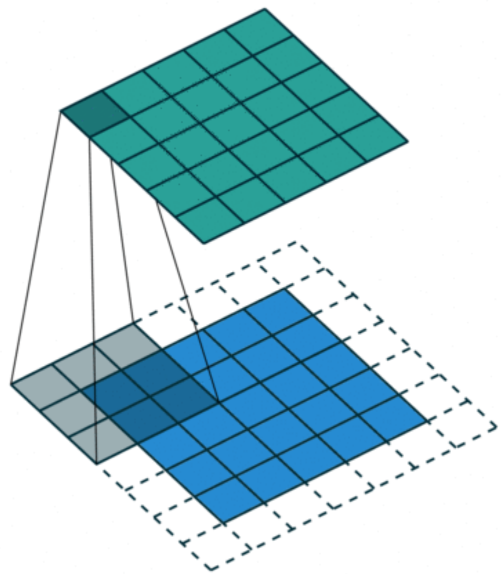
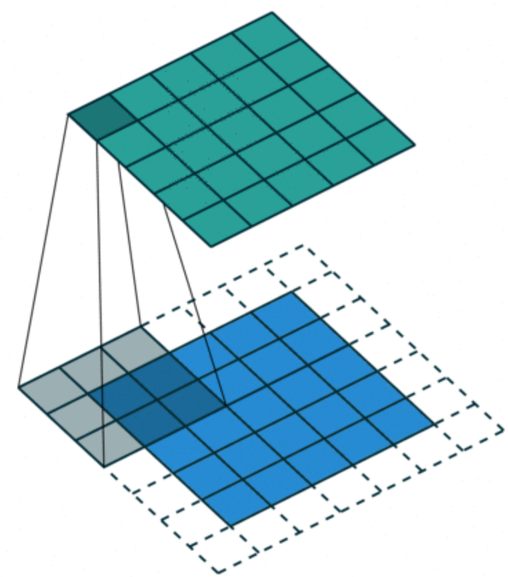
CSCI 497P/597P: Computer Vision



Lecture 3: Convolution and its Properties

CSCI 497P/597P: Computer Vision



Lecture 3: Convolution and its Properties

Announcements

- HW1 is out today
  - Covers filtering and convolution
  - Due in 1 week (10pm next Tuesday)

Goals

- Understand the distinction between cross-correlation and convolution.
- Know the properties of cross-correlation and convolution:
  - Linearity and shift-invariance (both)
    Associativity and commutativity (convolution only)
- Understand the design of several common image filters:
  - Box blur and Gaussian blur
  - Sharpening
- Understand the limitations of linear filtering

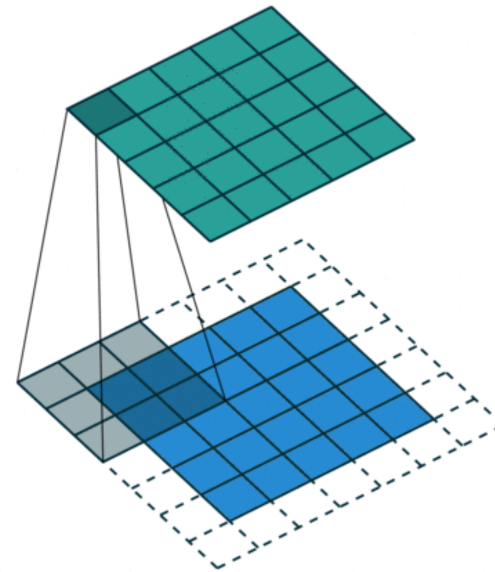
Computing Cross-Correlation

$$g = f \otimes w$$

output image

input image

weights, or
filter, or
kernel



```
for x = 0 to w:
  for y = 0 to h:
    for i in -k to k:
      for j in -k to k:
        out[x,y] += w[i,j] * in[x+i, y+j]
```

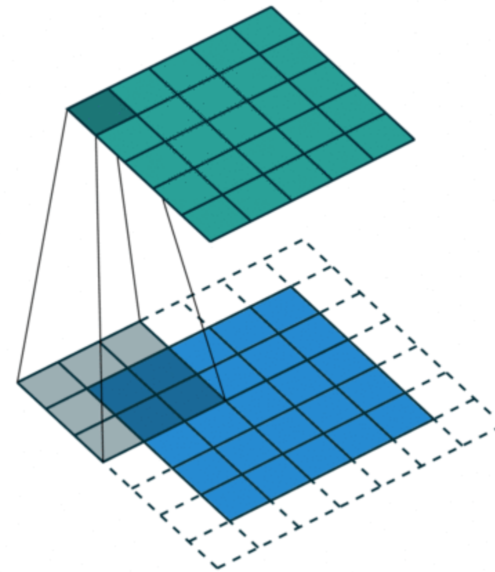
Computing Cross-Correlation

$$g = f \otimes w$$

weights, or
filter, or
kernel

output image

input image



```
for x = 0 to w:
  for y = 0 to h:
    for i in -k to k:
      for j in -k to k:
        out[x,y] += w[i,j] * in[x+i, y+j]
```

A bit of practice

In groups: work on Problems #1-4

- Problems are linked from the course webpage on the Schedule table

- Write answers in your Google Doc (pinned in group Discord channels)

$$\begin{bmatrix} 3 & 6 & 3 \\ 4 & 8 & 4 \\ 3 & 6 & 3 \end{bmatrix} \quad \begin{bmatrix} 4 & 8 & 4 \\ 4 & 8 & 4 \\ 4 & 8 & 4 \end{bmatrix} \quad [8]$$

$$\begin{matrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{matrix}$$

$$\begin{array}{ccc} & \begin{matrix} 0 & 1 & 0 \end{matrix} & \\ \begin{matrix} 0 \\ 0 \\ 0 \end{matrix} & \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix} & \begin{matrix} 0 \\ 0 \\ 0 \end{matrix} \\ & \begin{matrix} 0 & 1 & 0 \end{matrix} & \end{array}$$

$$\begin{bmatrix} 3 & 6 & 3 \\ 4 & 8 & 4 \\ 3 & 6 & 3 \end{bmatrix}$$

Questions remain

- What happens at the edges?
- What properties does this operator have?
- What can and can't this operator do?

A shift filter

Cross-correlate the image *f* with the kernel *w*.

Use "same" output size, with zero-padding for out-of-bounds values.

$$\underset{f}{\begin{bmatrix}1&2&1\\1&3&1\\0&1&3\end{bmatrix}} \otimes \underset{w}{\begin{bmatrix}0&0&0\\0&0&1\\0&0&0\end{bmatrix}} = \begin{bmatrix}\ &\ &\ \\ \ &\ &\ \\ \ &\ &\ \end{bmatrix}$$

Cross-correlation vs Convolution

- Cross-correlation: $g = f \otimes w$

$$g(x,y) = \sum_{i=-k}^{k} \sum_{j=-k}^{k} w(i,j) f(x+i, y+j)$$

- Convolution: $g = f * w$

$$g(x,y) = \sum_{i=-k}^{k} \sum_{j=-k}^{k} w(i,j) f(x-i, y-j)$$

These are related: $f * w = f \otimes \text{fliphorz}(\text{flipvert}(w))$

Properties

Assume: f is an image; w and v are filters; s, t are scalars.

filter img = shift img, filter shift result

Shift invariance (both) $f(x, y) \otimes w = [f(x - s, y - t) \otimes w)](x - s, y - t)$

Linearity (both) *and*

$$(f \otimes w) + (f \otimes v) = f \otimes (w + v)$$

$$(f \otimes sw) = s(f \otimes w)$$

Commutativity (conv only) $f * w = w * f$

$f \otimes w \neq w \otimes f$

expensive

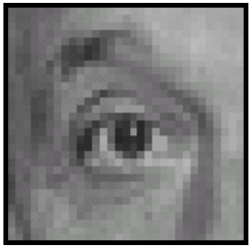
expensive!

Associativity (conv only) $(f * w) * v = f * (w * v)$

cheap!

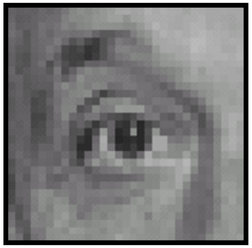
What can we do with this?

What can we do with this?

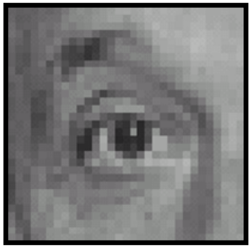


$$\otimes \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} =$$

What can we do with this?

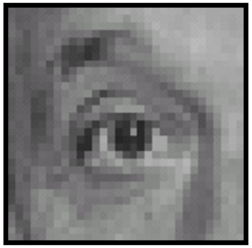


$$* \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} =$$



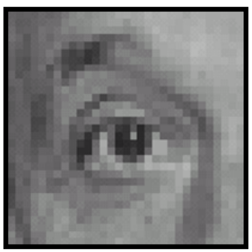
Identity filter: output = input

What can we do with this?

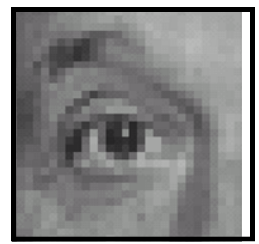


$$* \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} =$$

What can we do with this?

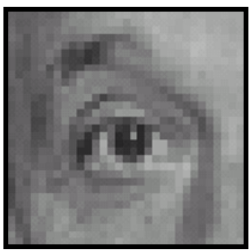


$$*\;\begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}\;=$$



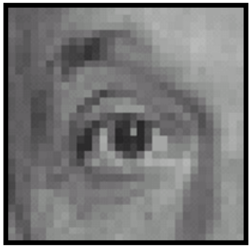
left shift

What can we do with this?

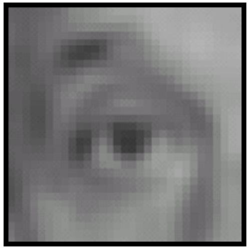


$$* \frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix} =$$

What can we do with this?

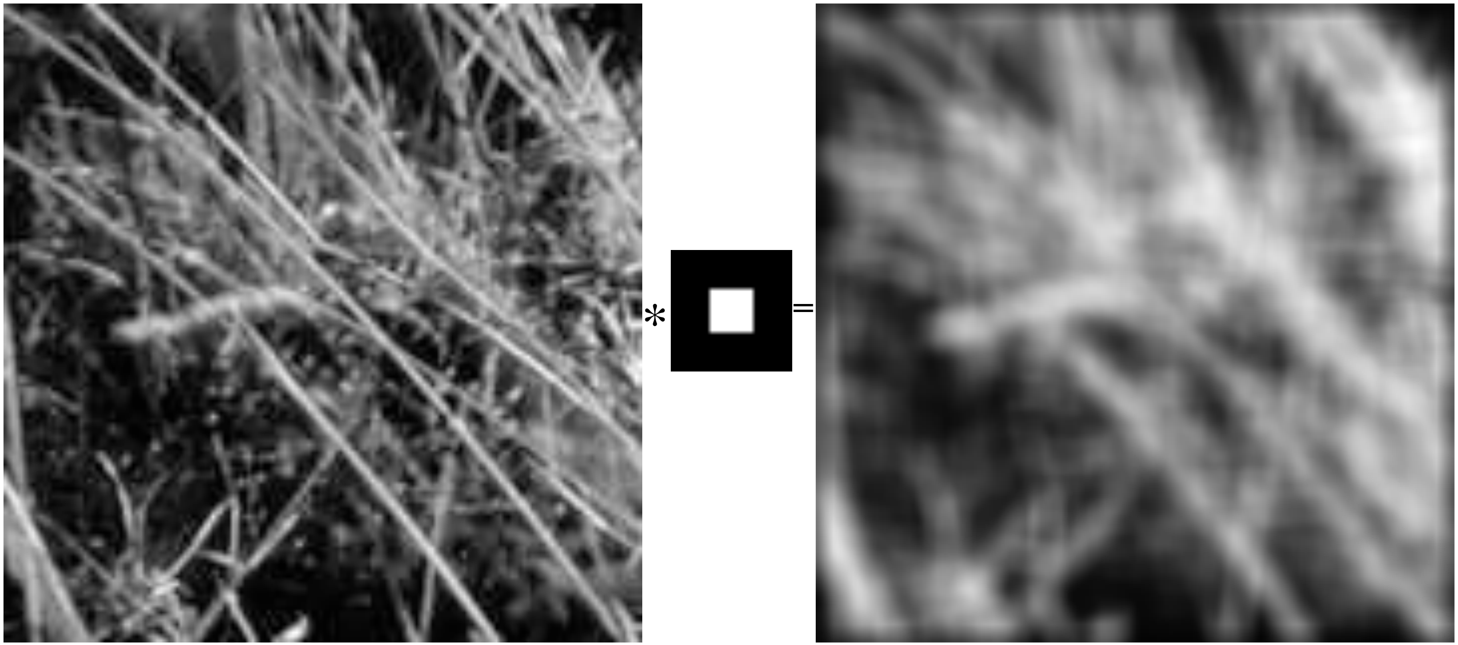


$$* \ \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} =$$



mean filter, or
box blur

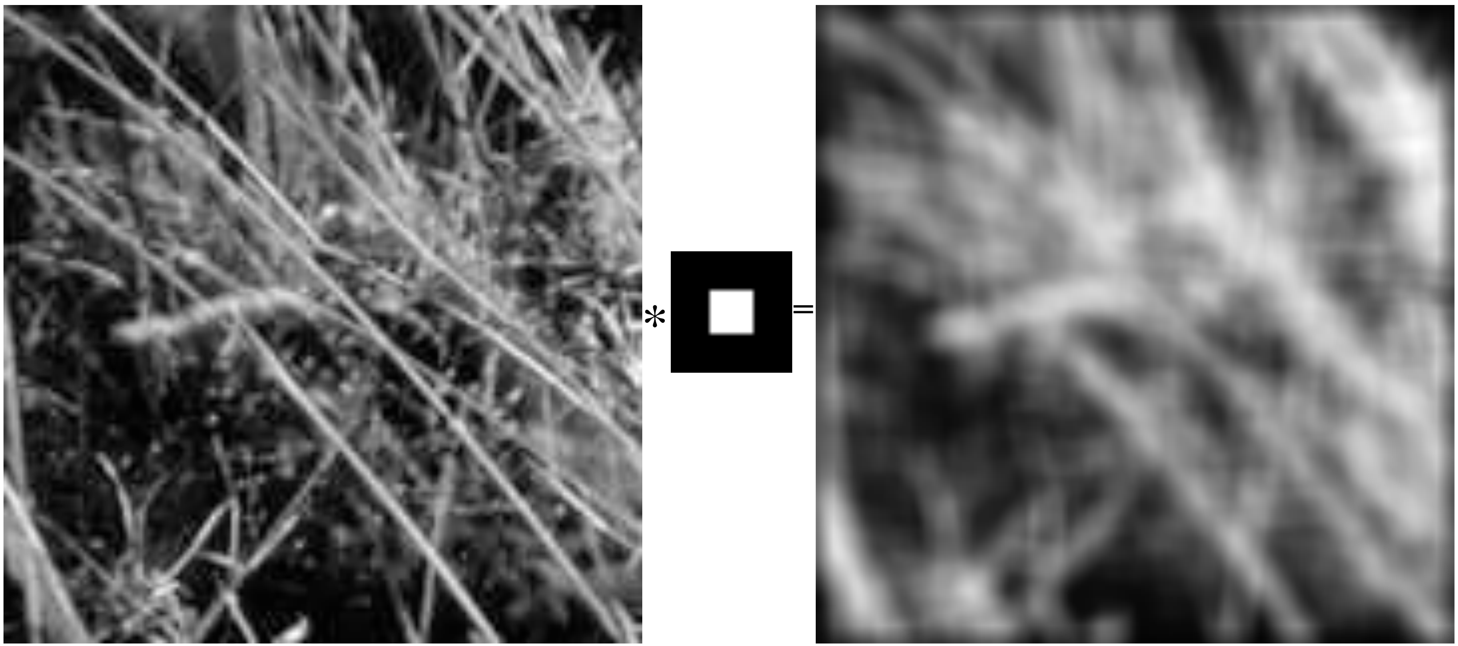
Blurring: Another example



Notice: lattice-like texture

What motivated the mean filter?

Blurring: Another example



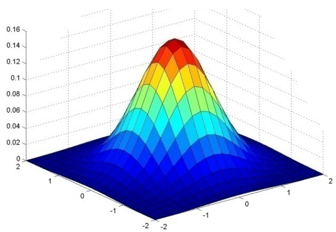
Notice: lattice-like texture

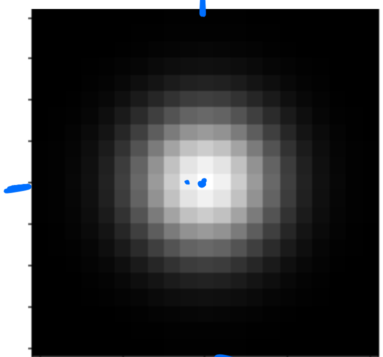
What motivated the mean filter?

Idea: the closer the pixel, the more likely it is to be similar

Gaussian Blur

- Idea: weight closer pixels more heavily using a Gaussian kernel:



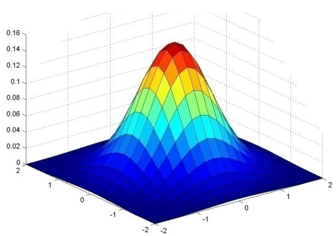


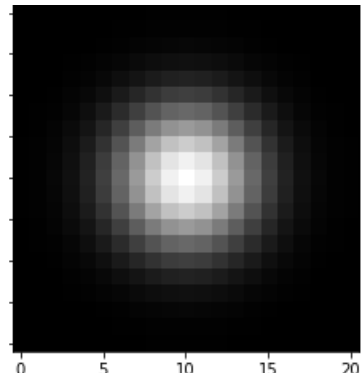
This is a bivariate (2D) Gaussian function:

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

Gaussian Blur

- Idea: weight closer pixels more heavily using a Gaussian kernel:





$\frac{1}{16}$

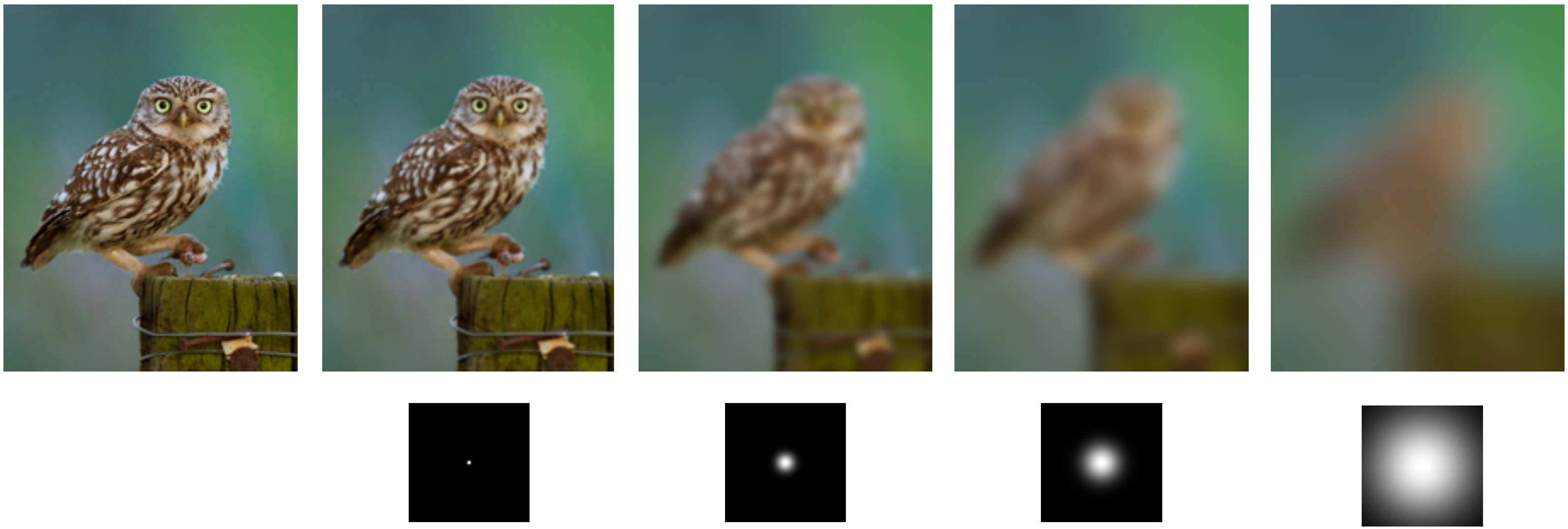
1	2	1
2	4	2
1	2	1

3x3 approximation

This is a bivariate (2D) Gaussian function:

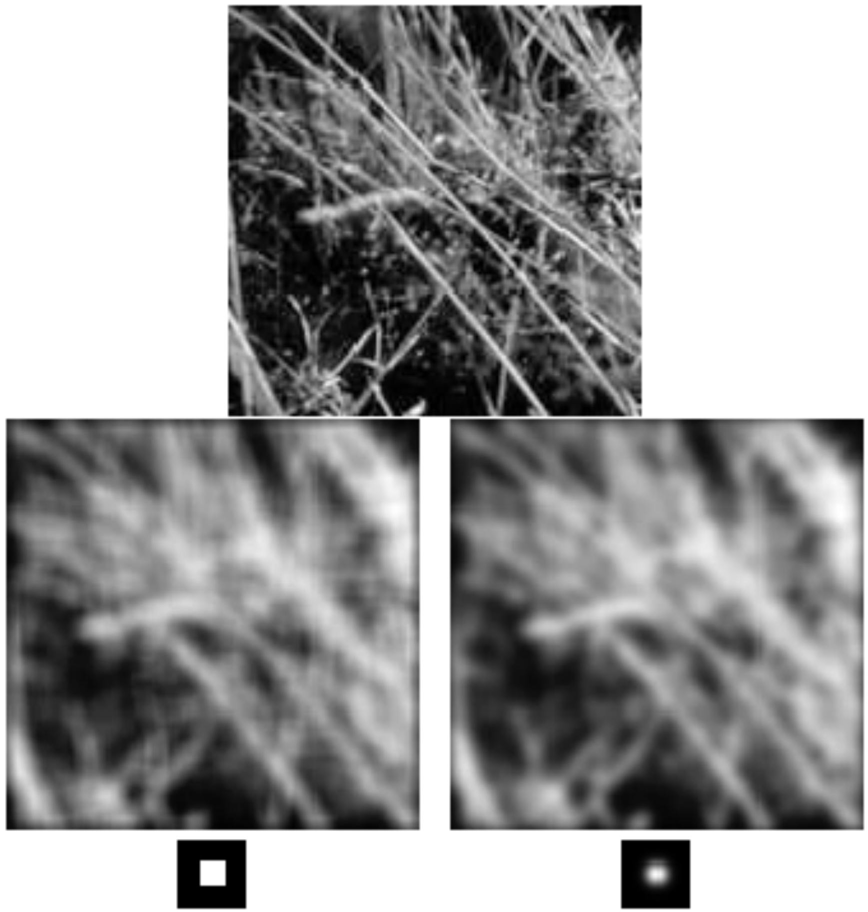
$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

Gaussian Filters



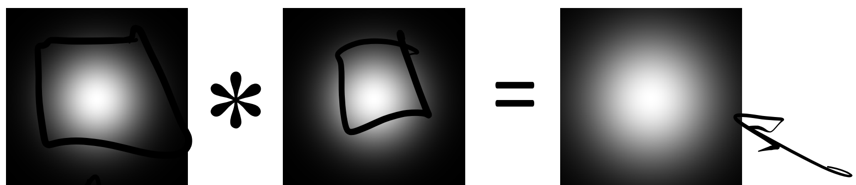
σ = 1 pixel σ = 5 pixels σ = 10 pixels σ = 30 pixels

Mean vs. Gaussian



Composing Filters

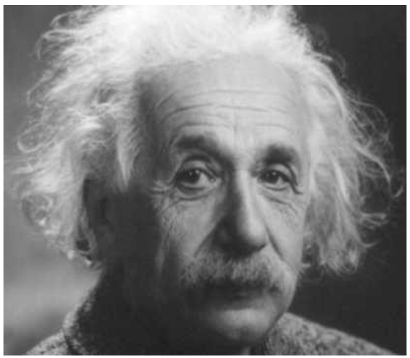
- Recall associativity:



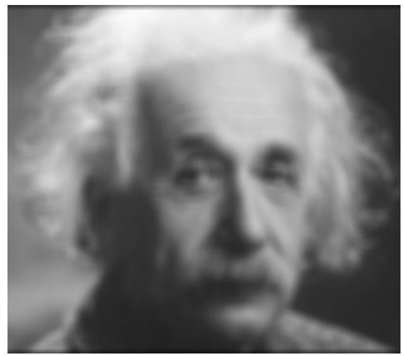
$$G_\sigma * G_\sigma = G_{\sqrt{2}\sigma}$$

Sharpening!?

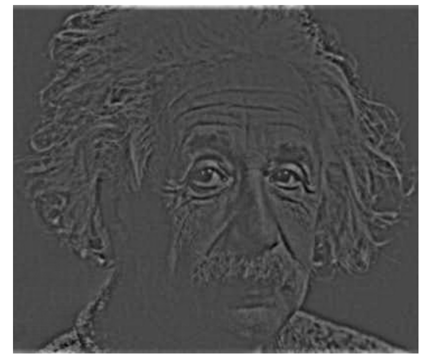
- What gets removed when we blur?



−



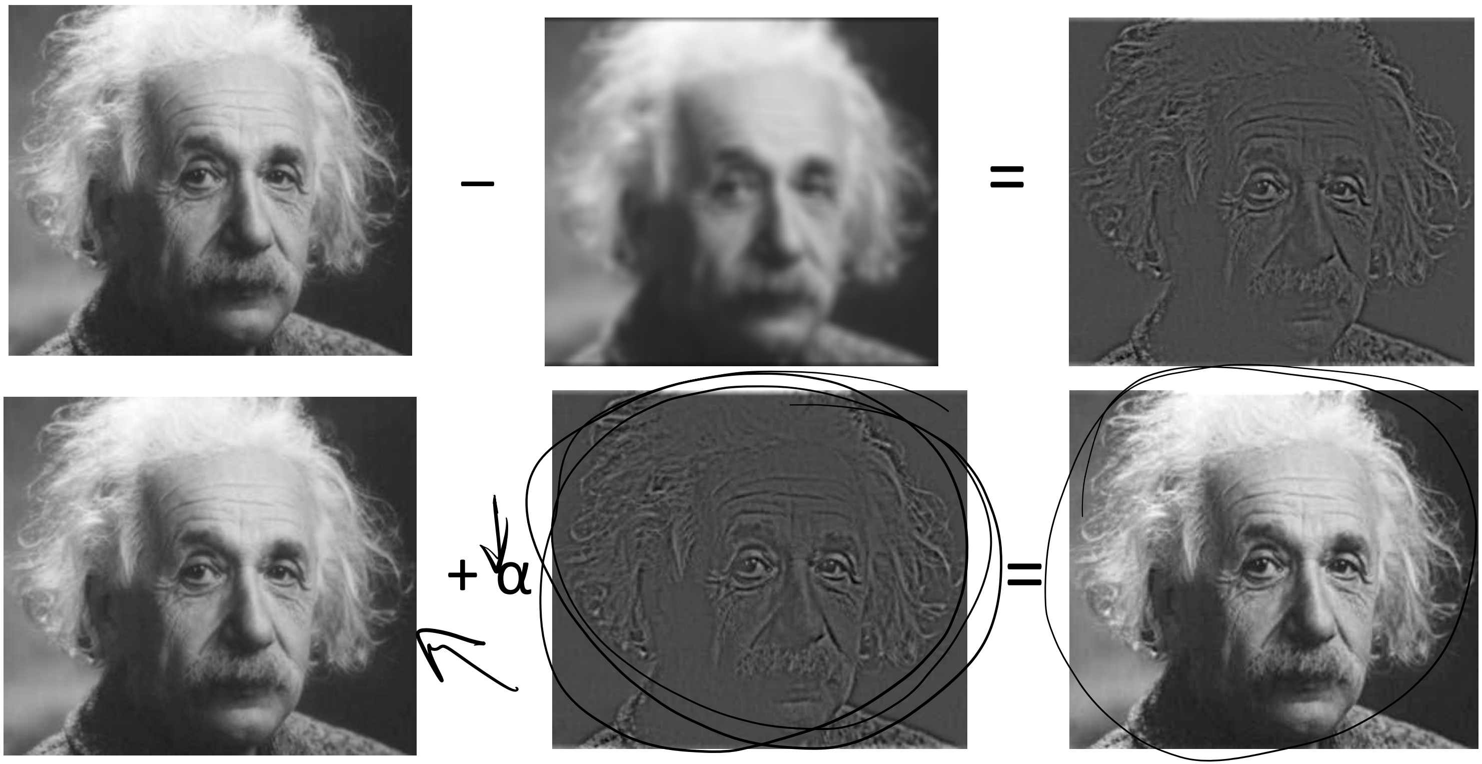
=



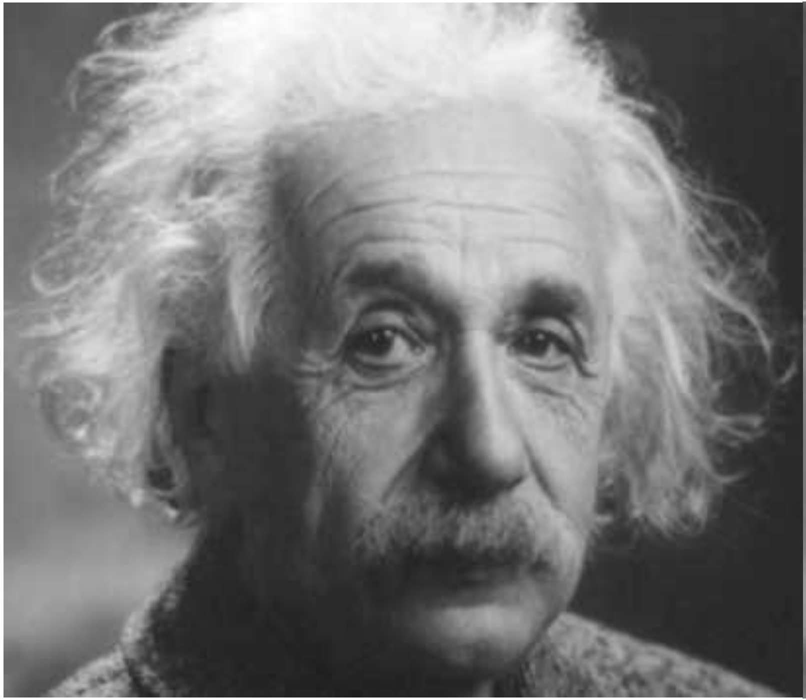
$f \quad - \quad f * B$

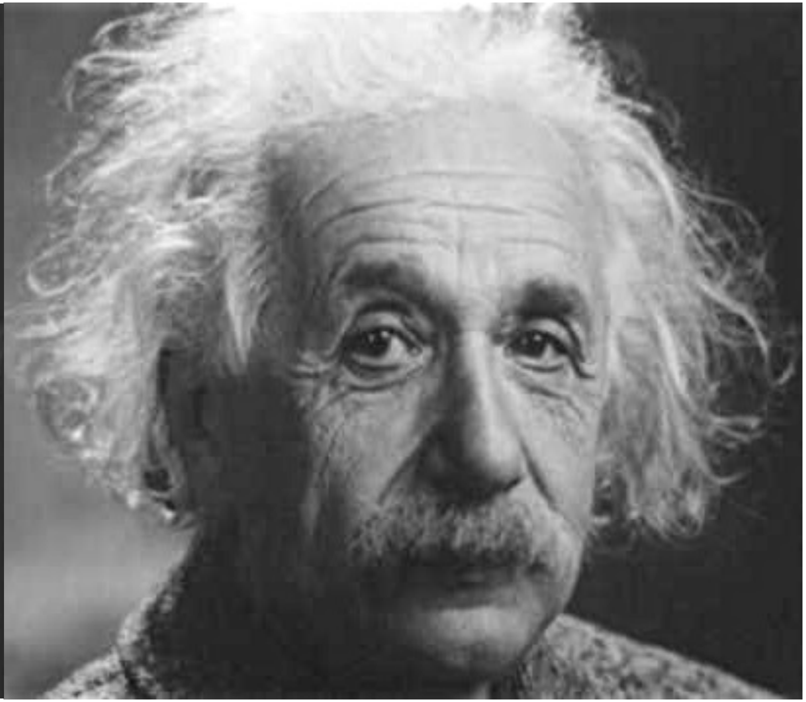
Sharpening!?

- What gets removed when we blur?

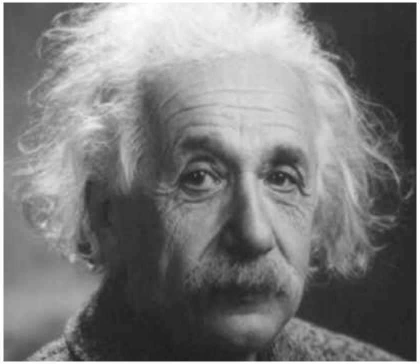
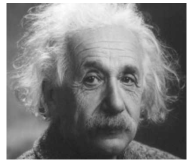
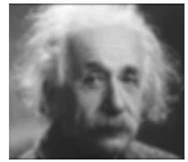
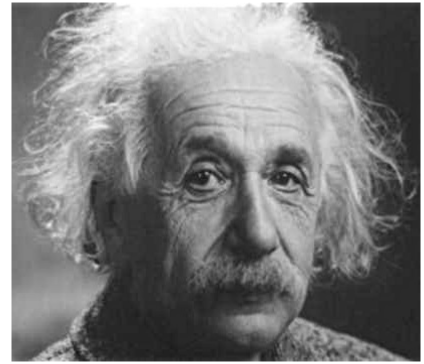


Sharpening


Before


After

Sharpening: once more with mathing

 + ( - ) = 

image

blur filter

$$f + f - (f * B)$$

introduce identity filter

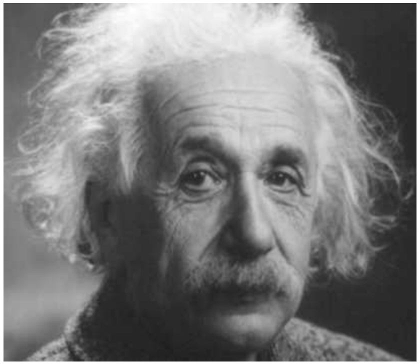
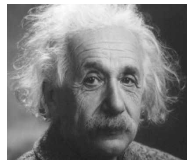
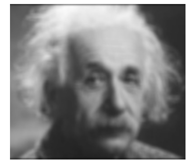
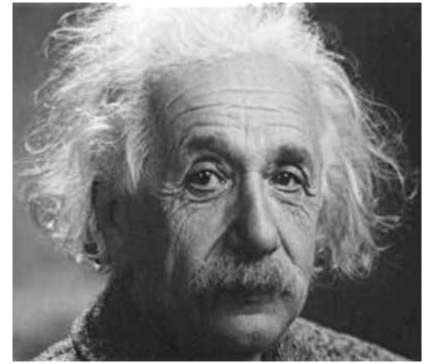
$$2f - f * B$$

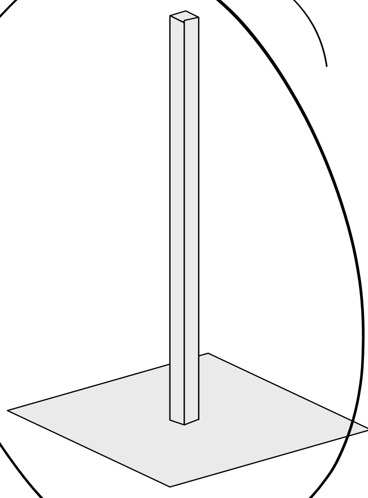
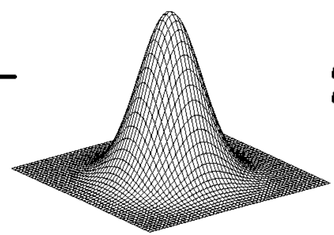
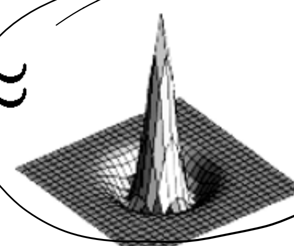
$$(f * 2I) - (f * B)$$

use linearity

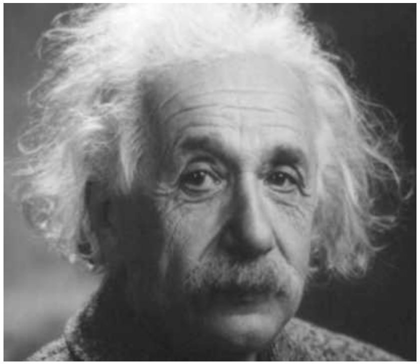
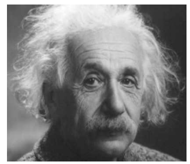
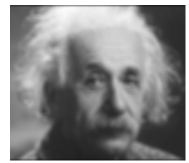
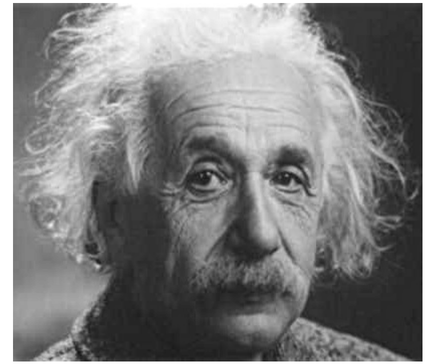
$$\boxed{f * \underbrace{(2I - B)}_{\text{Sharpening filter}}}$$

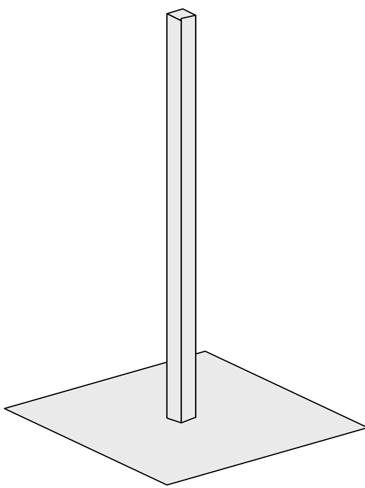
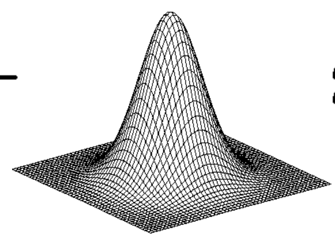
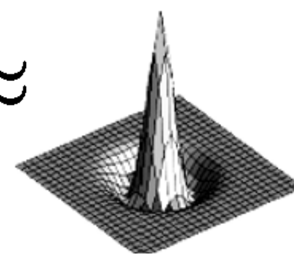
Sharpening: once more with mathing


+ (

-

) =



—

≈


Sharpening: once more with mathing

 + ( - ) = 

 —  ≈ 

0	-1	0
-1	5	-1
0	-1	0

Possible 3x3 Approximation

Effects of Sharpening

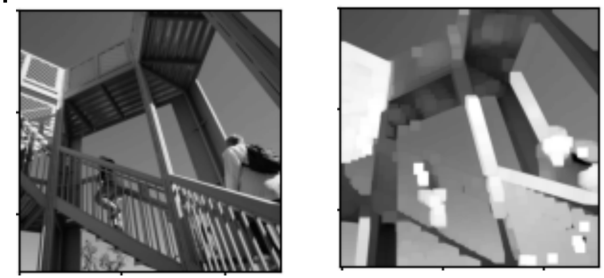
Orig

Sharpened

Limitations:
What *can't* convolution do?

Problem #5 - discuss in groups.

- Maximum filter?



- Threshold?

- y partial derivative?