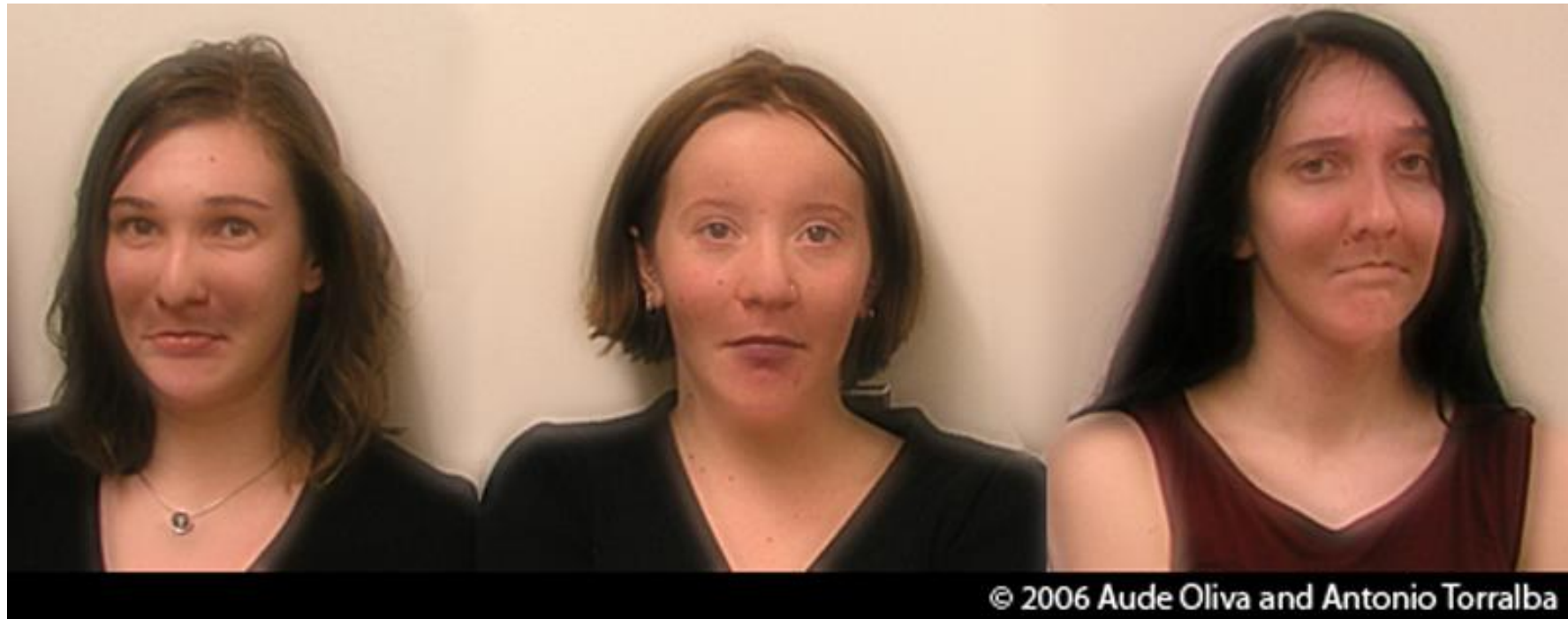


# CS5670: Intro to Computer Vision

Noah Snavely

## Lecture 1: Images and image filtering



Hybrid Images, Oliva et al., <http://olivalab.mit.edu/hybridimage.htm>

# CS5670: Intro to Computer Vision

Noah Snavely

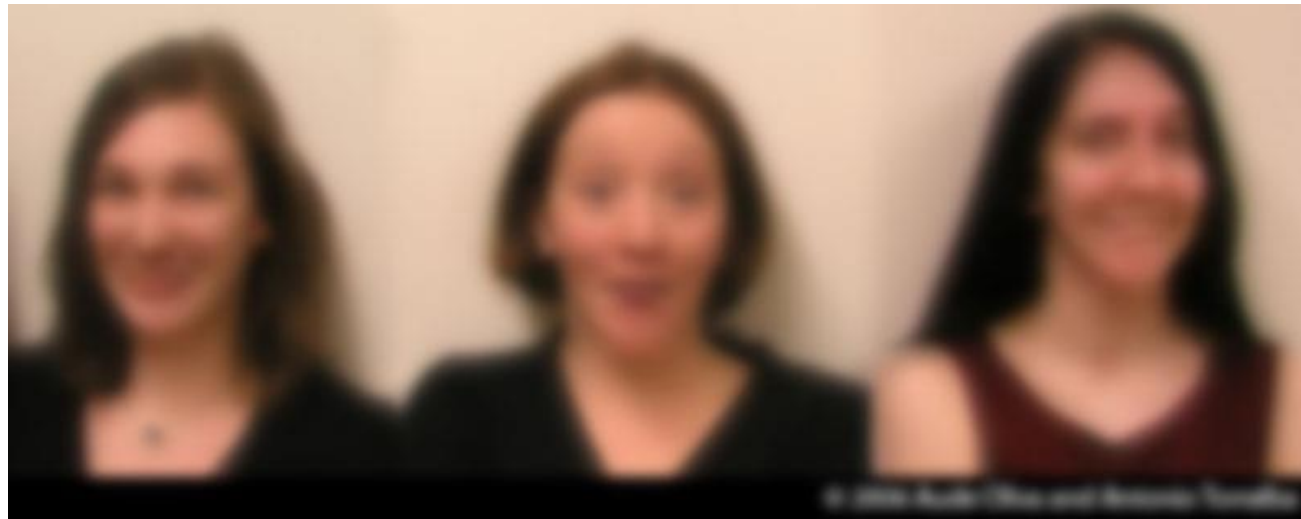
## Lecture 1: Images and image filtering



# CS5670: Intro to Computer Vision

Noah Snavely

## Lecture 1: Images and image filtering



Hybrid Images, Oliva et al., <http://olivalab.mit.edu/hybridimage.htm>

# Reading

- Szeliski, Chapter 3.1-3.3

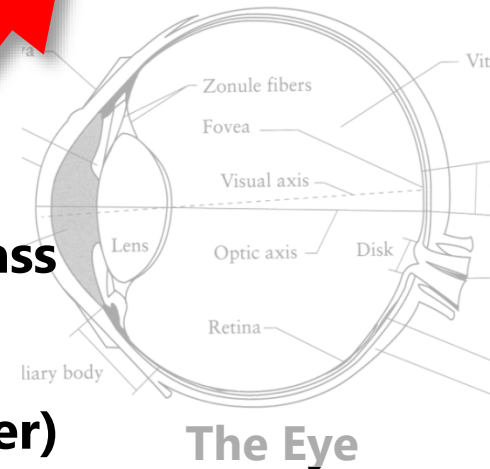
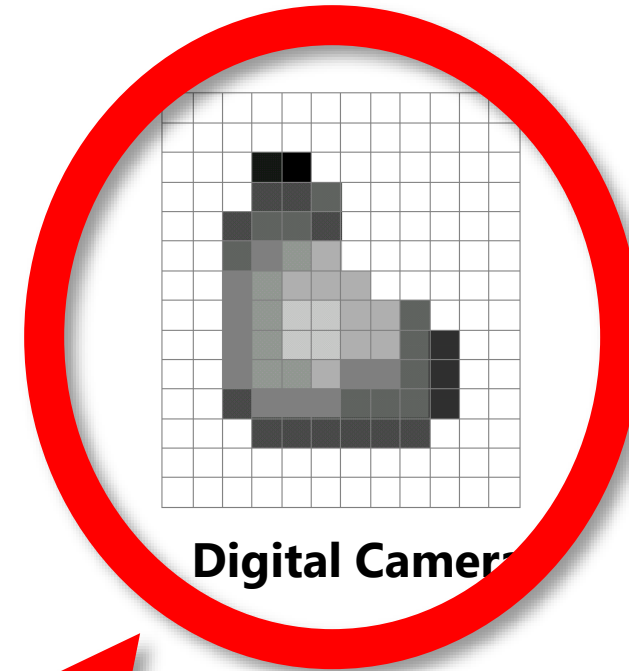
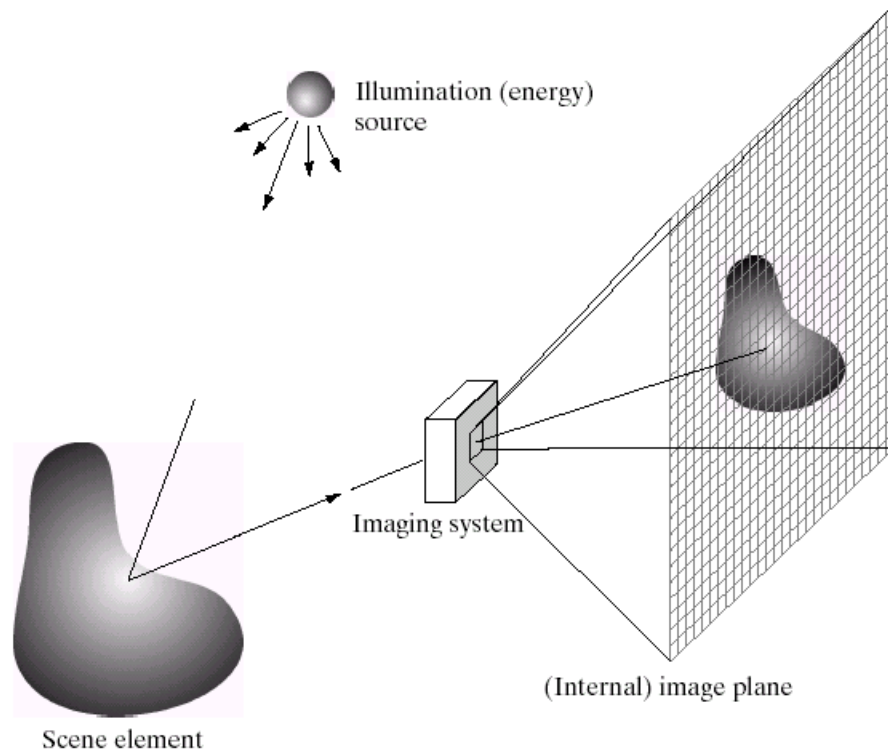
# Announcements

- Project 1 (Hybrid Images) will be released early next week
  - This project will be done solo
  - Other projects planned to be done in groups of 2
- Project is in Python – we will provide skeleton code and instructions for setting up a Python environment for the project

# What is an image?



# What is an image?

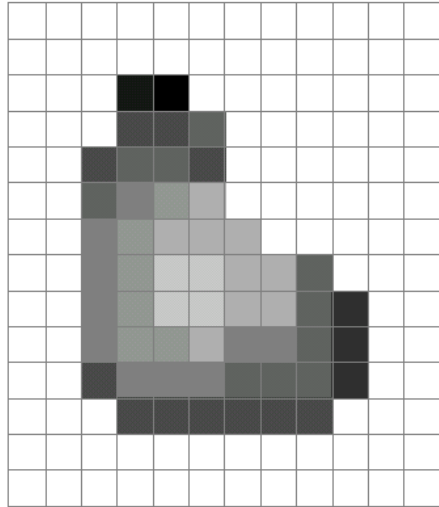


**We'll focus on these in this class**

**(More on this process later)**

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

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

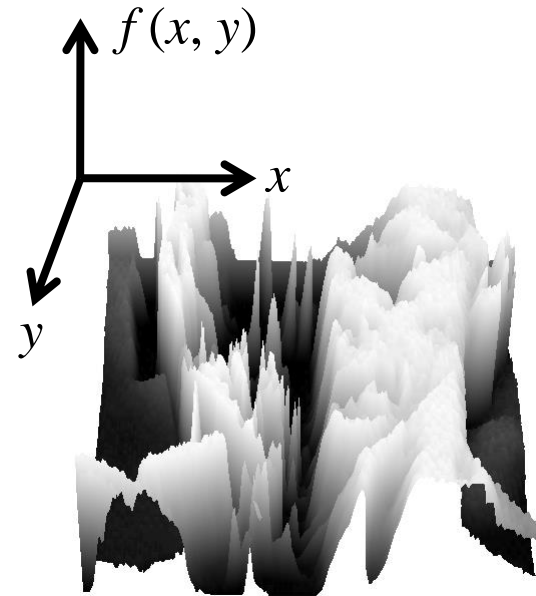


# What is an image?

- 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

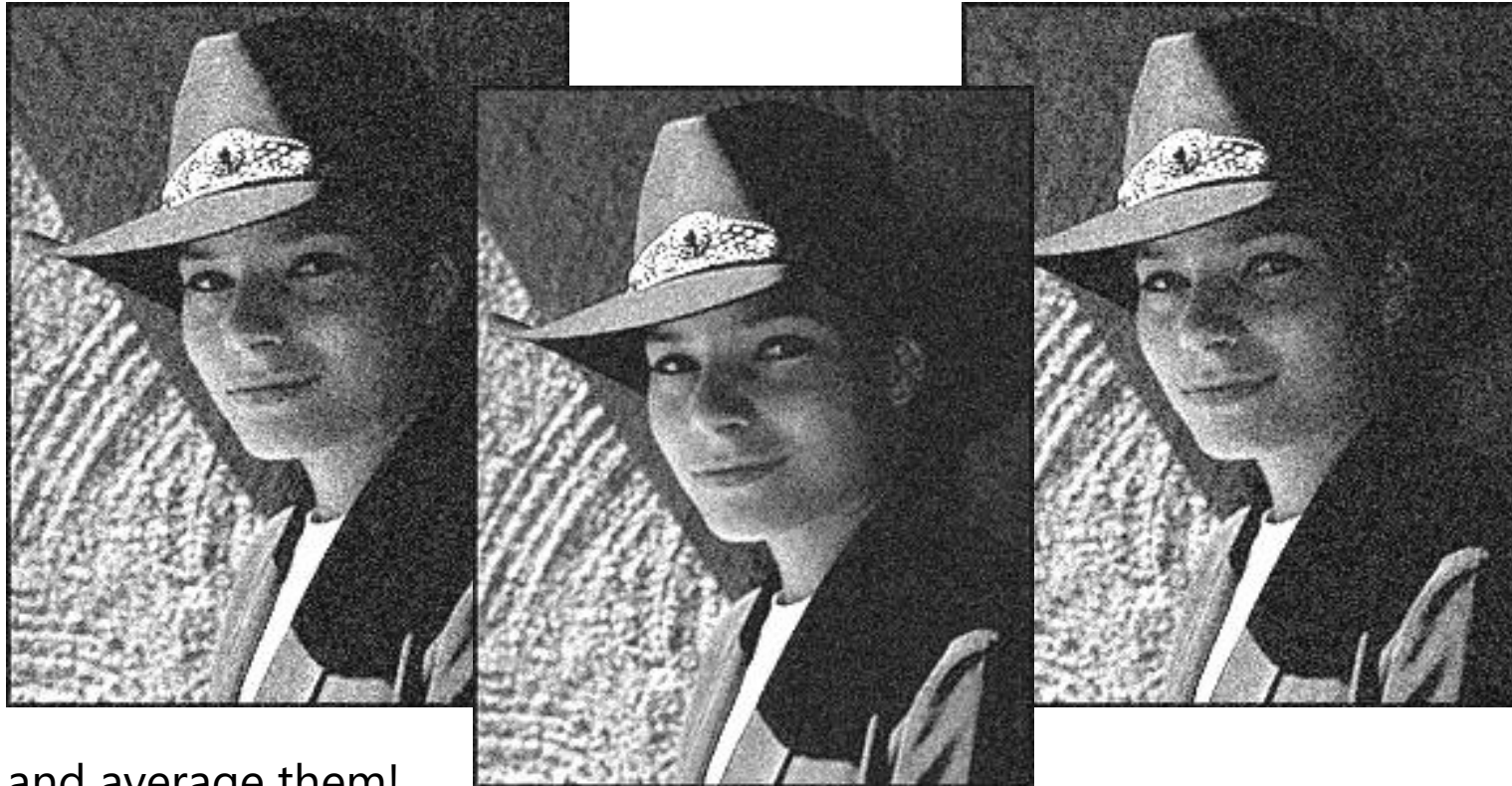
- Filtering
  - Form a new image whose pixel values are a combination of the original pixel values
- Why?
  - To get useful information from images
    - E.g., extract edges or contours (to understand shape)
  - To enhance the image
    - E.g., to remove noise
    - E.g., to sharpen and “enhance image”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

# Canonical Image Processing problems

- Image Restoration
  - denoising
  - deblurring
- Image Compression
  - JPEG, HEIF, MPEG, ...
- Computing Field Properties
  - optical flow
  - disparity
- 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?

# 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

Some function

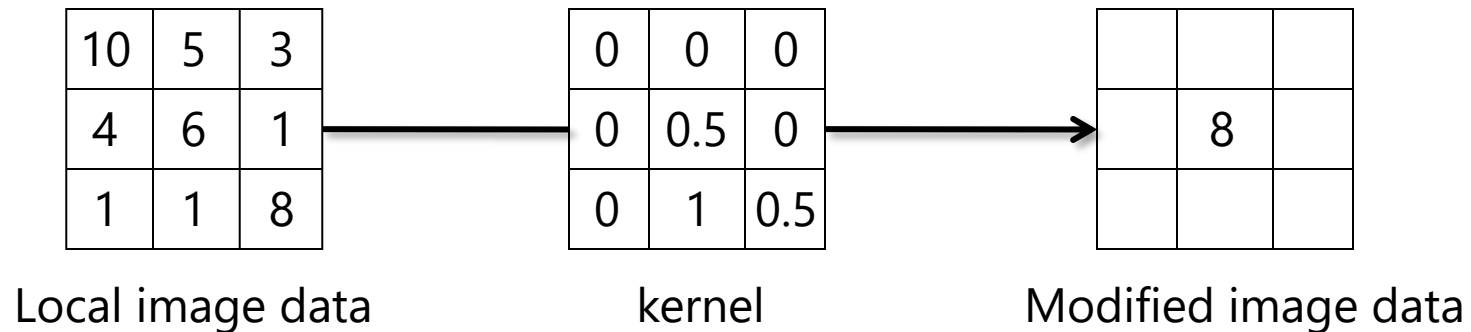


	7	

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”)



# Cross-correlation

Let  $F$  be the image,  $H$  be the kernel (of size  $2k+1 \times 2k+1$ ), and  $G$  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 **cross-correlation** operation:

$$G = H \otimes F$$

- Can think of as a “dot product” between local neighborhood and kernel for each pixel



# Convolution

- Same as cross-correlation, except that the kernel is “flipped” (horizontally and vertically)

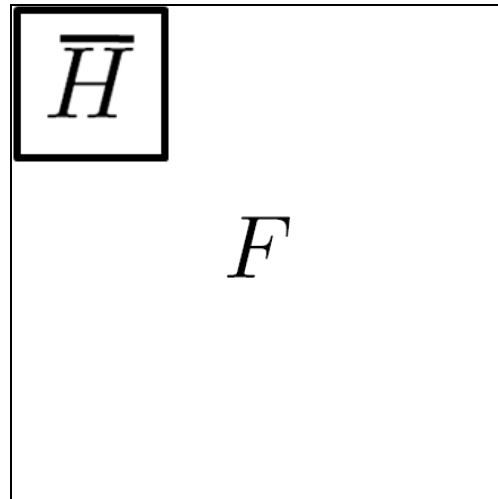
$$G[i, j] = \sum_{u=-k}^k \sum_{v=-k}^k H[u, v] F[i - u, j - v]$$

This is called a **convolution** operation:

$$G = H * F$$

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

# Convolution



















# Linear filters: examples



Original



0	0	0
0	1	0
0	0	0

slido



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

① Start presenting to display the poll results on this slide.

# Linear filters: examples



Original



0	0	0
0	1	0
0	0	0



Identical image

# Linear filters: examples



Original



0	0	0
1	0	0
0	0	0

slido



**What image operation does filtering with this kernel perform? ( $\begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$ )**

① Start presenting to display the poll results on this slide.

# Linear filters: examples



Original



0	0	0
1	0	0
0	0	0



Shifted left by 1 pixel

# Linear filters: examples



Original



$\frac{1}{9}$

1	1	1
1	1	1
1	1	1



Blur (with a mean filter)



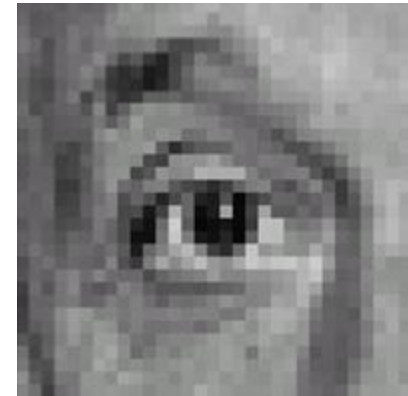
# Linear filters: examples



Original

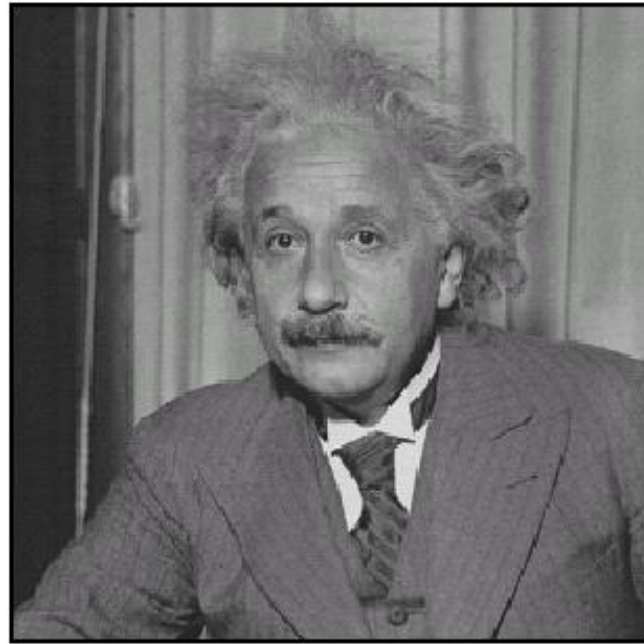


$$\left( \begin{array}{|c|c|c|} \hline 0 & 0 & 0 \\ \hline 0 & 2 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array} - \frac{1}{9} \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array} \right)$$

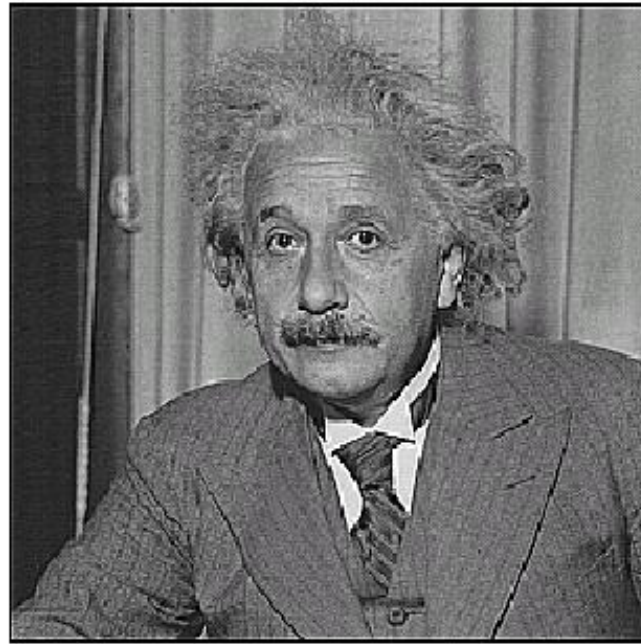


**Sharpening filter**  
(accentuates edges)

# Sharpening

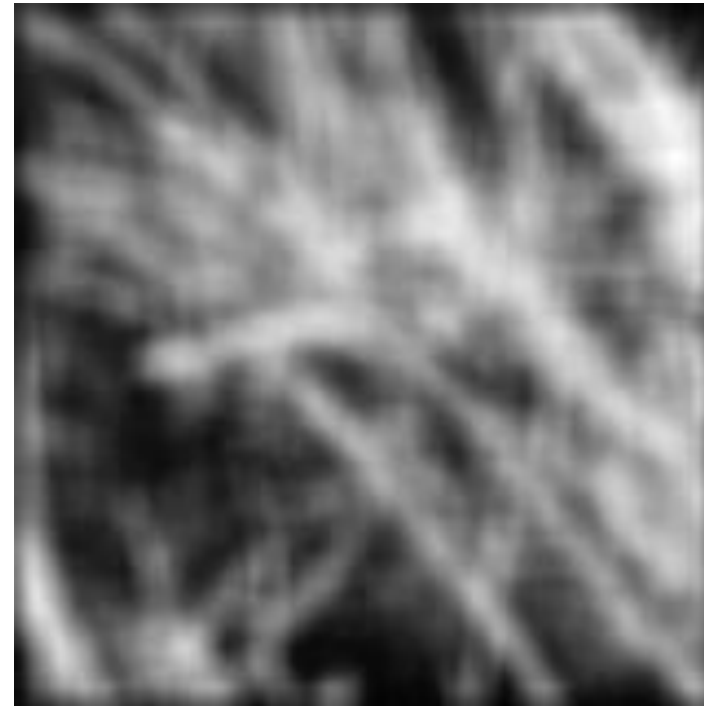


before

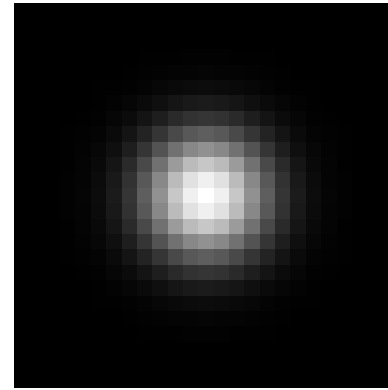
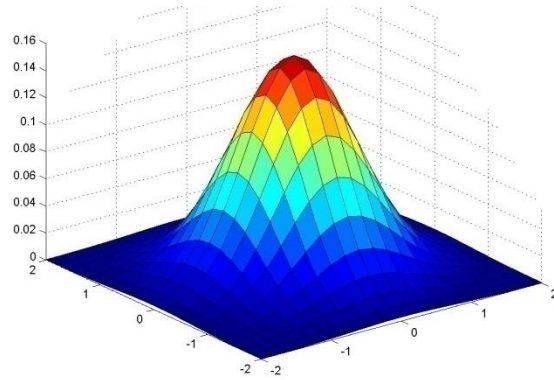


after

# Smoothing with box filter revisited

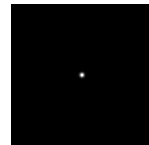
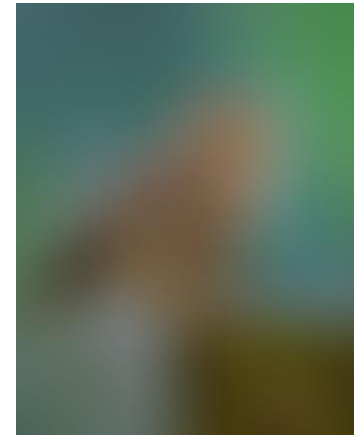
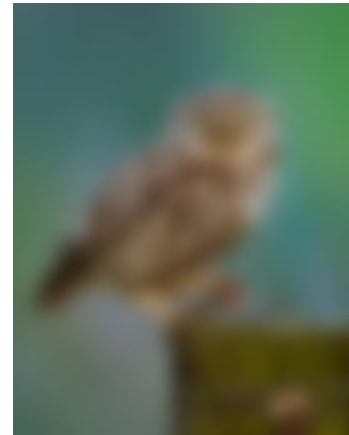


# Gaussian kernel

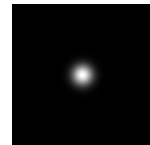


$$G_{\sigma} = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

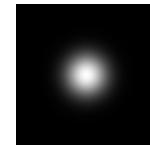
# Gaussian filters



$\sigma = 1$  pixel



$\sigma = 5$  pixels

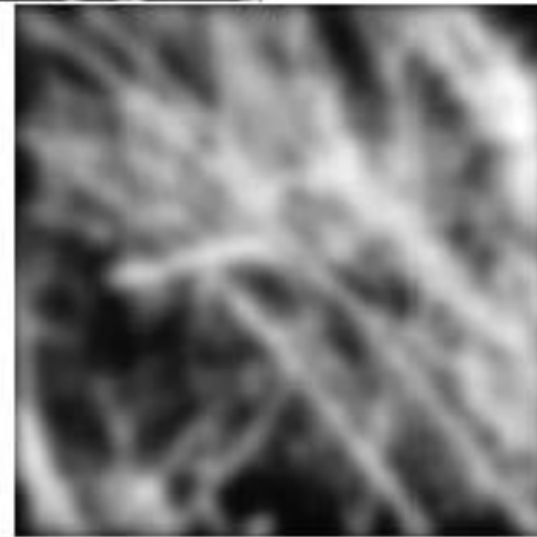
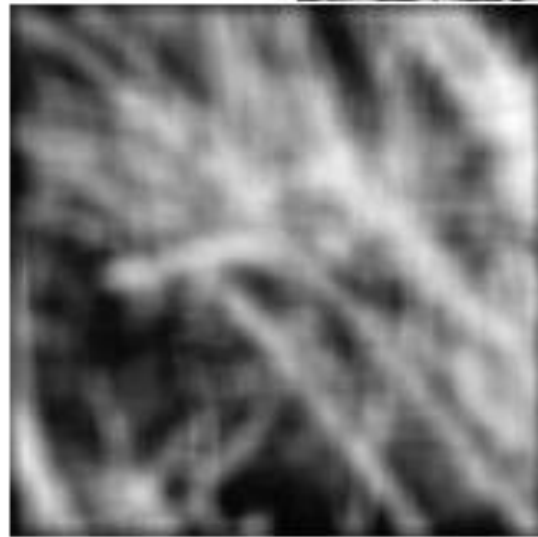


$\sigma = 10$  pixels



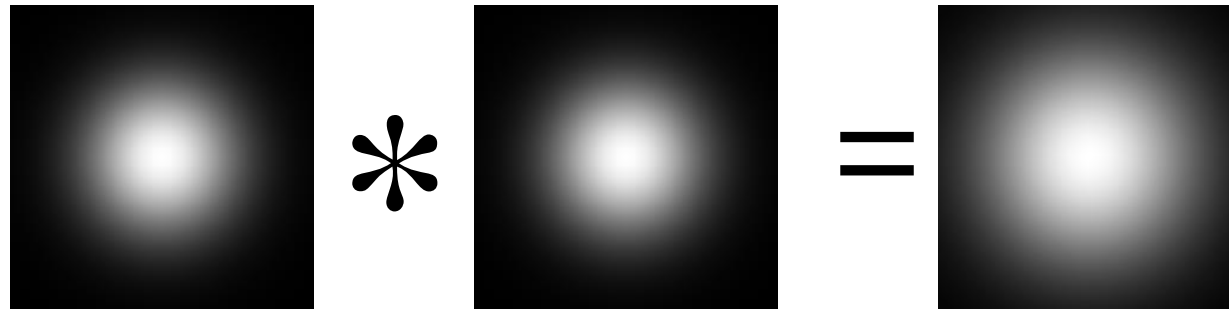
$\sigma = 30$  pixels

# Mean vs. Gaussian filtering



# Gaussian filter

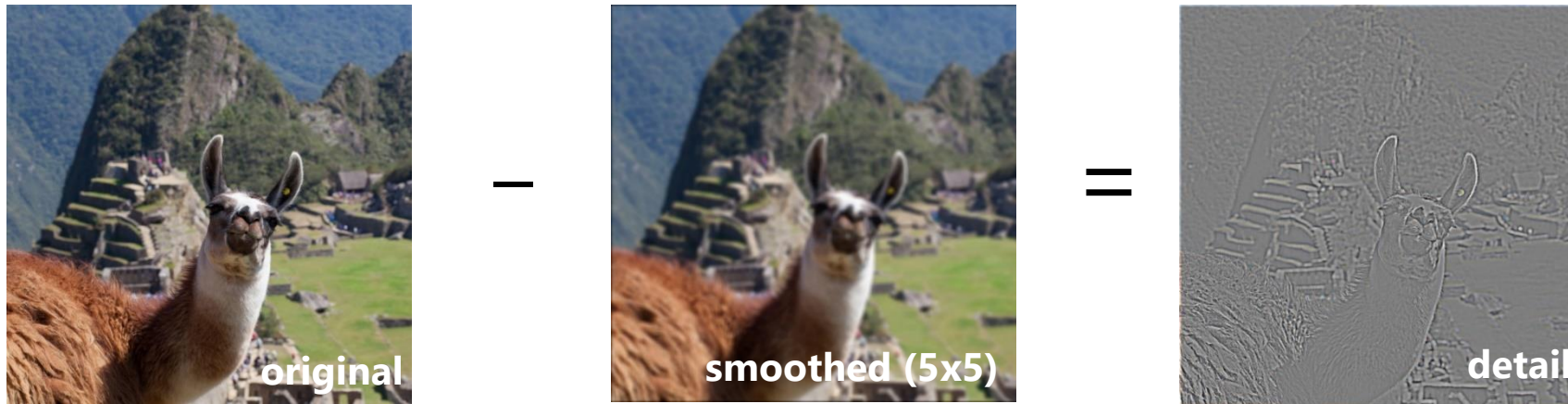
- Removes “high-frequency” components from the image (low-pass filter)
- Convolution with self is another Gaussian



- Convolving twice with Gaussian kernel of width  $\sigma$   
= convolving once with kernel of width  $\sigma\sqrt{2}$

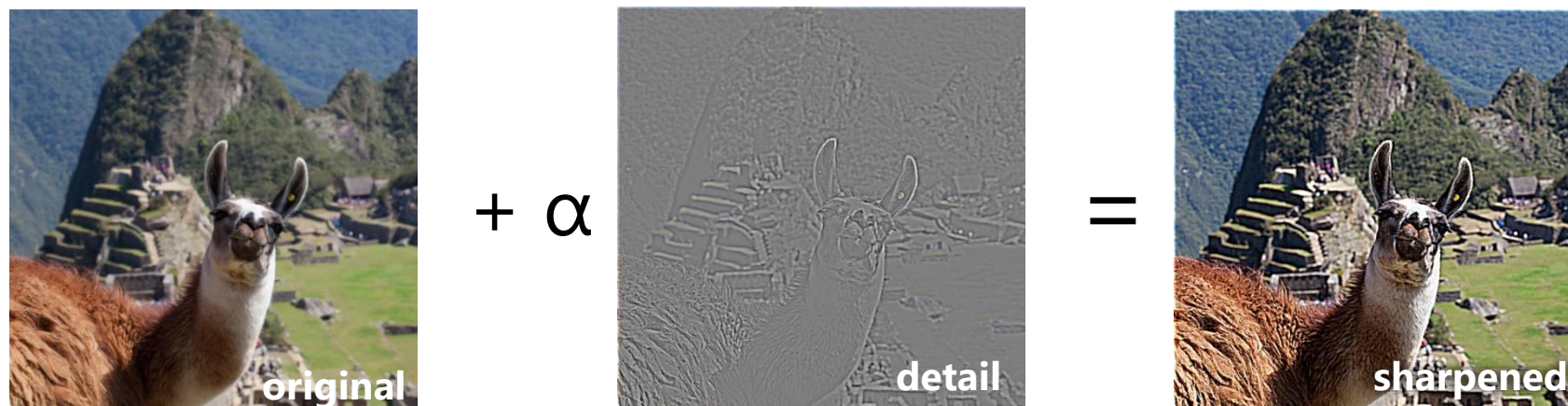
# Sharpening revisited

- What does blurring take away?



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

Let's add it back:

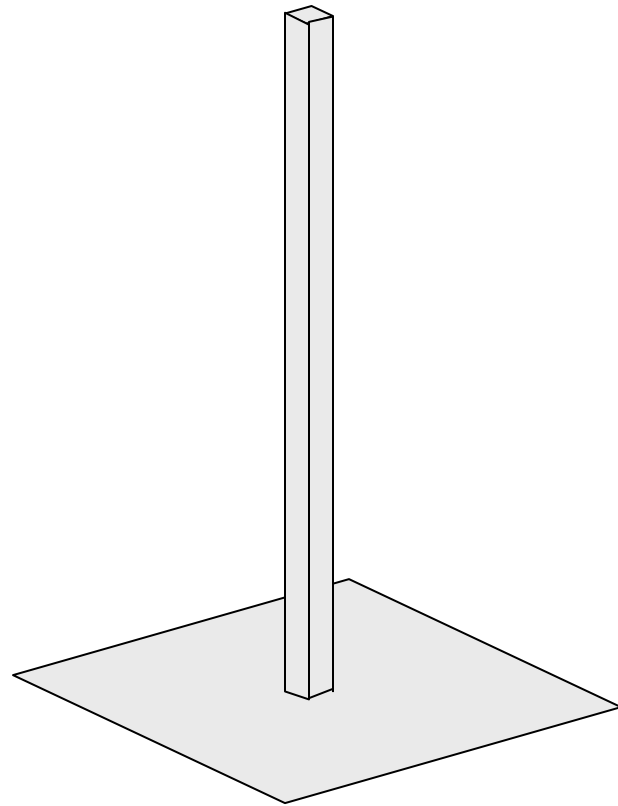




# Sharpen filter

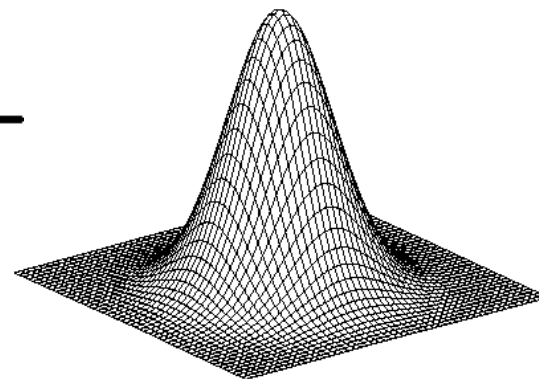
$$F + \alpha (F - \underbrace{F * H}_{\text{blurred image}}) =$$

↑  
image



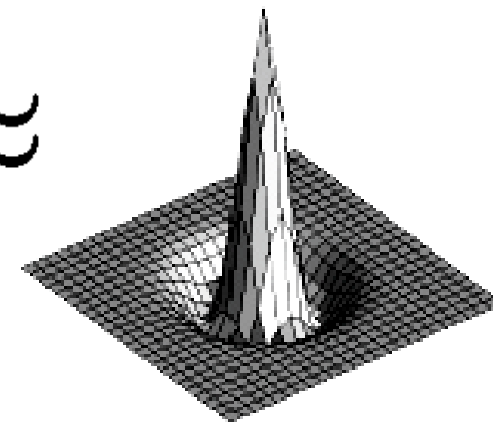
scaled impulse

—



Gaussian

≈



Sharpen filter

↑  
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/](https://www.diyphotography.net/diy_create_your_own_bokeh/)

# Filters: Thresholding



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

# Linear filters

- Can thresholding be implemented with a linear filter?

slido



**Can thresholding be implemented with a linear filter?**

① Start presenting to display the poll results on this slide.

**Questions?**