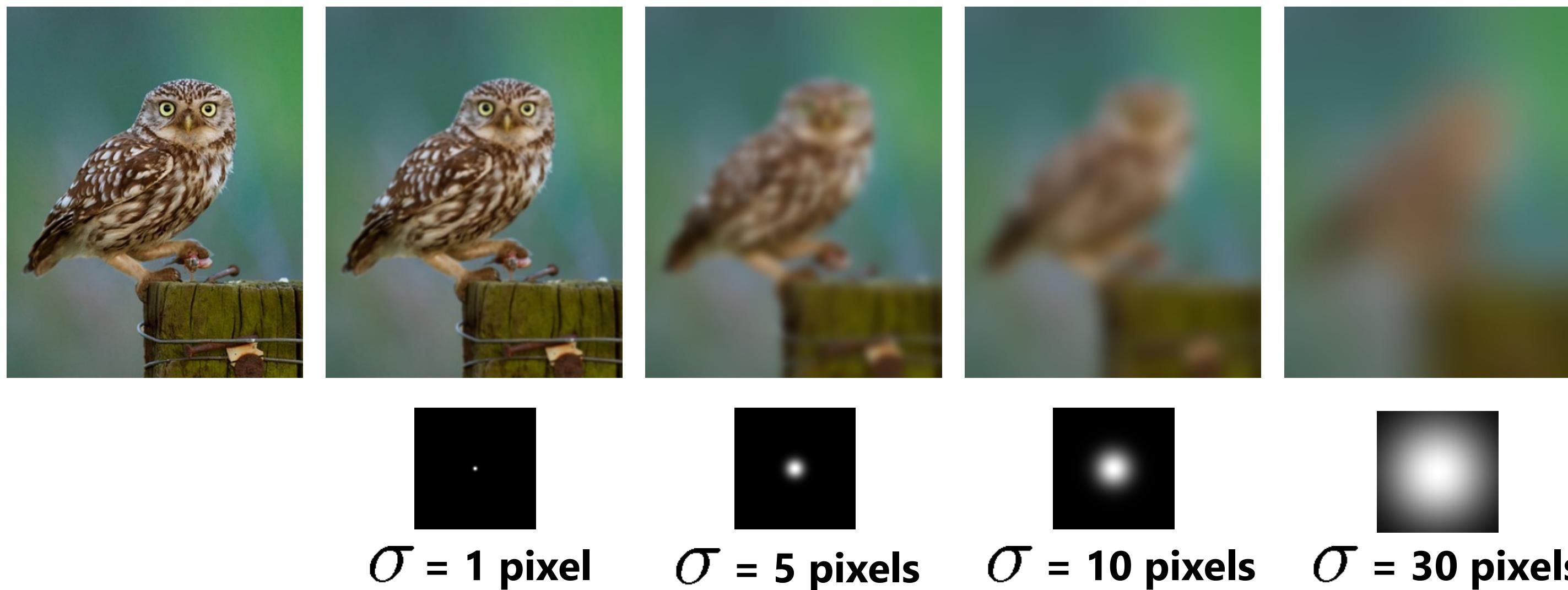
Source: C. Rasmussen

- x,y : Coordinates in 2D space.
- σ : Standard deviation, controlling the spread of the Gaussian function.
- e : Base of the natural logarithm.

The Gaussian function has the following properties:

- Symmetry: It is symmetric around its mean.
- Bell Shape: It has the shape of a "bell" which is determined by its standard deviation (σ).
- Non-Zero Everywhere: It is technically non-zero everywhere, though it approaches zero as you move away from the mean.
- Controlled by σ : A higher σ means a wider "bell," and a lower σ means a narrower "bell."

Gaussian filters

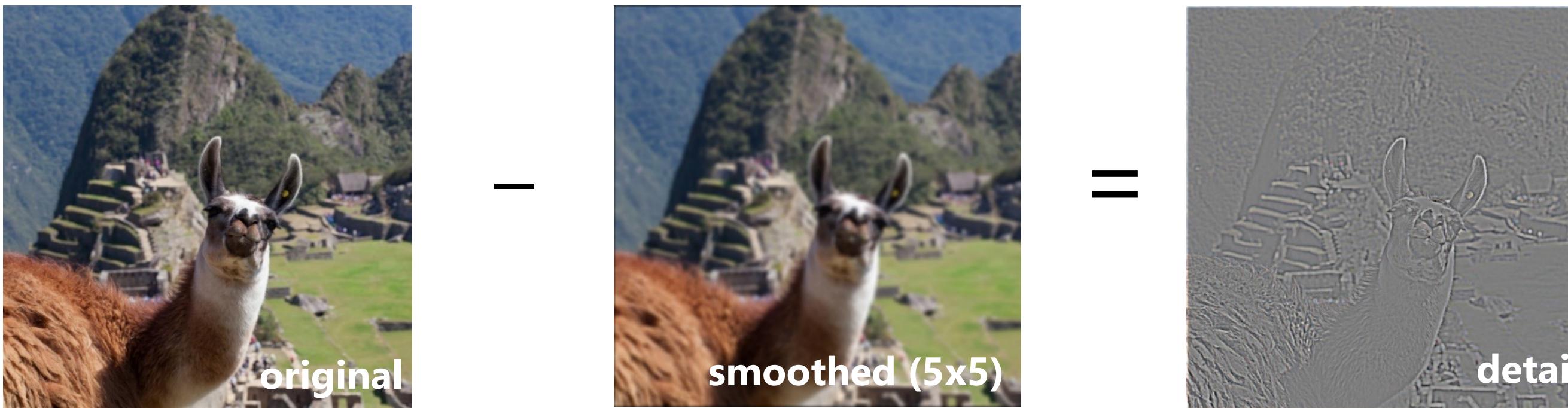


Gaussian filter

- Removes “high-frequency” components from the image
(low-pass filter)

Sharpening revisited

- What does blurring take away?



(This “detail extraction” operation is also called a **high-pass filter**)

Let's add it back:

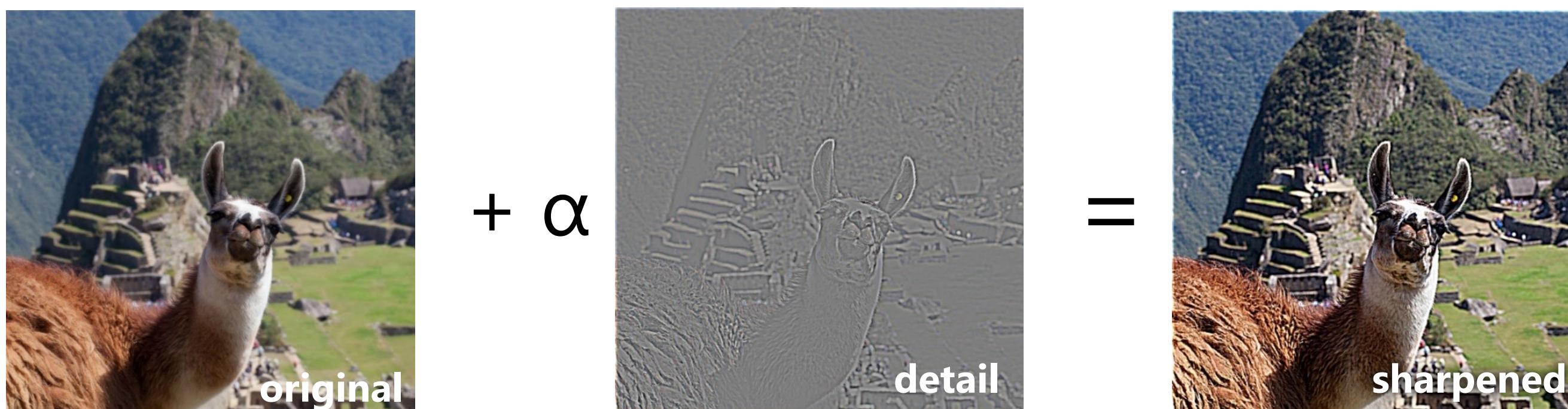
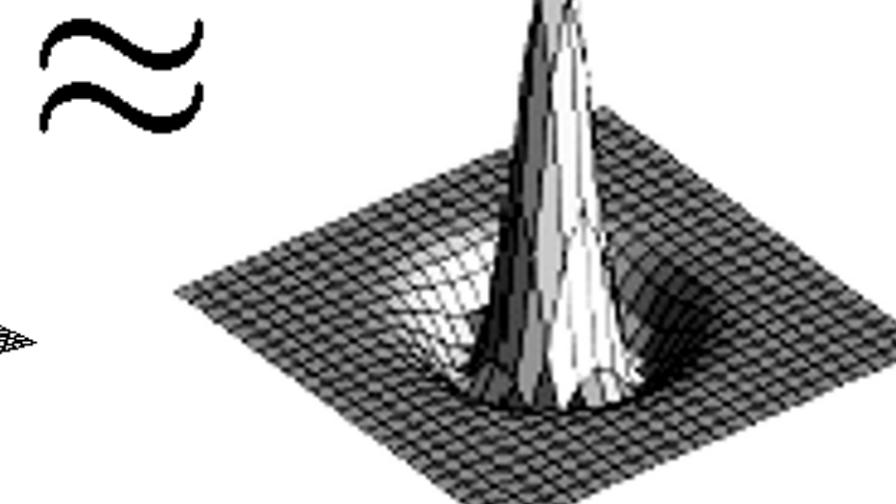
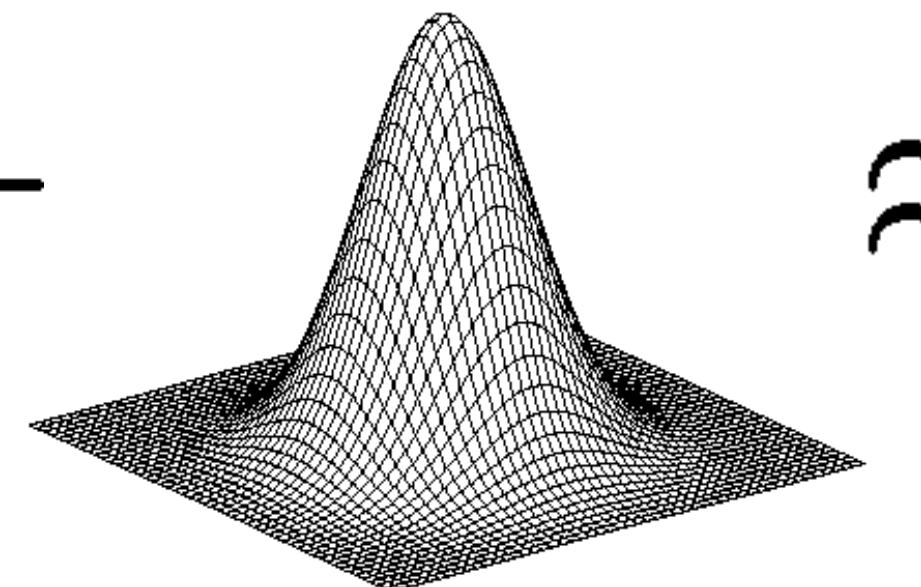
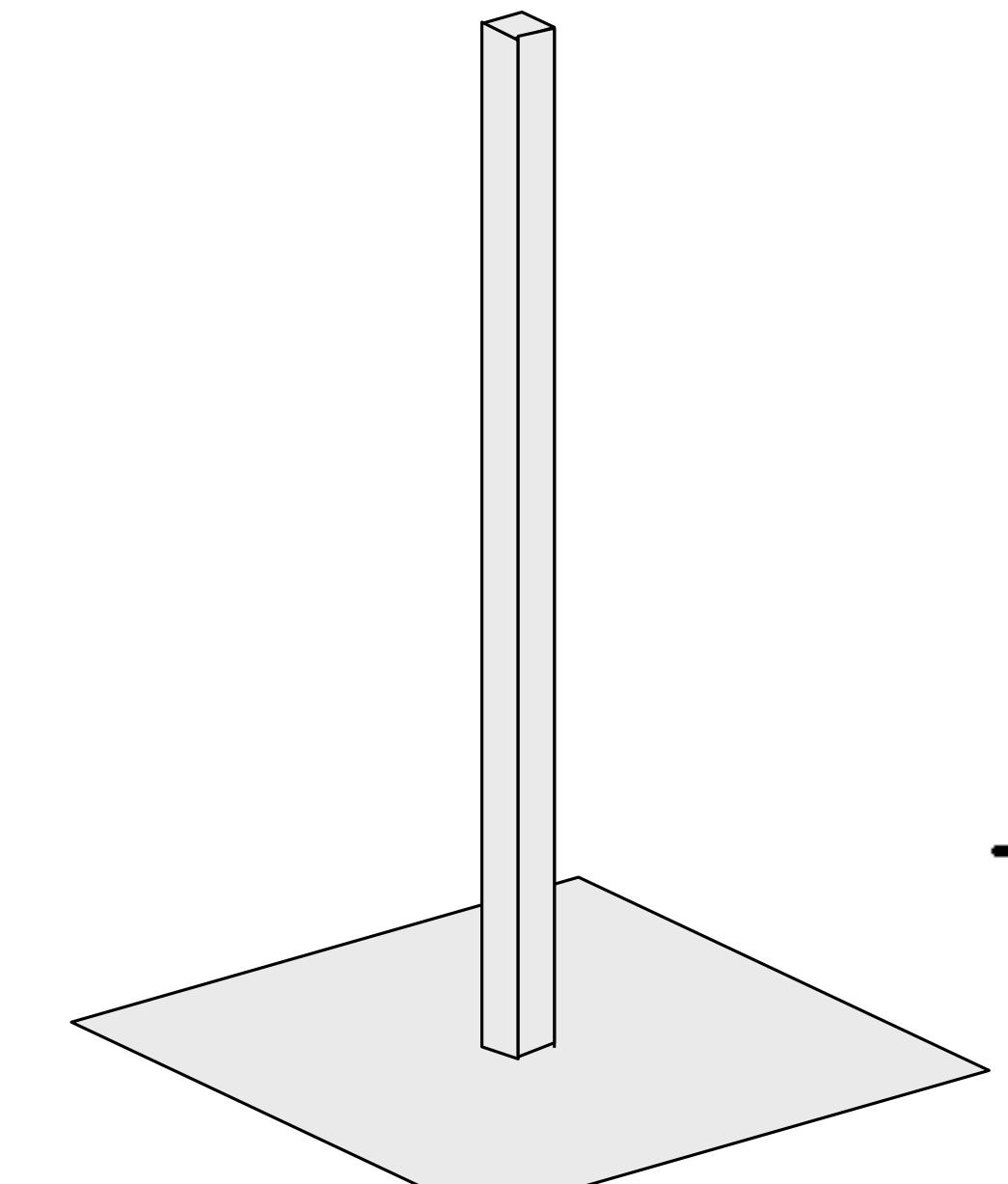


Photo credit: <https://www.flickr.com/photos/geezaweezer/16089096376/>

Sharpen filter

$$F + \alpha (F - \underbrace{F * H}_{\text{blurred image}}) =$$



↑
unit impulse
(identity kernel
with single 1 in
center, zeros
elsewhere)

Sharpen filter



“Optical” convolution

Camera shake



Source: Fergus, et al. “Removing Camera Shake from a Single Photograph”, SIGGRAPH 2006

Bokeh: Blur in out-of-focus regions of an image.



Source: https://www.diyphotography.net/diy_create_your_own_bokeh/

Filters: Thresholding



$$g(m, n) = \begin{cases} 255, & f(m, n) > A \\ 0 & otherwise \end{cases}$$

Can thresholding be implemented with a linear filter?