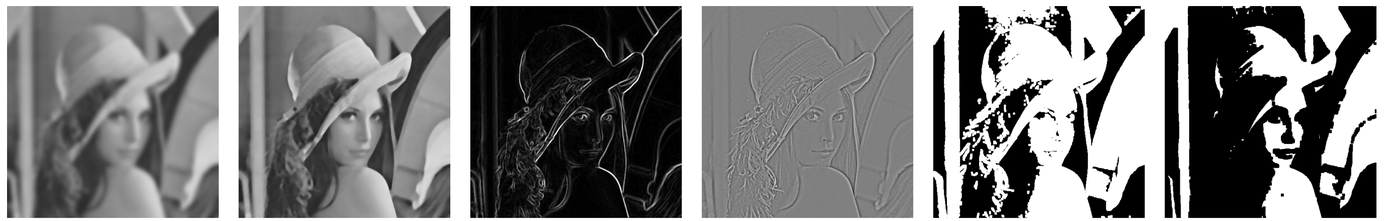
THE UNIVERSITY OF BRITISH COLUMBIA

CPSC 425: Computer Vision



Lecture 4: Image Filtering (continued)

(unless otherwise stated slides are taken or adopted from **Bob Woodham, Jim Little** and **Fred Tung**)

Menu for Today (**September 16, 2025**)

Topics:

- **Box, Gaussian, Pillbox** filters
- **Separability**
- **Low / High Pass** Filters
- More **examples** of Filtering

Readings:

- **Today's** Lecture: none
- **Next** Lecture: Forsyth & Ponce (2nd ed.) 4.4, Szeliski 3.2

Reminders:

- **Assignment 1** (graded) is due Wednsday, **September 24**
- **TA** and Office **Hours have started**
- **Quiz** on **September 23rd**

Today's "**fun**" Example: Rolling Shutter

Today's "**fun**" Example: Rolling Shutter

Today's "**fun**" Example: Rolling Shutter

Rolling shutter effect

Today's "**fun**" Example: Rolling Shutter

Rolling shutter effect

Lecture 4: Re-cap Linear Filter

$F(X,Y)$

filter

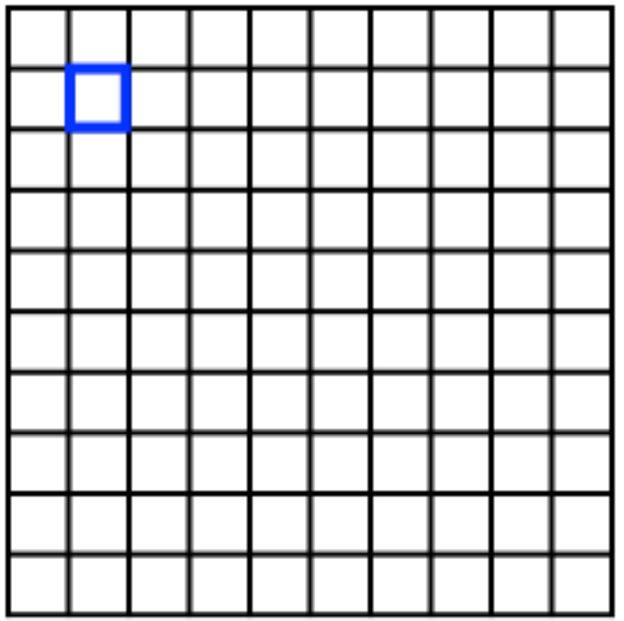
$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

image $I(X,Y)$

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

output $I'(X,Y)$



$$I'(X,Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i,j) I(X+i, Y+j)$$

output | filter | image (signal)

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Lecture 3: Re-cap Linear Filter

$F(X,Y)$

filter

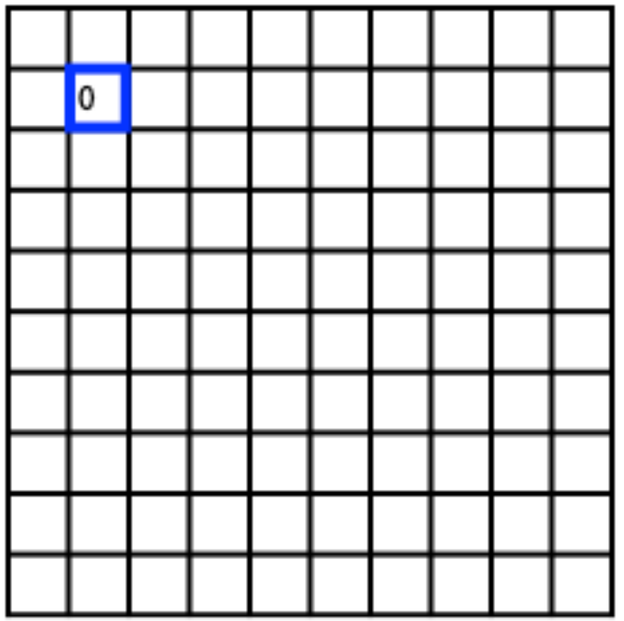
$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

image $I(X,Y)$

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

output $I'(X,Y)$



$$I'(X,Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i,j) I(X+i, Y+j)$$

output | filter | image (signal)

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Lecture 3: Re-cap Linear Filter

$F(X,Y)$

filter

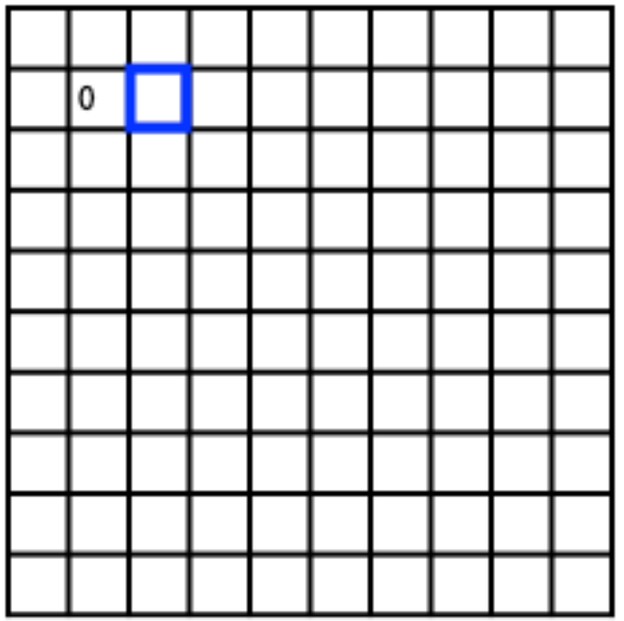
$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

image $I(X,Y)$

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

output $I'(X,Y)$



$$I'(X,Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i,j) I(X+i, Y+j)$$

output | filter | image (signal)

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Lecture 3: Re-cap Linear Filter

$F(X,Y)$

filter

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

image $I(X,Y)$

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

output $I'(X,Y)$

	0	10							

$$I'(X,Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i,j) I(X+i, Y+j)$$

output | filter | image (signal)

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Lecture 3: Re-cap Linear Filter

$F(X,Y)$

filter

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

image $I(X,Y)$

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

output $I'(X,Y)$

	0	10							

$$I'(X,Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i,j) I(X+i, Y+j)$$

output | filter | image (signal)

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Lecture 3: Re-cap Linear Filter

$F(X,Y)$

filter

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

image $I(X,Y)$

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

output $I'(X,Y)$

	0	10	20						

$$I'(X,Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i,j) I(X+i, Y+j)$$

output | filter | image (signal)

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Lecture 3: Re-cap Linear Filter

$F(X,Y)$

filter

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

image $I(X,Y)$

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

output $I'(X,Y)$

	0	10	20						

$$I'(X,Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i,j) I(X+i, Y+j)$$

output | filter | image (signal)

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Lecture 3: Re-cap Linear Filter

$F(X,Y)$

filter

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

image $I(X,Y)$

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

output $I'(X,Y)$

	0	10	20	30					

$$I'(X,Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i,j) I(X+i, Y+j)$$

$I'(X,Y)$: output

$F(i,j)$: filter

$I(X+i, Y+j)$: image (signal)

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Lecture 3: Re-cap Linear Filter

$F(X,Y)$

filter

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

image $I(X,Y)$

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

output $I'(X,Y)$

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	0	10	20	30	30	30	20	10	
	10	10	10	10	0	0	0	0	
	10	10	10	10	0	0	0	0	

$$I'(X,Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i,j) I(X+i, Y+j)$$

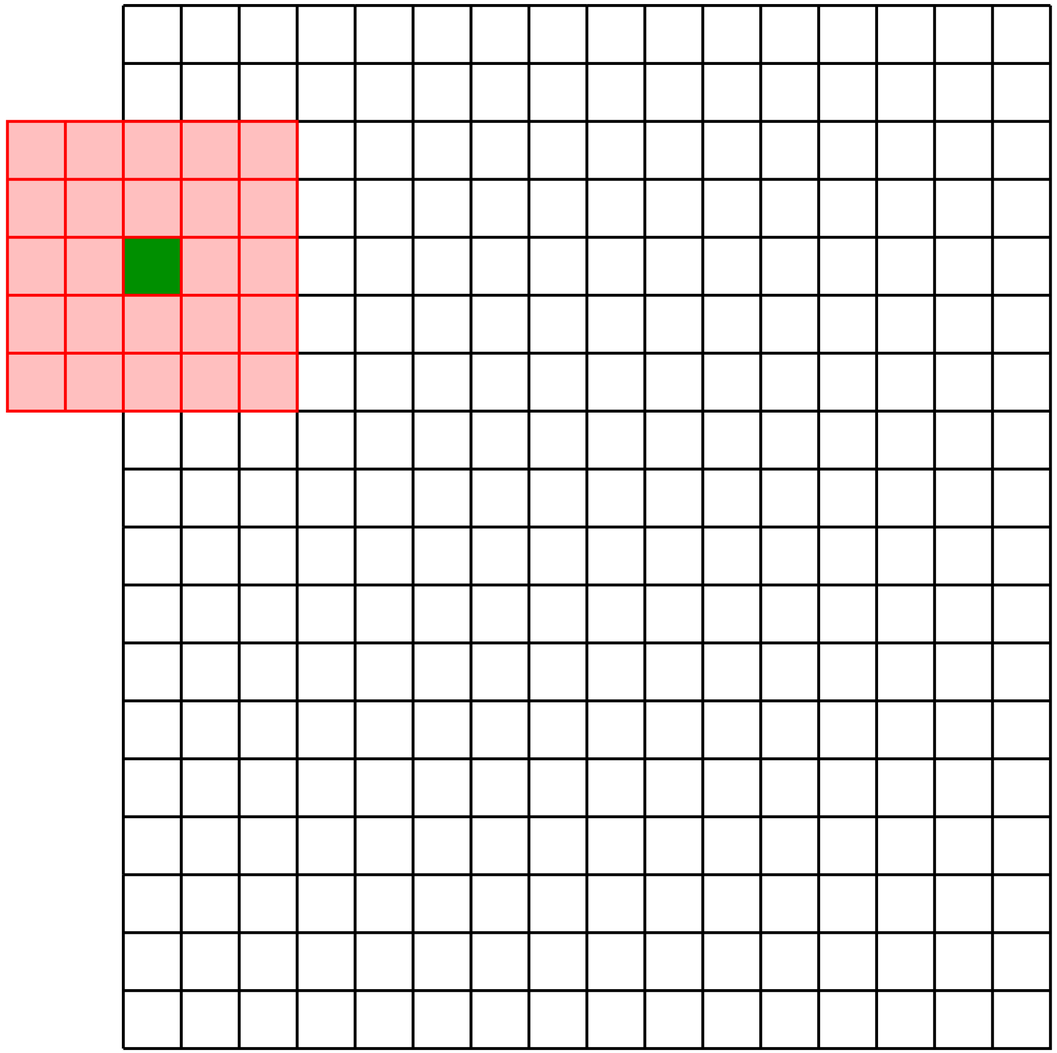
$I'(X,Y)$: output

$F(i,j)$: filter

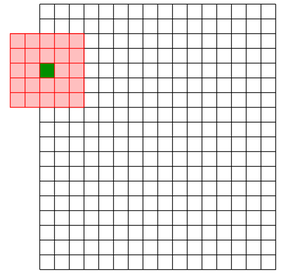
$I(X+i, Y+j)$: image (signal)

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Lecture 3: Re-cap Linear Filter **Boundary** Effects



Lecture 3: Re-cap Linear Filter **Boundary** Effects



Four standard ways to deal with boundaries:

1. **Ignore these locations:** Make the computation undefined for the top and bottom k rows and the leftmost and rightmost k columns

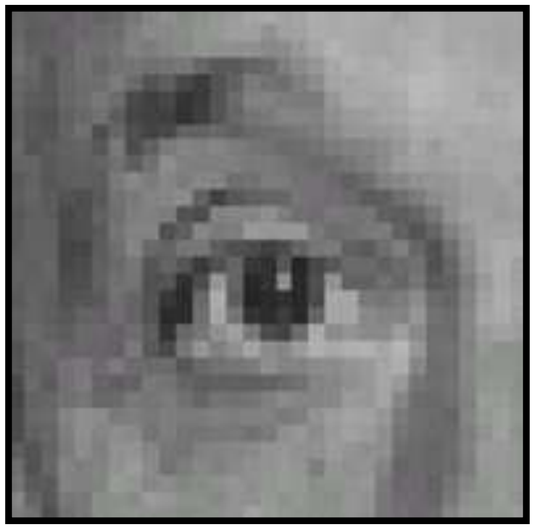
2. **Pad the image with zeros**: Return zero whenever a value of I is required at some position outside the defined limits of X and Y

3. **Assume periodicity**: The top row wraps around to the bottom row; the leftmost column wraps around to the rightmost column

4. **Reflect boarder**: Copy rows/columns locally by reflecting over the edge

A short exercise …

Example 1: Warm up

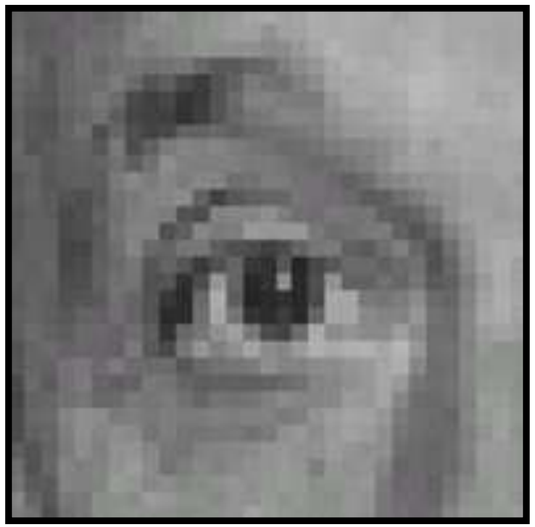


0	0	0
0	1	0
0	0	0

?

Original **Filter** **Result**

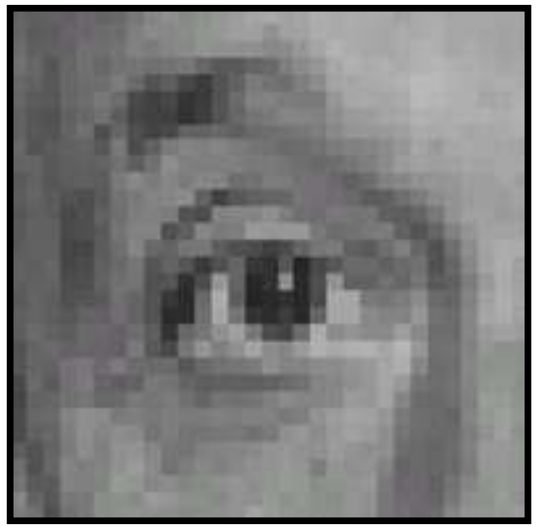
Example 1: Warm up



Original

0	0	0
0	1	0
0	0	0

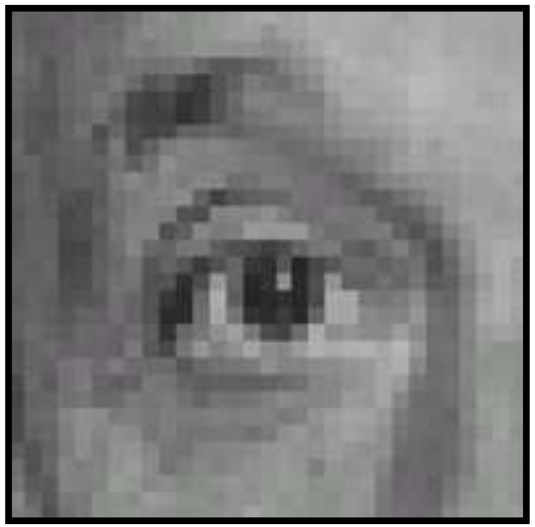
Filter



Result

(no change)

Example 2:



Original

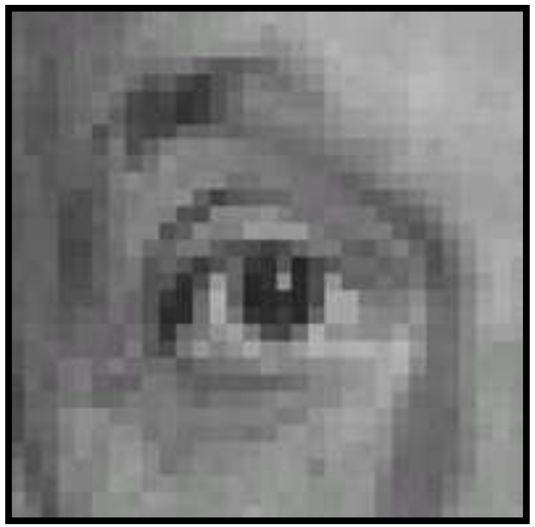
0	0	0
0	0	1
0	0	0

Filter

?

Result

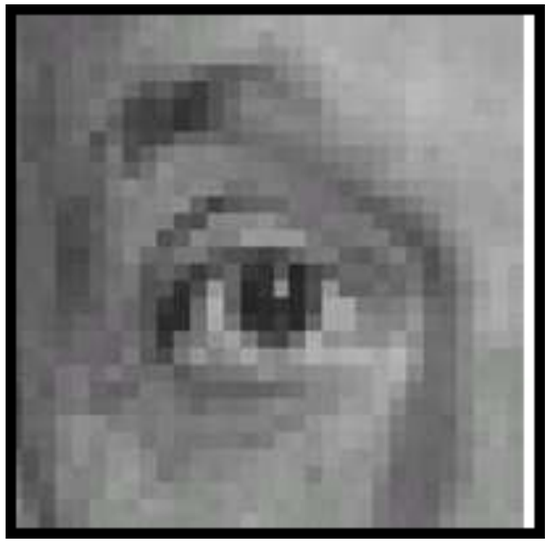
Example 2:



Original

0	0	0
0	0	1
0	0	0

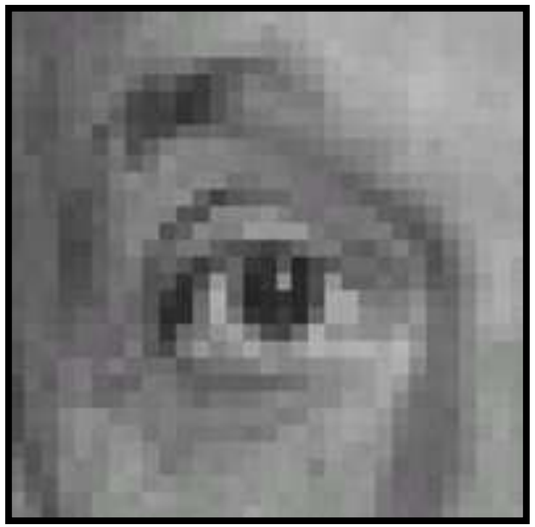
Filter



Result

(sift left by 1 pixel)

Example 3:



Original

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

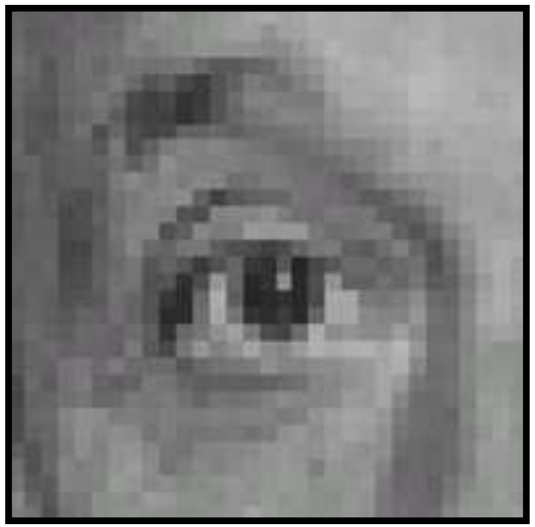
Filter

(filter sums to 1)

?

Result

Example 3:

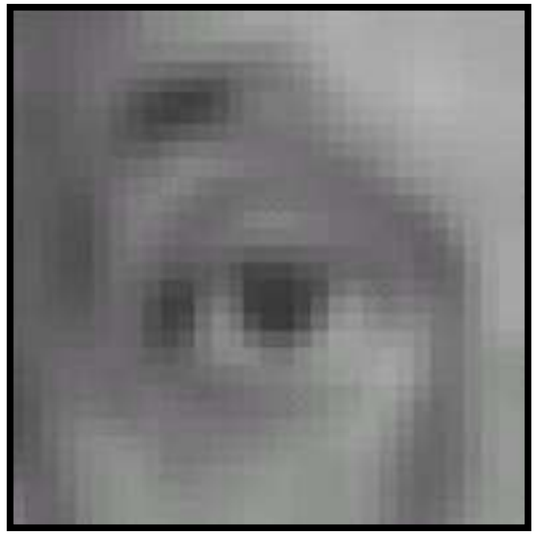


Original

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

Filter

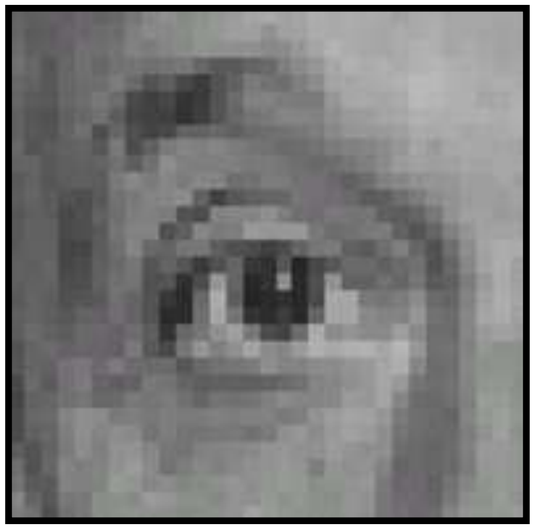
(filter sums to 1)



Result

(blur with a box filter)

Example 4:



Original

$$\frac{1}{9}\begin{bmatrix} -1 & -1 & -1 \\ -1 & 17 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Filter

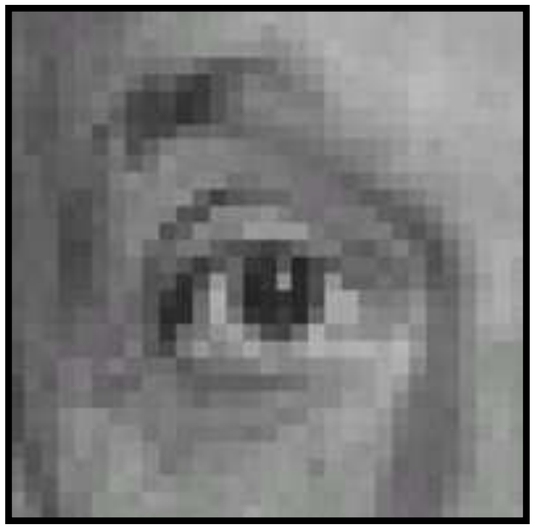
(filter sums to 1)

?

Result

Example 4:

$$\frac{1}{9}\begin{bmatrix} -1 & -1 & -1 \\ -1 & 17 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$



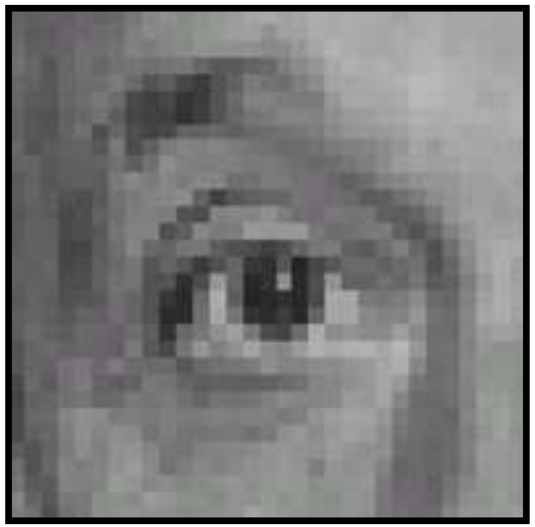
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

?

Original **Filter** **Result**

(filter sums to 1)

Example 4:



Original

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

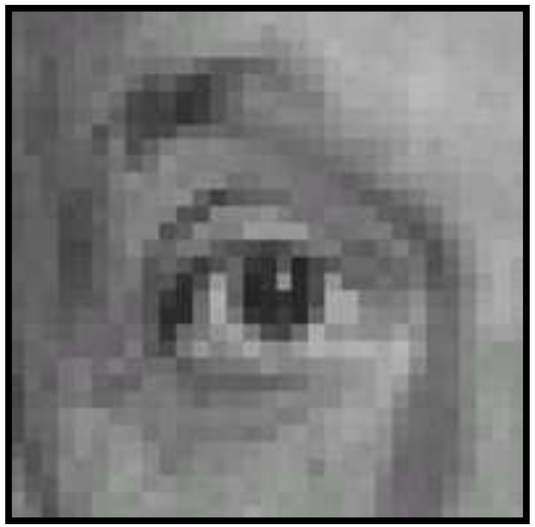
Filter

(filter sums to 1)

Result

(sharpening)

Example 4:



(Scaled)
Image Itself

0	0	0
0	2	0
0	0	0

$-\frac{1}{9}$

Blurred Version

1	1	1
1	1	1
1	1	1

Original

Filter

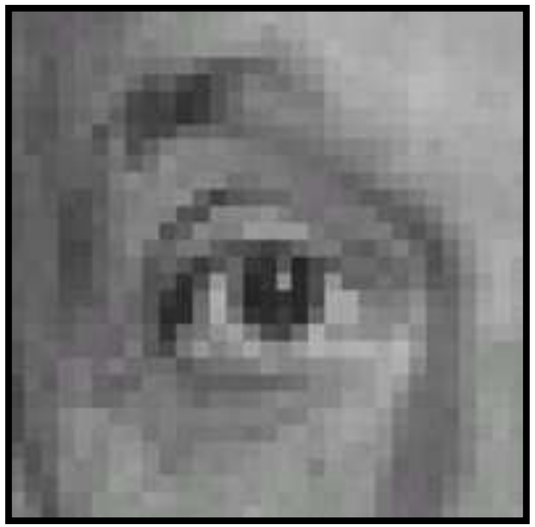
(filter sums to 1)

Result

(sharpening)

Example 4:

Why have filters sum up to 1?



Original

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

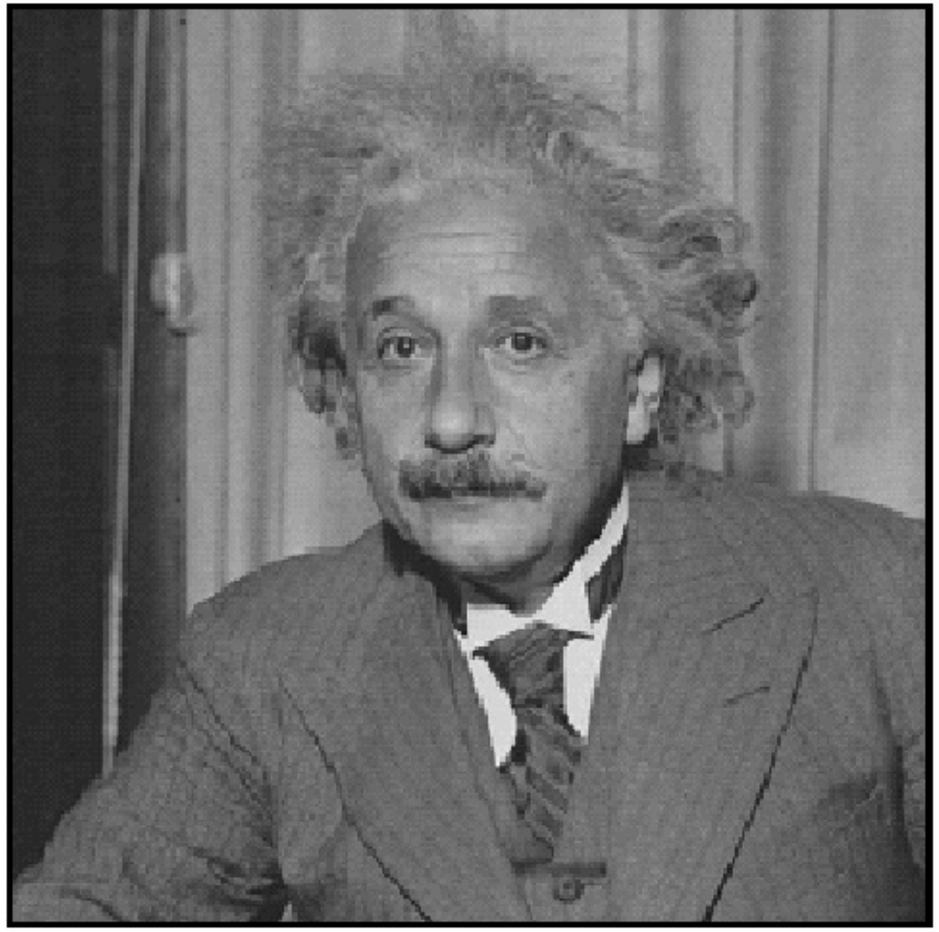
Filter

(filter sums to 1)

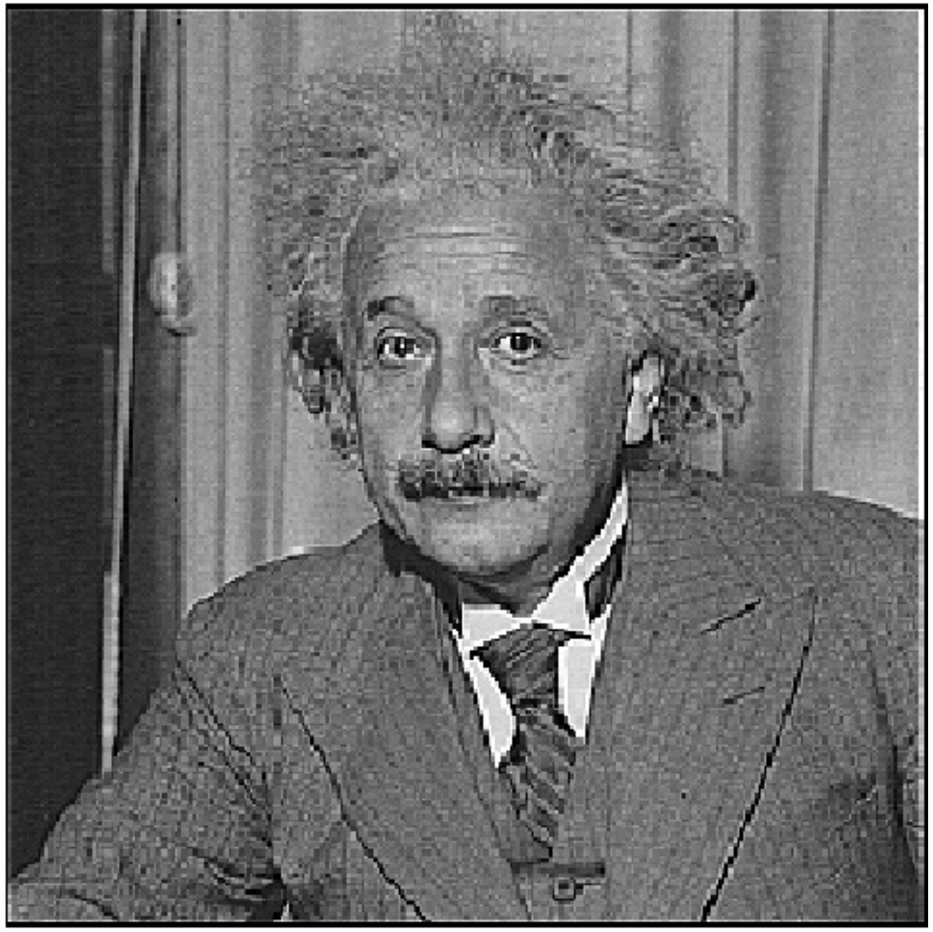
Result

(sharpening)

Example 4: Sharpening



Before



After

Example 4: Sharpening

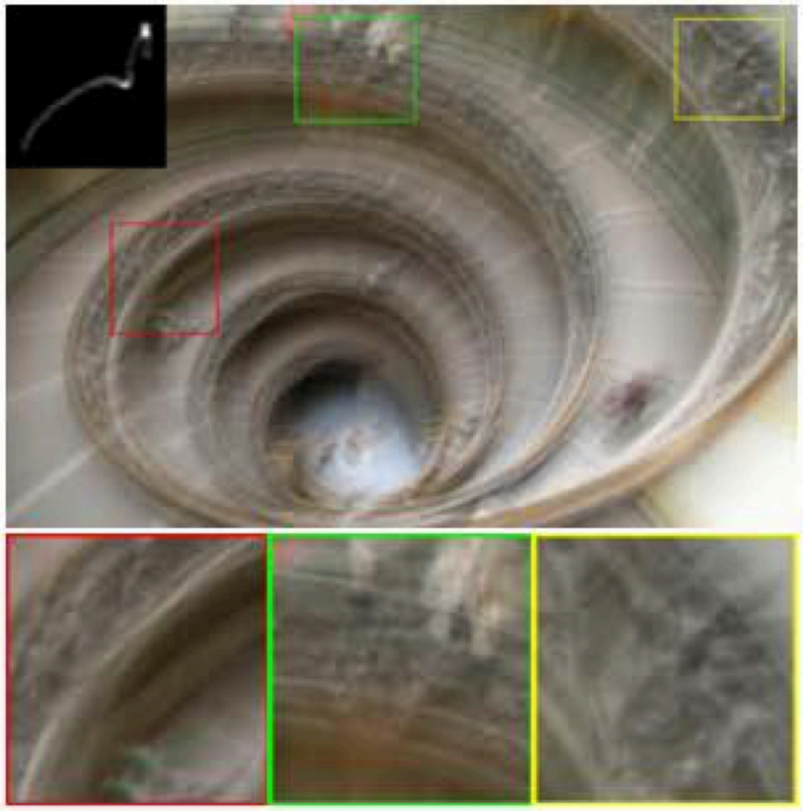
Before

After

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

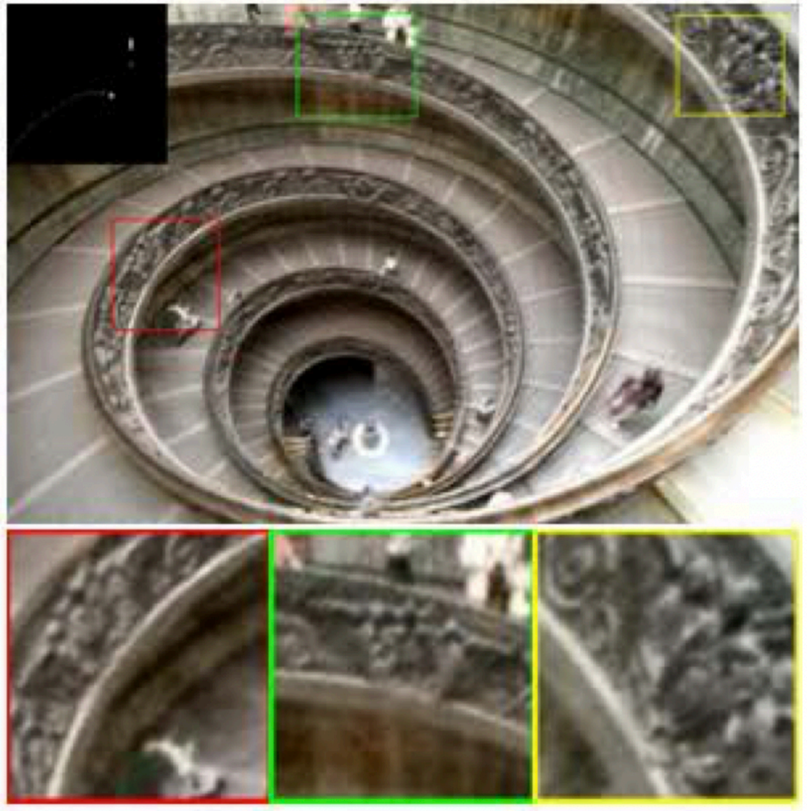
Aside: Blind deconvolution

Ren et al., CVPR 2020



Blurry image

Ground-truth



SelfDeblur

Linear Filters: Correlation vs. Convolution

Definition: **Correlation**

$$I'(X,Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i,j) I(X+i, Y+j)$$

Linear Filters: Correlation vs. Convolution

Definition: **Correlation**

$$I'(X,Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i,j) I(X+i, Y+j)$$

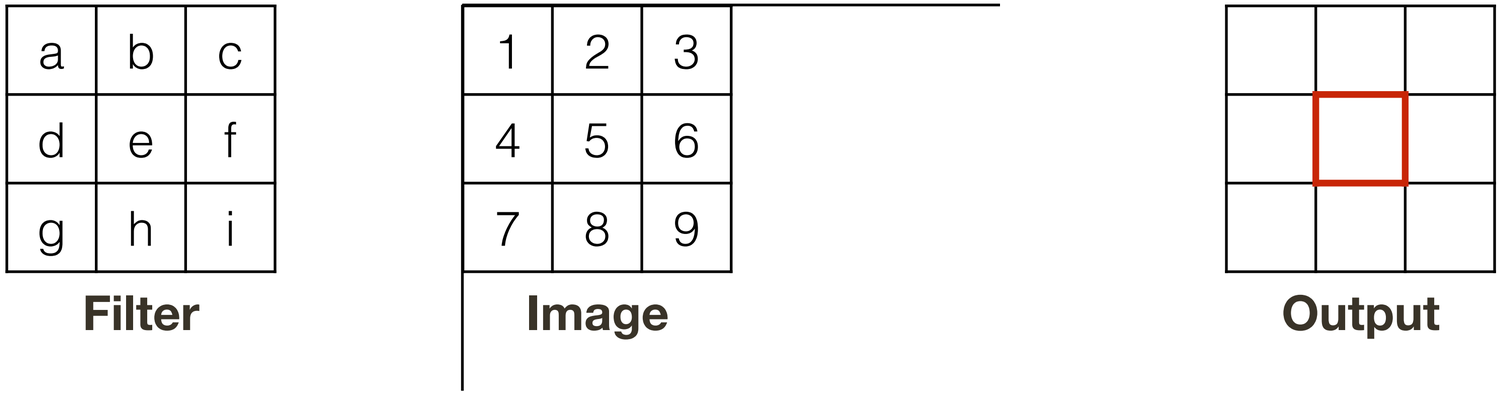
Definition: **Convolution**

$$I'(X,Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i,j) I(X-i, Y-j)$$

Linear Filters: Correlation vs. Convolution

Definition: **Correlation**

$$I'(X,Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i,j) I(X+i, Y+j)$$



Linear Filters: Correlation vs. Convolution

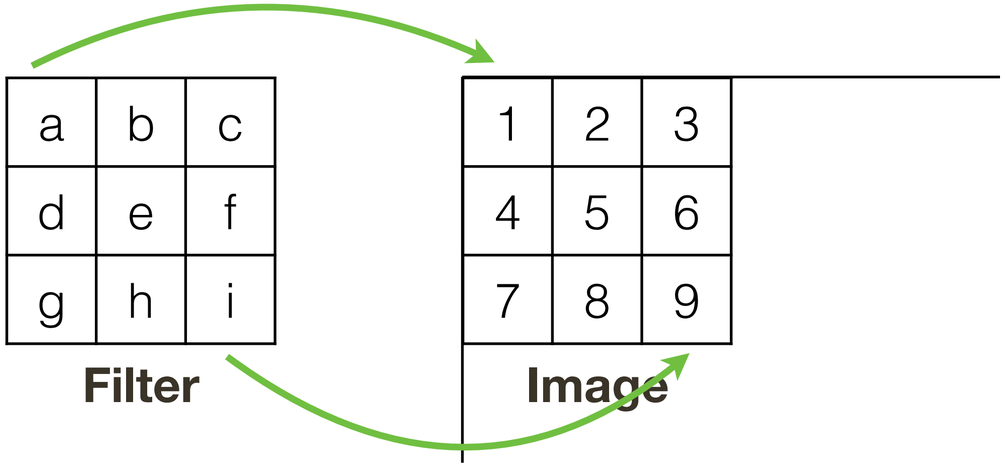
Definition: **Correlation**

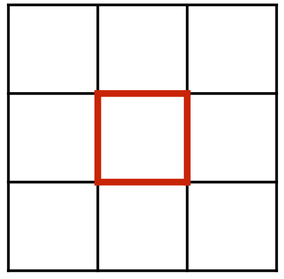
$$I'(X,Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i,j) I(X+i, Y+j)$$

Linear Filters: Correlation vs. Convolution

Definition: **Correlation**

$$I'(X,Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i,j) I(X+i, Y+j)$$





= 1a + 2b + 3c
+ 4d + 5e + 6f
+ 7g + 8h + 9i

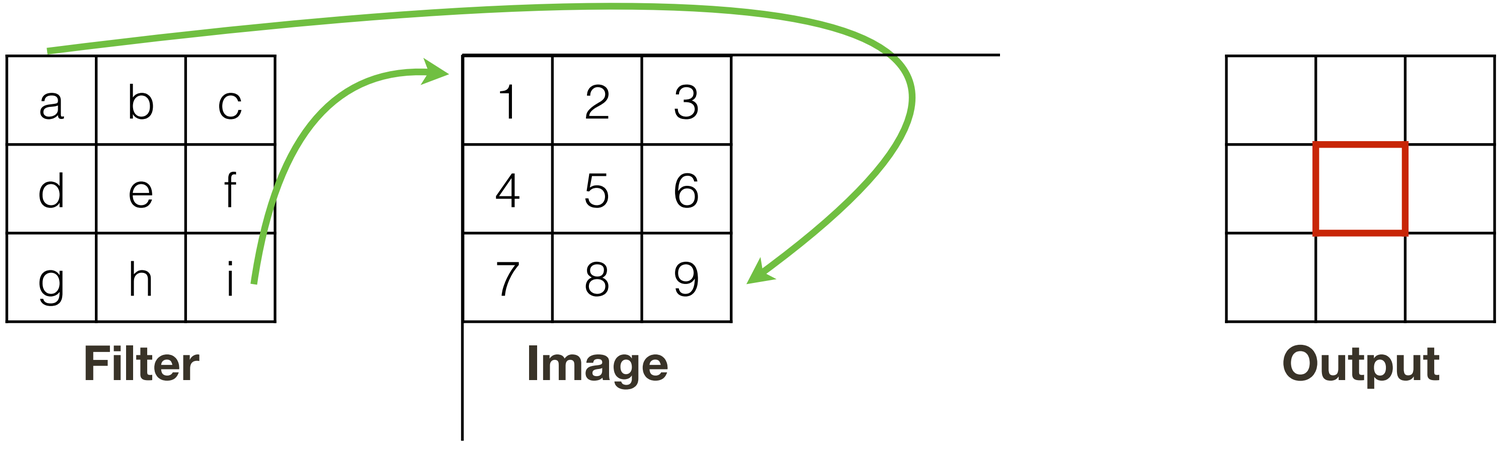
Linear Filters: Correlation vs. Convolution

Definition: **Correlation**

$$I'(X,Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i,j) I(X+i, Y+j)$$

Definition: **Convolution**

$$I'(X,Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i,j) I(X-i, Y-j)$$



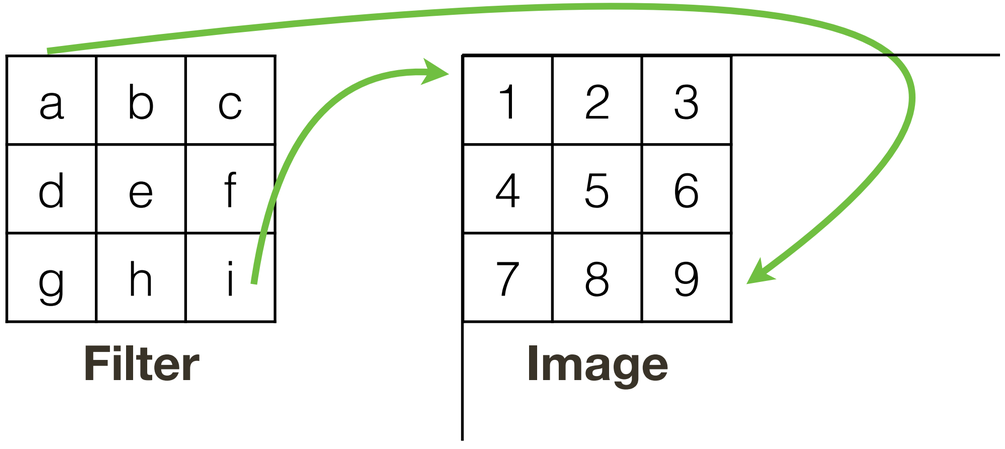
Linear Filters: Correlation vs. Convolution

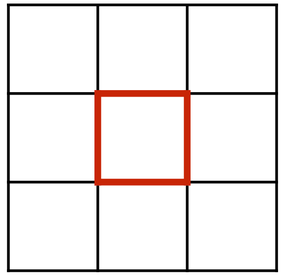
Definition: **Correlation**

$$I'(X,Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i,j) I(X+i, Y+j)$$

Definition: **Convolution**

$$I'(X,Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i,j) I(X-i, Y-j)$$





= 9a + 8b + 7c
+ 6d + 5e + 4f
+ 3g + 2h + 1i

Linear Filters: Correlation vs. Convolution

Definition: **Correlation**

$$I'(X,Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i,j) I(X+i, Y+j)$$

Definition: **Convolution**

$$I'(X,Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i,j) I(X-i, Y-j)$$

Filter
(rotated by 180)

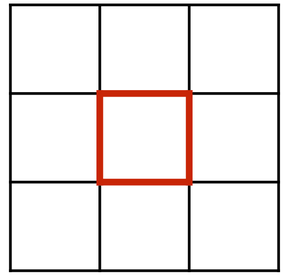
i	h	g
f	e	d
c	b	a

a	b	c
d	e	f
g	h	i

Filter

1	2	3
4	5	6
7	8	9

Image



Output

= 9a + 8b + 7c
+ 6d + 5e + 4f
+ 3g + 2h + 1i

Linear Filters: Correlation vs. Convolution

Definition: **Correlation**

$$I'(X,Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i,j) I(X+i, Y+j)$$

Definition: **Convolution**

$$I'(X,Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i,j) I(X-i, Y-j)$$

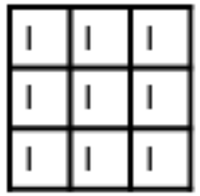
$$= \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(-i,-j) I(X+i, Y+j)$$

Note: if $F(X,Y) = F(-X,-Y)$ then correlation = convolution.

Linear Filters: Correlation vs. Convolution

$F(X,Y)$

filter

$\frac{1}{9}$ 

180 degree symmetric => when applied as **correlation** or **convolution** it will yield **same result**

Note: if $F(X,Y) = F(-X,-Y)$ then correlation = convolution.

Linear Filters: Correlation vs. Convolution

$F(X,Y)$

filter

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

180 degree symmetric => when applied as **correlation** or **convolution** it will yield **same result**

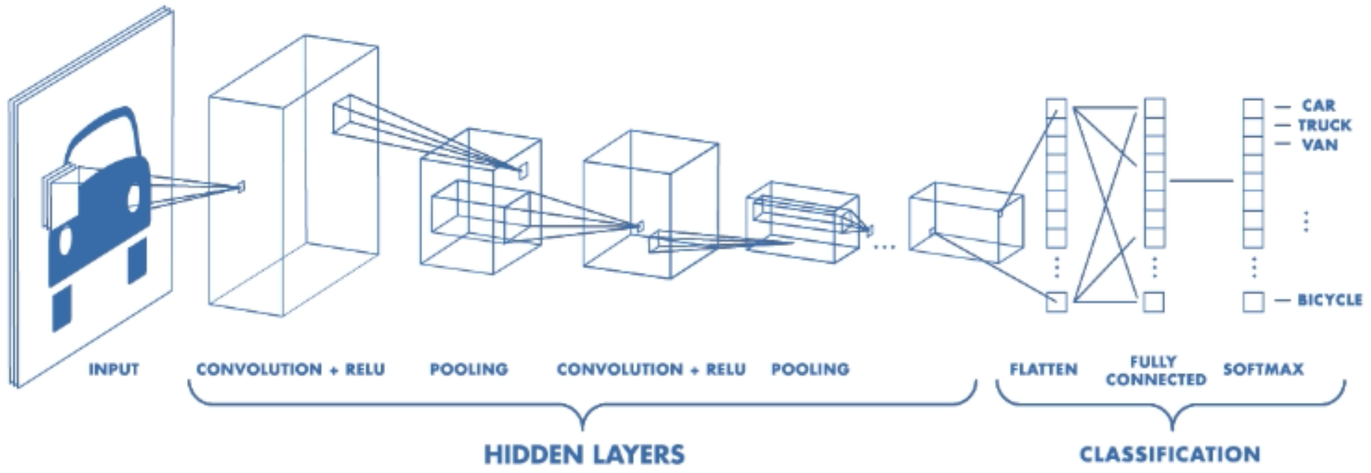
6	7	1
2	0	2
1	7	6

… so is this one

Note: if $F(X,Y) = F(-X,-Y)$ then correlation = convolution.

Preview: Why convolutions are important?

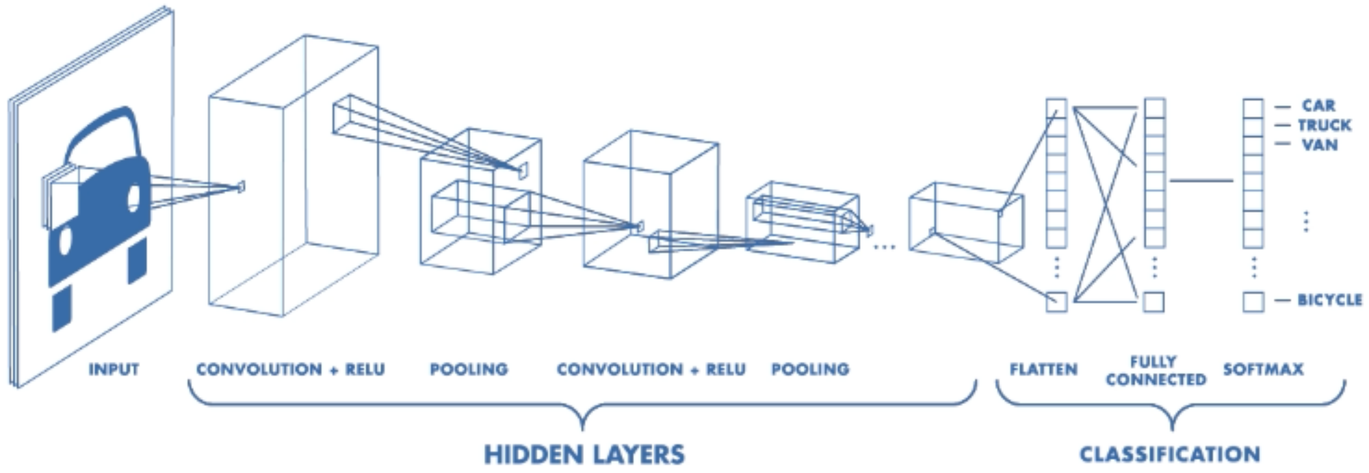
Who has heard of **Convolutional Neural Networks** (CNNs)?



Preview: Why convolutions are important?

Who has heard of **Convolutional Neural Networks** (CNNs)?

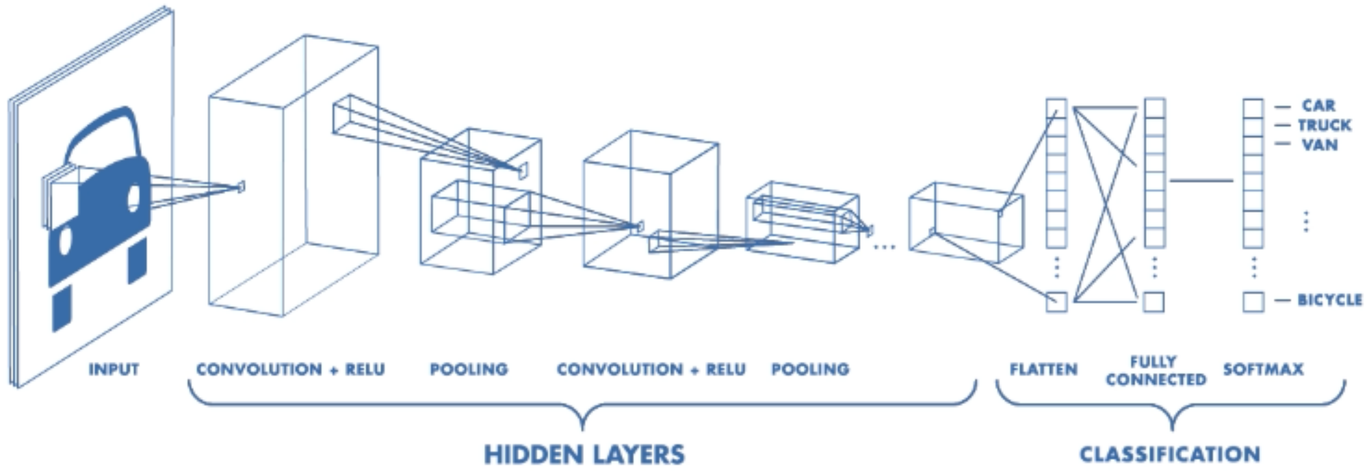
What about **Deep Learning**?



Preview: Why convolutions are important?

Who has heard of **Convolutional Neural Networks** (CNNs)?

What about **Deep Learning**?



Basic operations in CNNs are convolutions (with learned linear filters) followed by non-linear functions.

Note: This results in non-linear filters.

Aside: Convolution as Matrix Multiplication

Filter

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

Image

$$\begin{bmatrix} 90 & 90 & 90 & 90 & 90 \\ 90 & 90 & 90 & 90 & 90 \\ 90 & 0 & 90 & 90 & 90 \\ 90 & 90 & 90 & 90 & 90 \end{bmatrix}$$

Aside: Convolution as Matrix Multiplication

Filter **Image**

$$\begin{bmatrix} 80 & 80 & 90 \\ 80 & 80 & 90 \end{bmatrix} = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \oplus \begin{bmatrix} 90 & 90 & 90 & 90 & 90 \\ 90 & 90 & 90 & 90 & 90 \\ 90 & 0 & 90 & 90 & 90 \\ 90 & 90 & 90 & 90 & 90 \end{bmatrix}$$

Aside: Convolution as Matrix Multiplication

Filter **Image**

$$\begin{bmatrix} 80 & 80 & 90 \\ 80 & 80 & 90 \end{bmatrix} = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \oplus \begin{bmatrix} 90 & 90 & 90 & 90 & 90 \\ 90 & 90 & 90 & 90 & 90 \\ 90 & 0 & 90 & 90 & 90 \\ 90 & 90 & 90 & 90 & 90 \end{bmatrix}$$

$$\begin{bmatrix} 90 \\ 90 \\ 90 \\ 90 \\ 90 \\ 90 \\ \hline 90 \\ 90 \\ 90 \\ 90 \\ 90 \\ \hline 90 \\ 0 \\ 90 \\ 90 \\ 90 \\ \hline 90 \\ 90 \\ 90 \\ 90 \\ 90 \end{bmatrix}$$

Aside: Convolution as Matrix Multiplication

Filter **Image**

$$\begin{bmatrix} 80 & 80 & 90 \\ 80 & 80 & 90 \end{bmatrix} = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \oplus \begin{bmatrix} 90 & 90 & 90 & 90 & 90 \\ 90 & 90 & 90 & 90 & 90 \\ 90 & 0 & 90 & 90 & 90 \\ 90 & 90 & 90 & 90 & 90 \end{bmatrix}$$

$$\frac{1}{9}\left[\begin{array}{ccccc|ccccc|ccccc|ccccc} 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \end{array}\right] \times \left[\begin{array}{c} 90 \\ 90 \\ 90 \\ 90 \\ 90 \\ \hline 90 \\ 90 \\ 90 \\ 90 \\ 90 \\ \hline 90 \\ 0 \\ 90 \\ 90 \\ 90 \\ \hline 90 \\ 90 \\ 90 \\ 90 \\ 90 \end{array}\right]$$

Aside: Convolution as Matrix Multiplication

$$\begin{bmatrix} 80 & 80 & 90 \\ 80 & 80 & 90 \end{bmatrix} = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \oplus \begin{bmatrix} 90 & 90 & 90 & 90 & 90 \\ 90 & 90 & 90 & 90 & 90 \\ 90 & 0 & 90 & 90 & 90 \\ 90 & 90 & 90 & 90 & 90 \end{bmatrix}$$

Filter **Image**

$$\frac{1}{9}\left[\begin{array}{ccccc|ccccc|ccccc|ccccc} 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \end{array}\right] \times \begin{bmatrix} 90 \\ 90 \\ 90 \\ 90 \\ 90 \\ 90 \\ \hline 90 \\ 90 \\ 90 \\ 90 \\ 90 \\ 90 \\ \hline 90 \\ 0 \\ 90 \\ 90 \\ 90 \\ \hline 90 \\ 90 \\ 90 \\ 90 \\ 90 \end{bmatrix}$$

Aside: Convolution as Matrix Multiplication

$$\begin{bmatrix} 80 & 80 & 90 \\ 80 & 80 & 90 \end{bmatrix} = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \oplus \begin{bmatrix} 90 & 90 & 90 & 90 & 90 \\ 90 & 90 & 90 & 90 & 90 \\ 90 & 0 & 90 & 90 & 90 \\ 90 & 90 & 90 & 90 & 90 \end{bmatrix}$$

Filter **Image**

$$\frac{1}{9}\left[\begin{array}{ccccc|ccccc|ccccc|ccccc} 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \end{array}\right] \times \begin{bmatrix} 90 \\ 90 \\ 90 \\ 90 \\ 90 \\ 90 \\ 90 \\ 90 \\ 90 \\ 90 \\ 90 \\ 90 \\ 0 \\ 90 \\ 90 \\ 90 \\ 90 \\ 90 \\ 90 \\ 90 \end{bmatrix}$$

Aside: Convolution as Matrix Multiplication

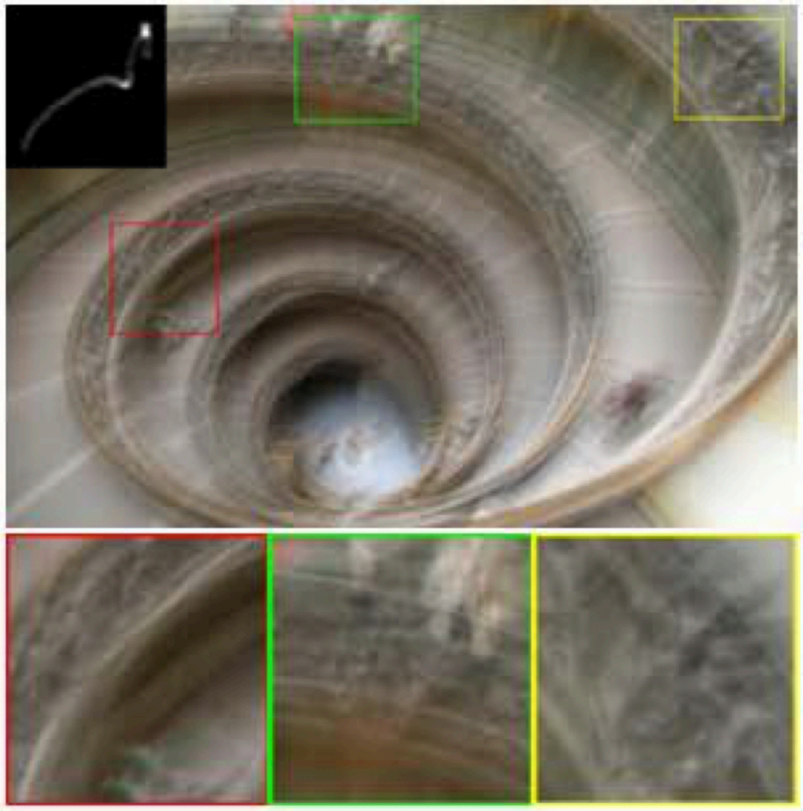
Filter **Image**

$$\begin{bmatrix} 80 & 80 & 90 \\ 80 & 80 & 90 \end{bmatrix} = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \oplus \begin{bmatrix} 90 & 90 & 90 & 90 & 90 \\ 90 & 90 & 90 & 90 & 90 \\ 90 & 0 & 90 & 90 & 90 \\ 90 & 90 & 90 & 90 & 90 \end{bmatrix}$$

$$\begin{bmatrix} 80 \\ 80 \\ 90 \\ \hline 80 \\ 80 \\ 90 \end{bmatrix} = \frac{1}{9}\left[\begin{array}{ccccc|ccccc|ccccc|ccccc} 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \end{array}\right] \times \begin{bmatrix} 90 \\ 90 \\ 90 \\ 90 \\ 90 \\ \hline 90 \\ 90 \\ 90 \\ 90 \\ 90 \\ \hline 90 \\ 0 \\ 90 \\ 90 \\ 90 \\ \hline 90 \\ 90 \\ 90 \\ 90 \\ 90 \end{bmatrix}$$

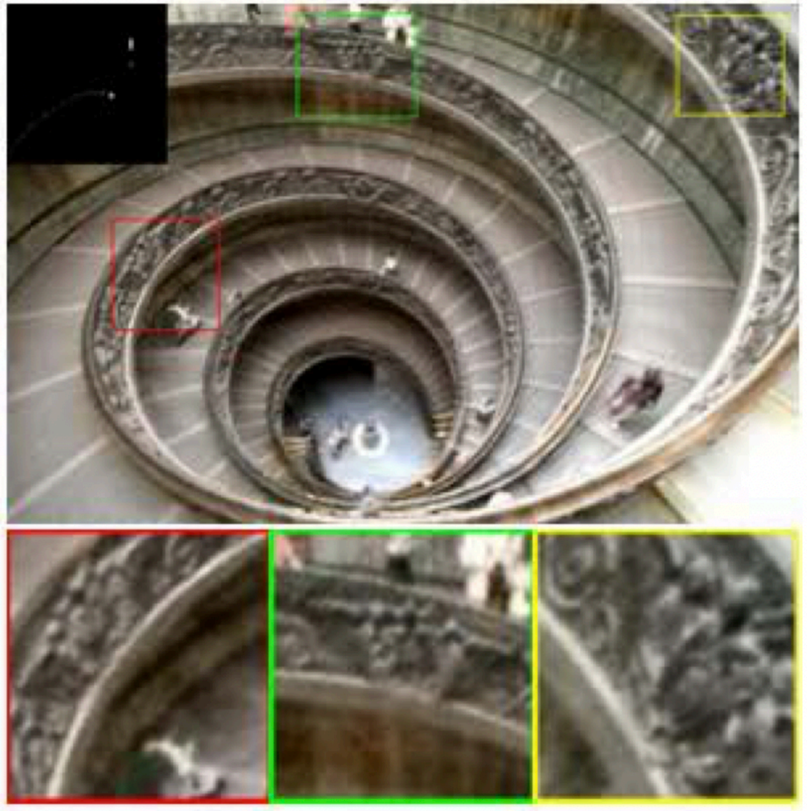
Aside: Blind deconvolution

Ren et al., CVPR 2020



Blurry image

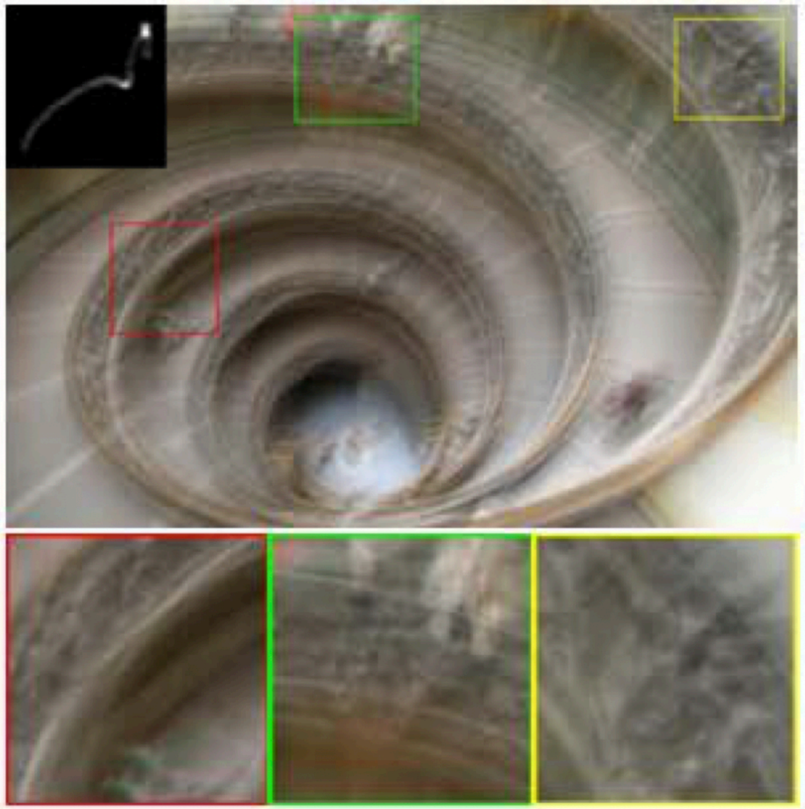
Ground-truth



SelfDeblur

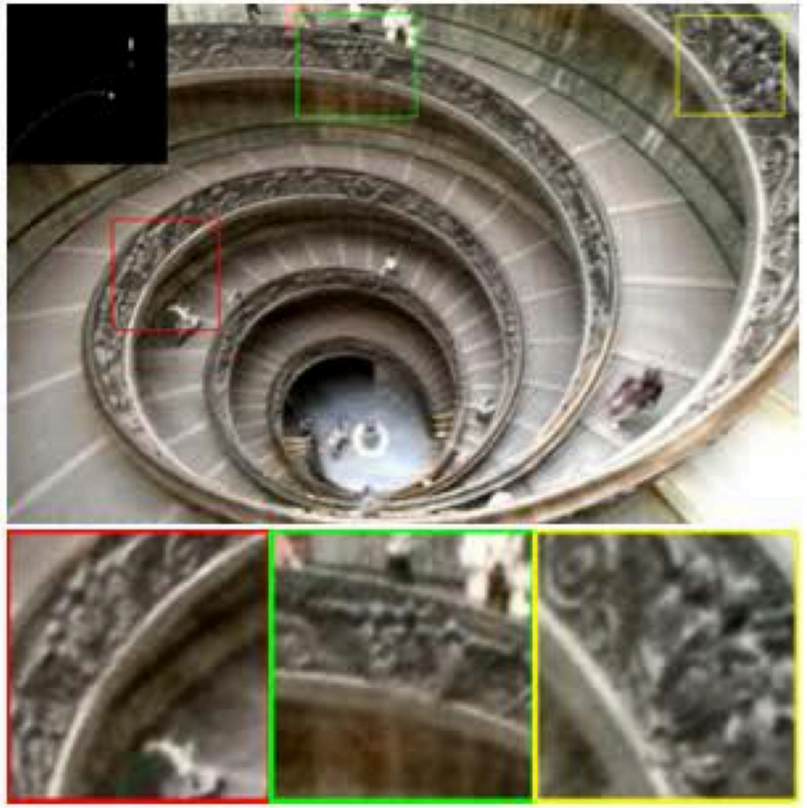
Aside: Blind deconvolution

Ren et al., CVPR 2020



Blurry image

Ground-truth



SelfDeblur

Challenge: We usually don’t know the kernel

Linear Filters: **Properties**

Let $\otimes$ denote convolution. Let $I(X, Y)$ be a digital image

Superposition: Let F_1 and F_2 be digital filters

$$(F_1 + F_2) \otimes I(X, Y) = F_1 \otimes I(X, Y) + F_2 \otimes I(X, Y)$$

Linear Filters: **Properties**

Let $\otimes$ denote convolution. Let $I(X, Y)$ be a digital image

Superposition: Let F_1 and F_2 be digital filters

$$(F_1 + F_2) \otimes I(X, Y) = F_1 \otimes I(X, Y) + F_2 \otimes I(X, Y)$$

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

Linear Filters: **Properties**

Let $\otimes$ denote convolution. Let $I(X, Y)$ be a digital image

Superposition: Let F_1 and F_2 be digital filters

$$(F_1 + F_2) \otimes I(X, Y) = F_1 \otimes I(X, Y) + F_2 \otimes I(X, Y)$$

Scaling: Let F be digital filter and let k be a scalar

$$(kF) \otimes I(X, Y) = F \otimes (kI(X, Y)) = k(F \otimes I(X, Y))$$

Linear Filters: **Properties**

Let $\otimes$ denote convolution. Let $I(X, Y)$ be a digital image

Superposition: Let F_1 and F_2 be digital filters

$$(F_1 + F_2) \otimes I(X, Y) = F_1 \otimes I(X, Y) + F_2 \otimes I(X, Y)$$

Scaling: Let F be digital filter and let k be a scalar

$$(kF) \otimes I(X, Y) = F \otimes (kI(X, Y)) = k(F \otimes I(X, Y))$$

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

Linear Filters: **Properties**

Let $\otimes$ denote convolution. Let $I(X, Y)$ be a digital image

Superposition: Let F_1 and F_2 be digital filters

$$(F_1 + F_2) \otimes I(X, Y) = F_1 \otimes I(X, Y) + F_2 \otimes I(X, Y)$$

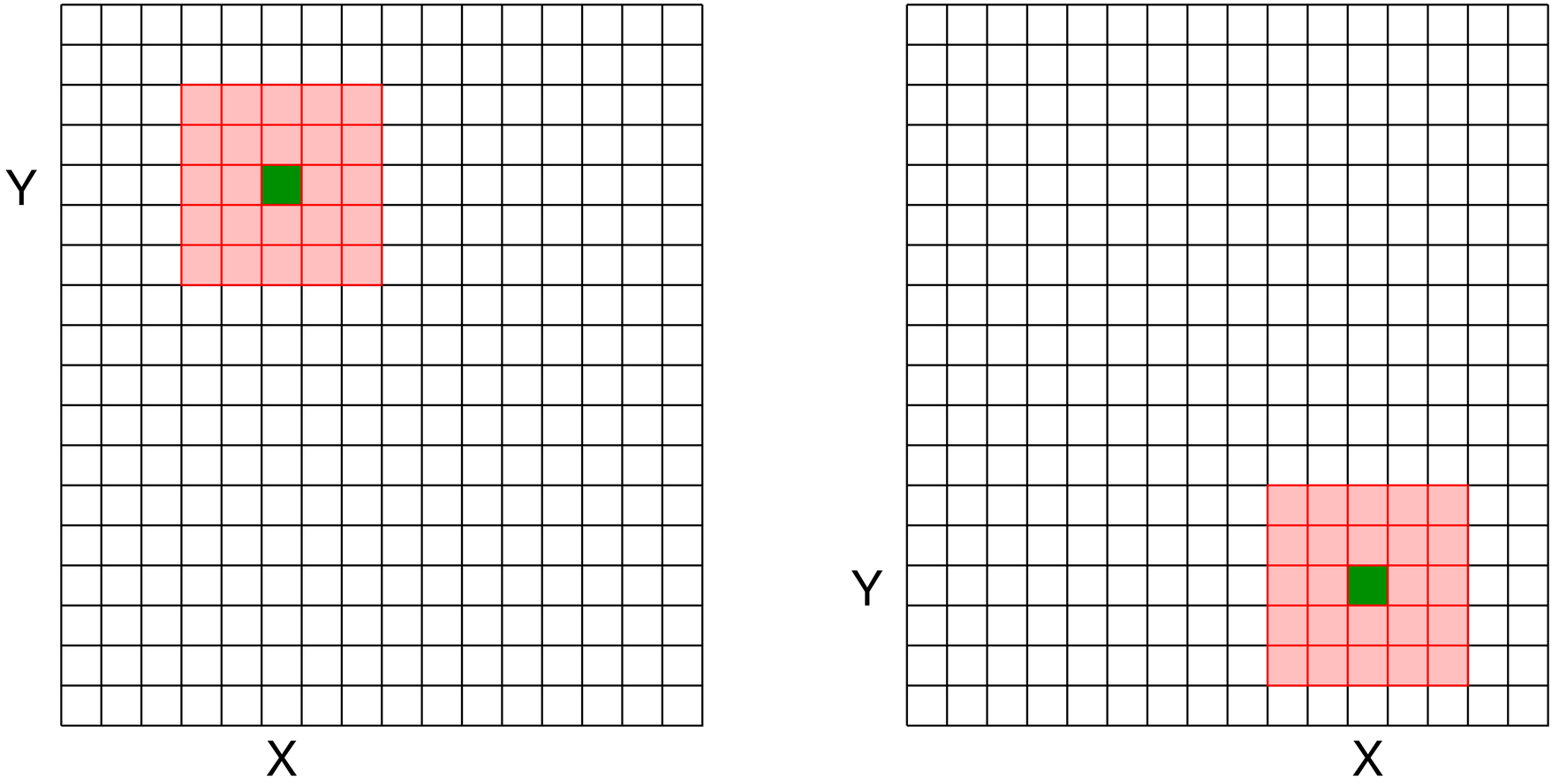
Scaling: Let F be digital filter and let k be a scalar

$$(kF) \otimes I(X, Y) = F \otimes (kI(X, Y)) = k(F \otimes I(X, Y))$$

Shift Invariance: Output is local (i.e., no dependence on absolute position)

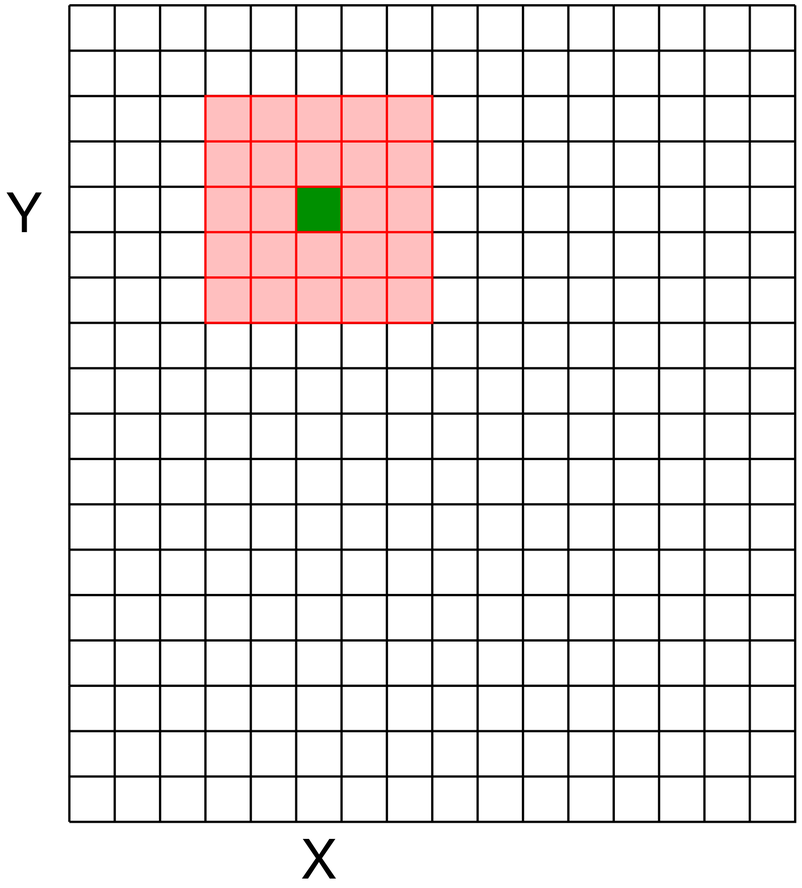
Linear Filters: Shift Invariance

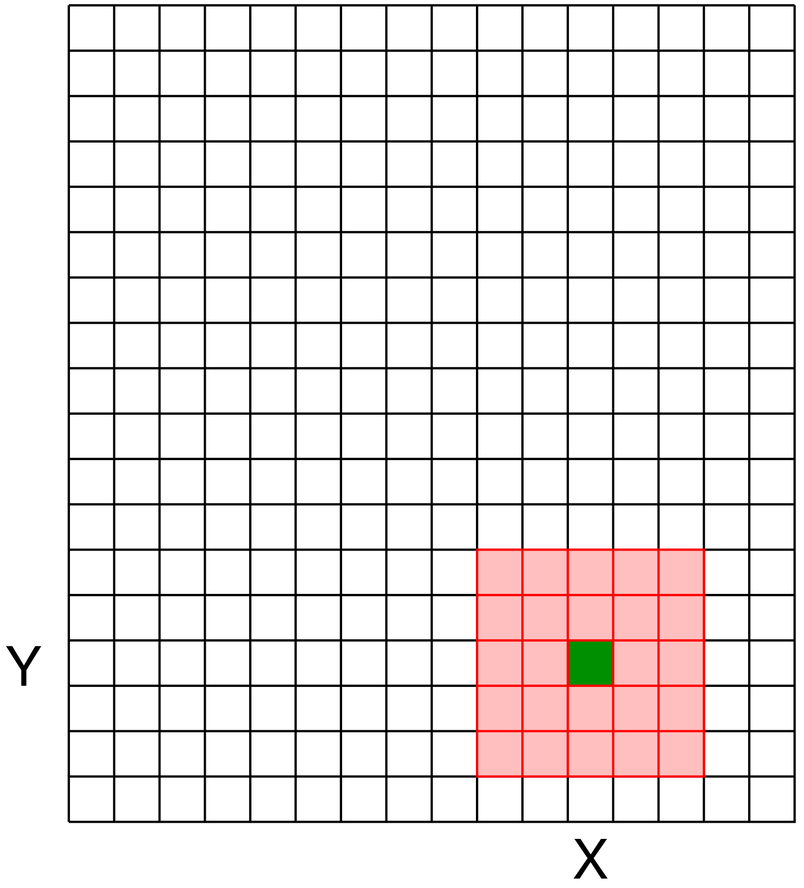
Output does **not** depend on absolute position



Linear Filters: Shift Invariance

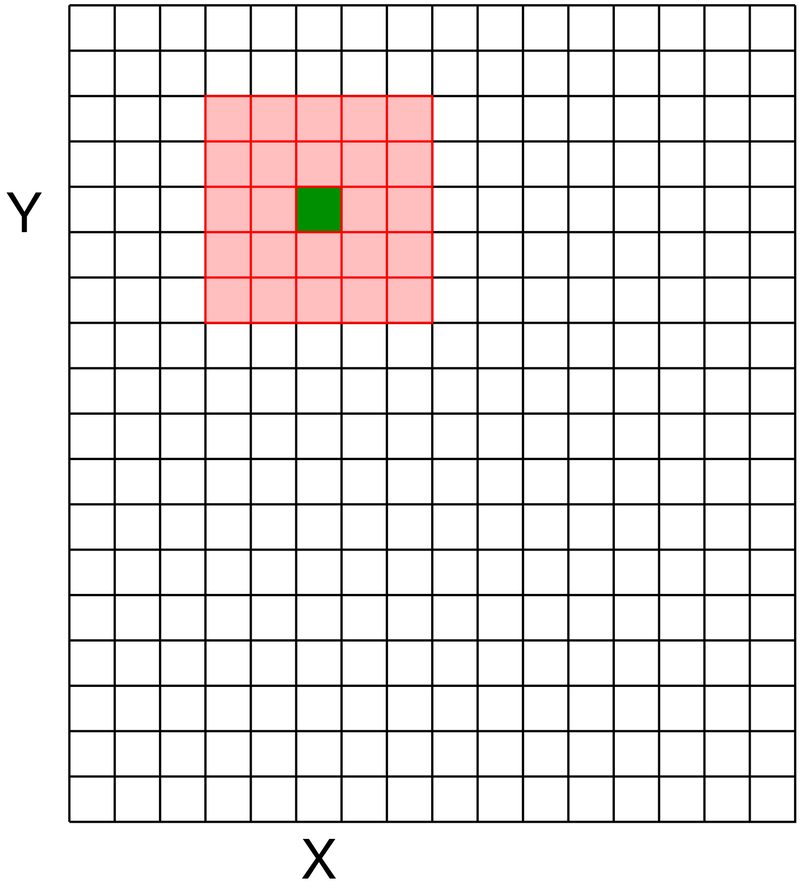
$$I'(X,Y) = f\left(F, I\left(X - \lfloor\frac{k}{2}\rfloor : X + \lfloor\frac{k}{2}\rfloor, Y - \lfloor\frac{k}{2}\rfloor : Y + \lfloor\frac{k}{2}\rfloor\right)\right)$$

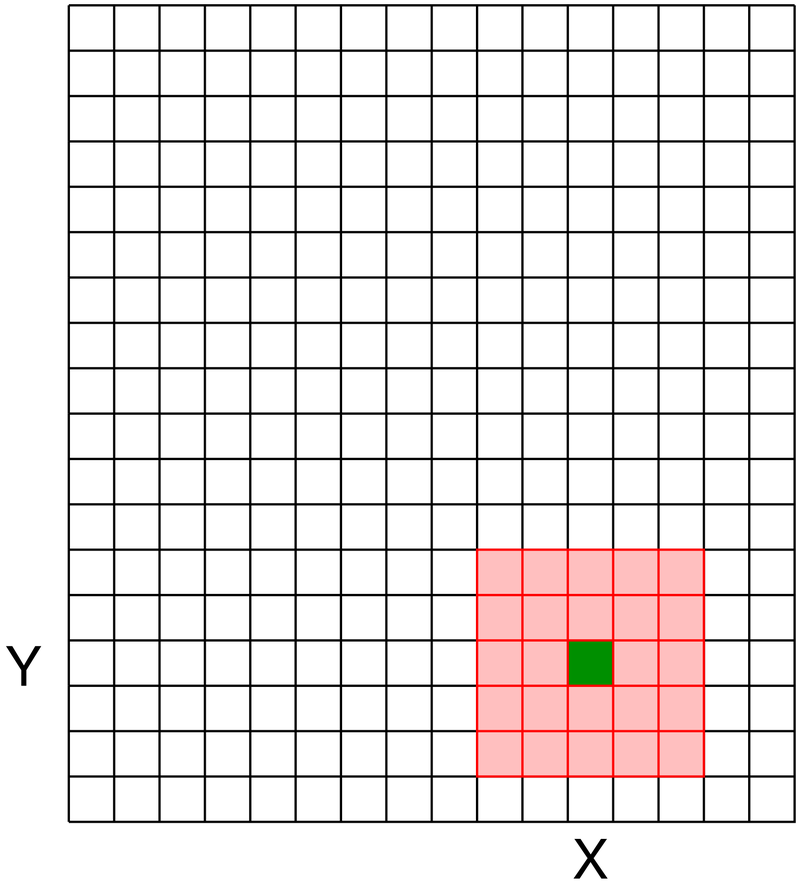




Linear Filters: Shift Variant

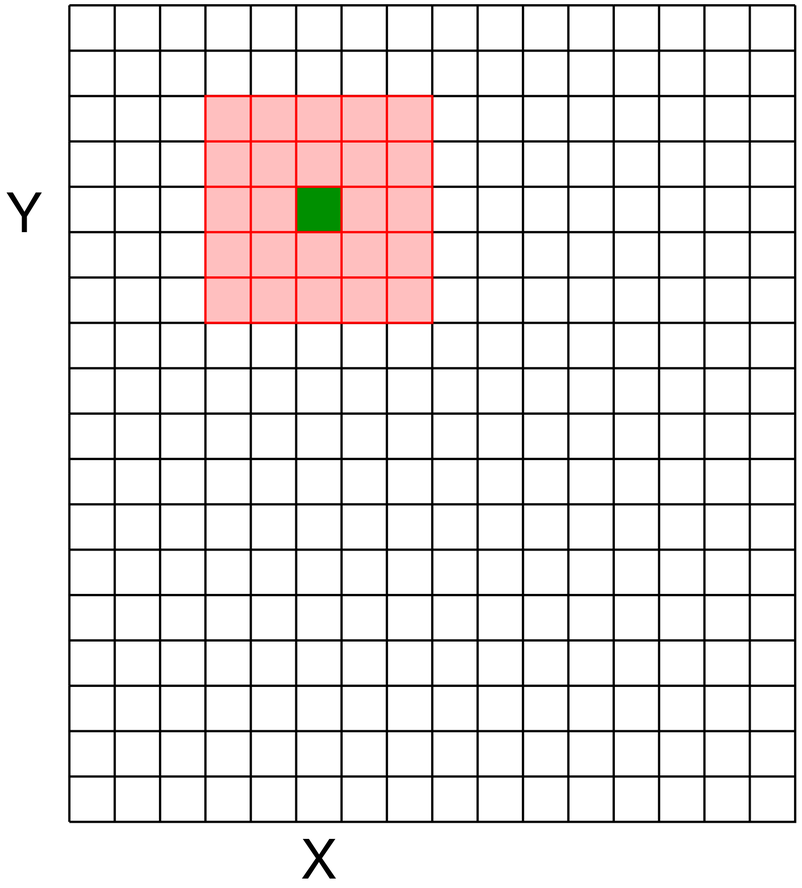
$$I'(X,Y) = f\left(F, I\left(X - \lfloor\frac{k}{2}\rfloor : X + \lfloor\frac{k}{2}\rfloor, Y - \lfloor\frac{k}{2}\rfloor : Y + \lfloor\frac{k}{2}\rfloor\right), X, Y\right)$$

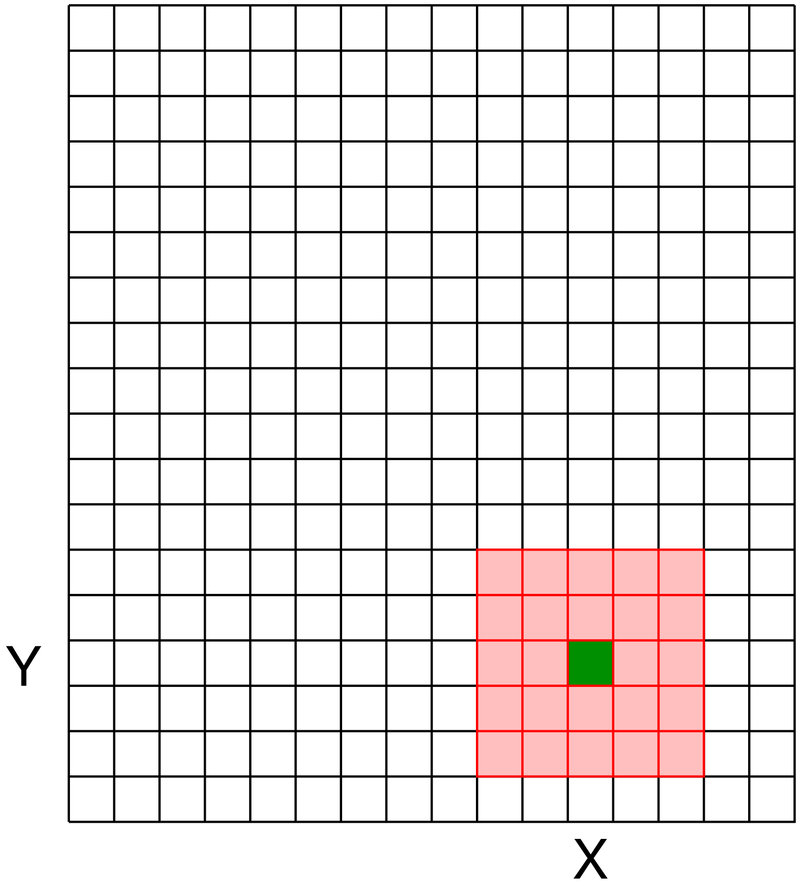




Linear Filters: Shift Variant

$$I'(X,Y) = f\left(F_{X,Y}, I\left(X - \lfloor\frac{k}{2}\rfloor : X + \lfloor\frac{k}{2}\rfloor, Y - \lfloor\frac{k}{2}\rfloor : Y + \lfloor\frac{k}{2}\rfloor\right)\right)$$





Linear Filters: **Properties**

Let $\otimes$ denote convolution. Let $I(X, Y)$ be a digital image

Superposition: Let F_1 and F_2 be digital filters

$$(F_1 + F_2) \otimes I(X, Y) = F_1 \otimes I(X, Y) + F_2 \otimes I(X, Y)$$

Scaling: Let F be digital filter and let k be a scalar

$$(kF) \otimes I(X, Y) = F \otimes (kI(X, Y)) = k(F \otimes I(X, Y))$$

Shift Invariance: Output is local (i.e., no dependence on absolute position)

An operation is **linear** if it satisfies both **superposition** and **scaling**

Linear Systems: Characterization Theorem

Any linear, shift invariant operation can be expressed as convolution

Up until now…

— The **correlation** of $F(X, Y)$ and $I(X, Y)$ is:

$$I'(X,Y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} F(i,j) I(X+i, Y+j)$$

output | filter | image (signal)

— **Visual interpretation**: Superimpose the filter F on the image I at (X, Y), perform an element-wise multiply, and sum up the values

— **Convolution** is like **correlation** except filter rotated 180°

if $F(X, Y) = F(-X, -Y)$ then correlation = convolution.

Up until now…

Ways to handle **boundaries**

- **Ignore/discard**. Make the computation undefined for top/bottom k rows and left/right-most k columns
- **Pad with zeros**. Return zero whenever a value of I is required beyond the image bounds
- **Assume periodicity.** Top row wraps around to the bottom row; leftmost column wraps around to rightmost column.

Simple **examples** of filtering:

- copy, shift, smoothing, sharpening

Linear filter **properties**:

- superposition, scaling, shift invariance

Characterization Theorem: Any linear, shift-invariant operation can be expressed as a convolution

Smoothing

Smoothing (or blurring) is an important operation in a lot of computer vision

— Captured images are naturally **noisy**, smoothing allows removal of noise

— It is important for **re-scaling** of images, to avoid sampling artifacts

— Fake image **defocus** (e.g., depth of field) for artistic effects

(many other uses as well)

Smoothing with a **Box Filter**

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

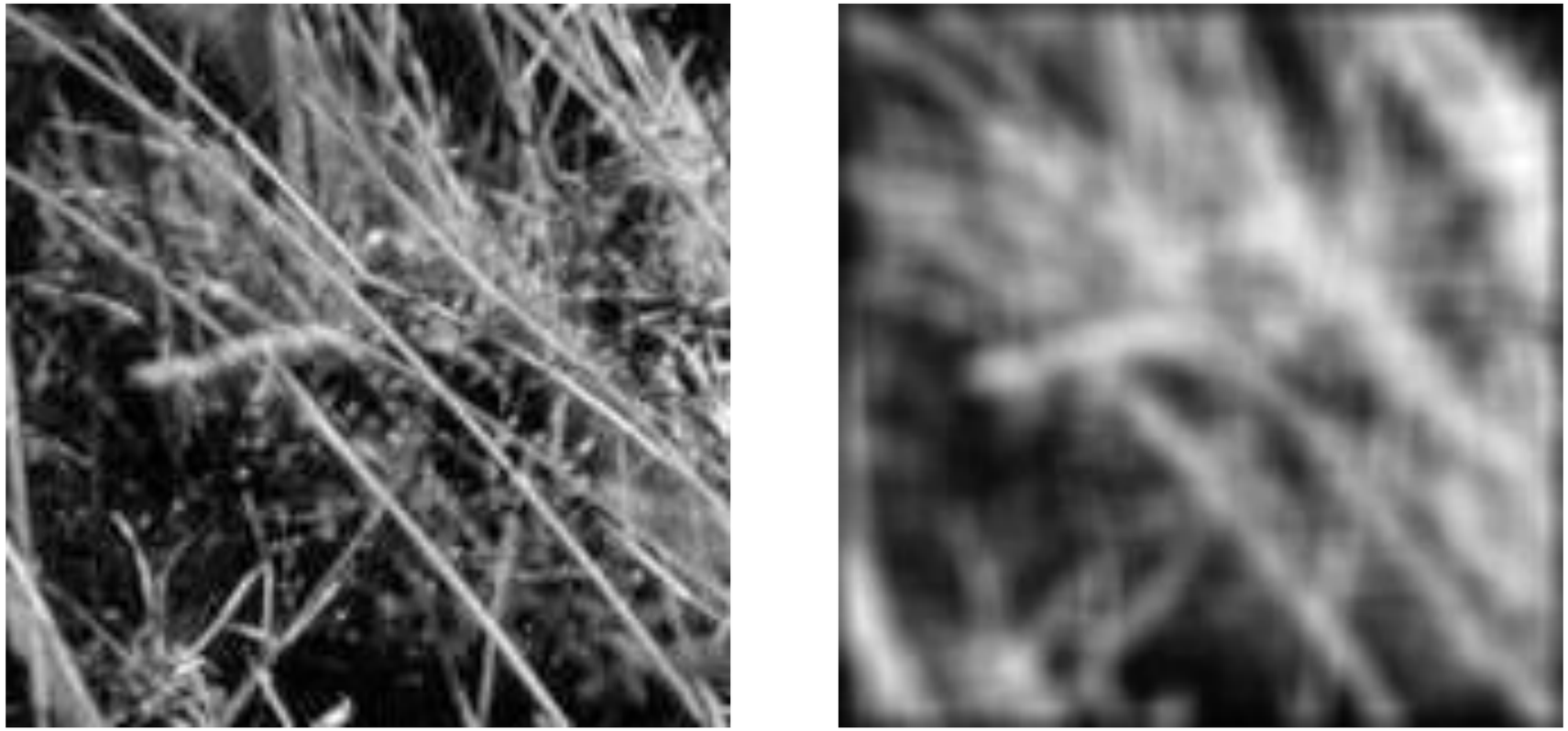
Image Credit: Ioannis (Yannis) Gkioulekas (CMU)

Filter has equal positive values that some up to 1

Replaces each pixel with the average of itself and its local neighborhood

— Box filter is also referred to as **average filter** or **mean filter**

Smoothing with a **Box Filter**

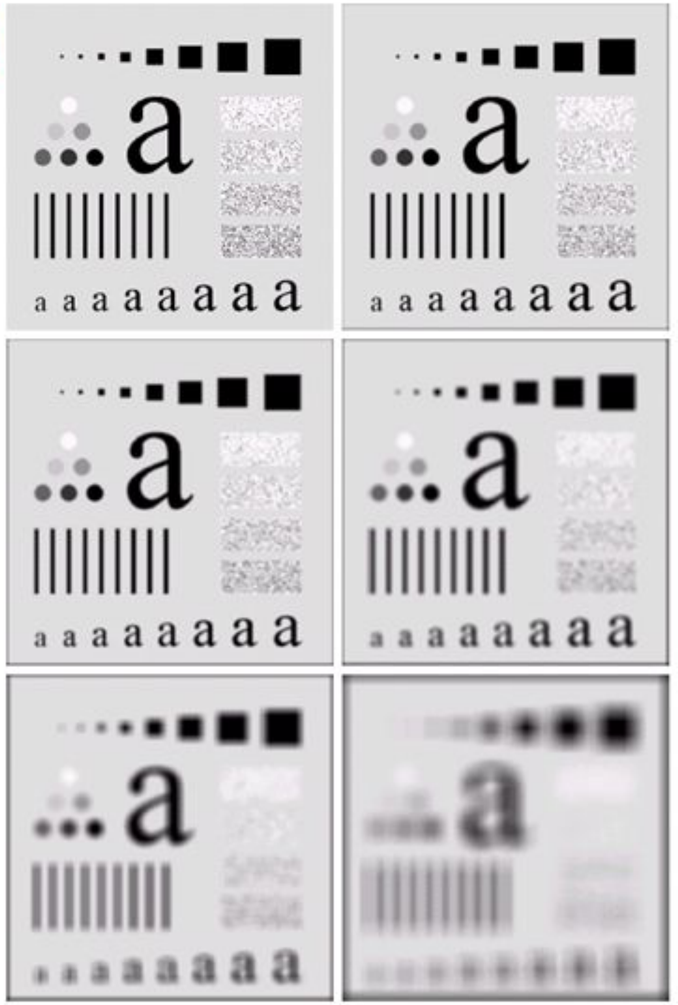


Forsyth & Ponce (2nd ed.) Figure 4.1 (left and middle)

Smoothing with a **Box Filter**

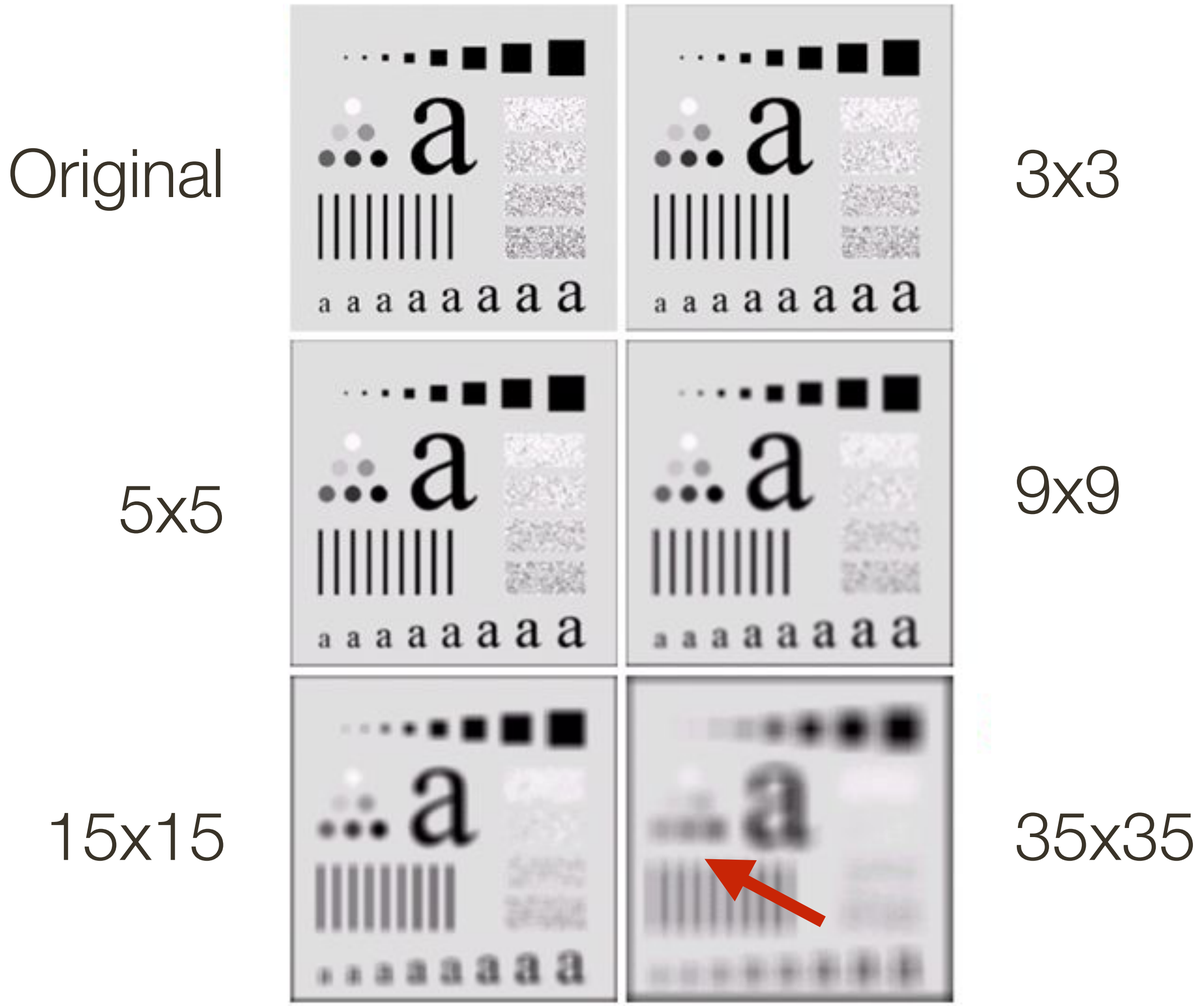
What happens if we increase the width (size) of the box filter?

Smoothing with a **Box Filter**



Gonzales & Woods (3rd ed.) Figure 3.3

Smoothing with a **Box Filter**



Gonzales & Woods (3rd ed.) Figure 3.3

Smoothing with a **Box Filter**

Smoothing with a box **doesn’t model lens defocus** well

- Smoothing with a box filter depends on direction
- Image in which the center point is 1 and every other point is 0

Smoothing with a **Box Filter**

Smoothing with a box **doesn't model lens defocus** well

- Smoothing with a box filter depends on direction
- Image in which the center point is 1 and every other point is 0

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

Filter

0	0	0	0	0
0	0	0	0	0
0	0	1	0	0
0	0	0	0	0
0	0	0	0	0

Image

Smoothing with a **Box Filter**

Smoothing with a box **doesn't model lens defocus** well

- Smoothing with a box filter depends on direction
- Image in which the center point is 1 and every other point is 0

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

Filter

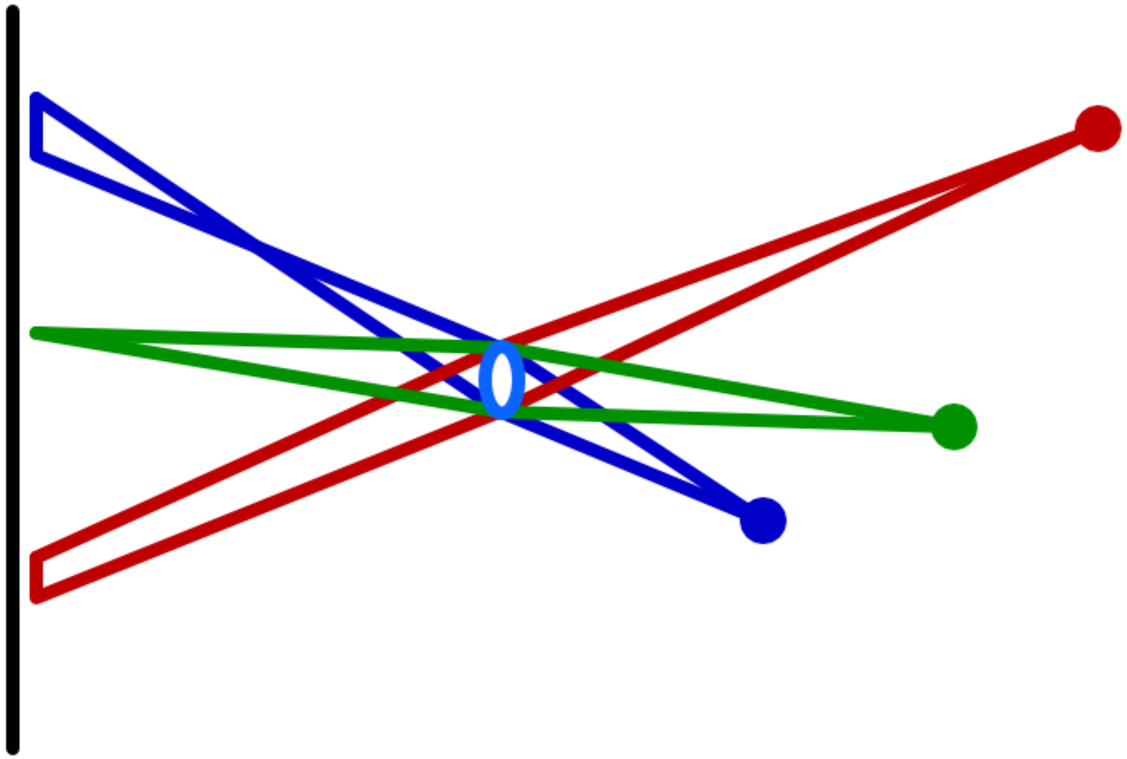
0	0	0	0	0
0	0	0	0	0
0	0	1	0	0
0	0	0	0	0
0	0	0	0	0

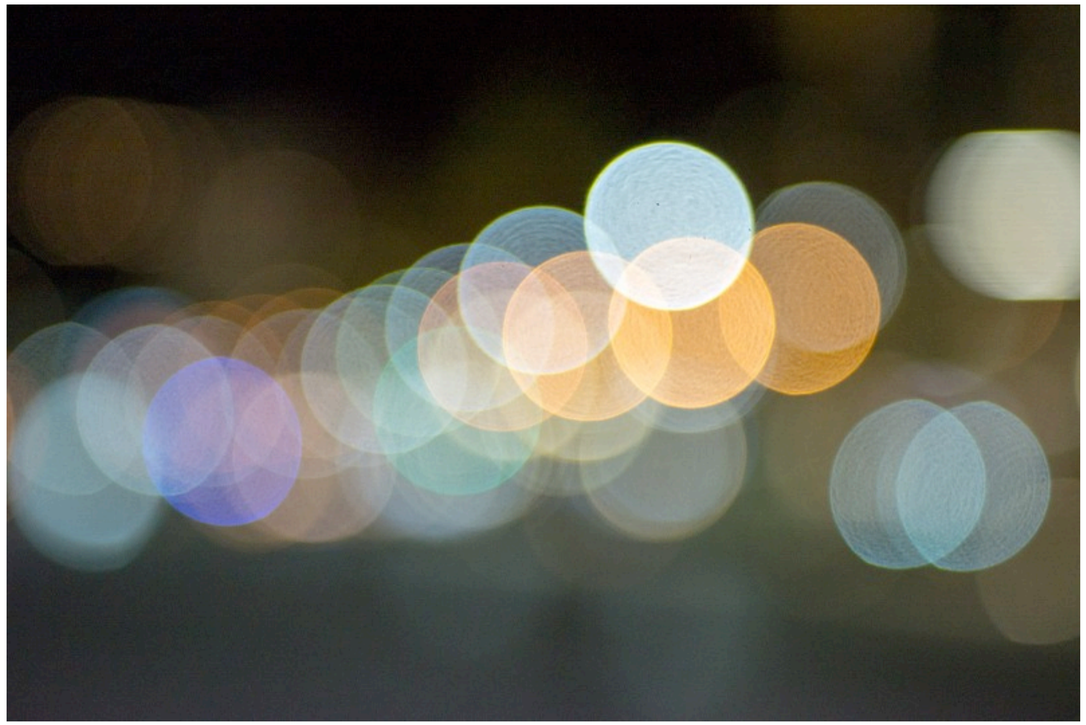
Image

0	0	0	0	0
0	$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$	0
0	$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$	0
0	$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$	0
0	0	0	0	0

Result

Smoothing: **Circular** Kernel





* image credit: https://catlikecoding.com/unity/tutorials/advanced-rendering/depth-of-field/circle-of-confusion/lens-camera.png

Smoothing

Smoothing with a box **doesn’t model lens defocus** well

— Smoothing with a box filter depends on direction

— Image in which the center point is 1 and every other point is 0

Smoothing

Smoothing with a box **doesn’t model lens defocus** well

— Smoothing with a box filter depends on direction

— Image in which the center point is 1 and every other point is 0

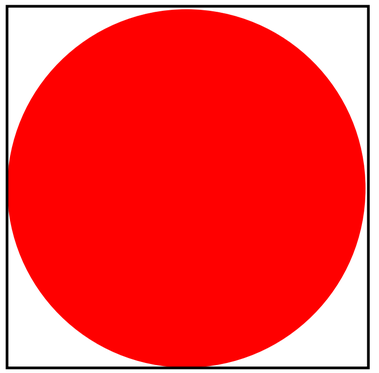
Smoothing with a (circular) **pillbox** is a better model for defocus (in geometric optics)

Pillbox Filter

Let the radius (i.e., half diameter) of the filter be r

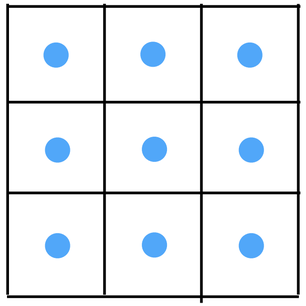
In a contentious domain, a 2D (circular) pillbox filter, $f(x, y)$, is defined as:

$$f(x,y) = \frac{1}{\pi r^2} \begin{cases} 1 & \text{if } \; x^2 + y^2 \leq r^2 \\ 0 & \text{otherwise} \end{cases}$$

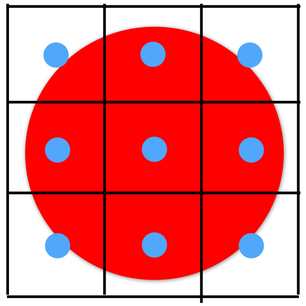


The scaling constant, $\frac{1}{\pi r^2}$, ensures that the area of the filter is one

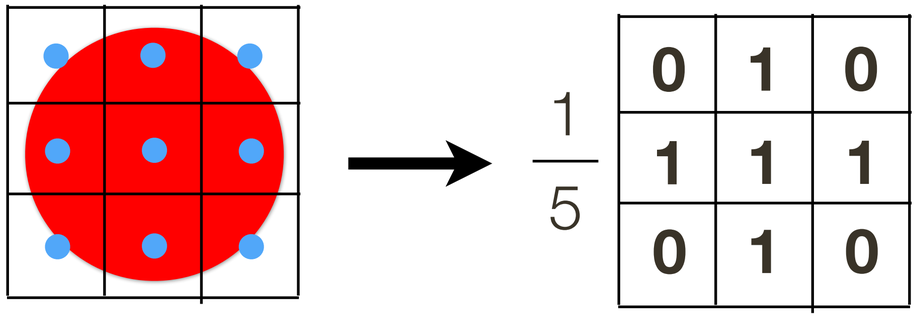
Pillbox Filter



Pillbox Filter

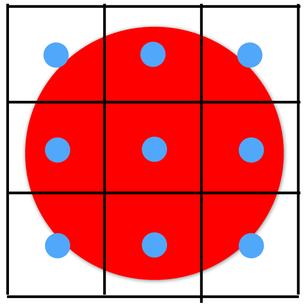


Pillbox Filter

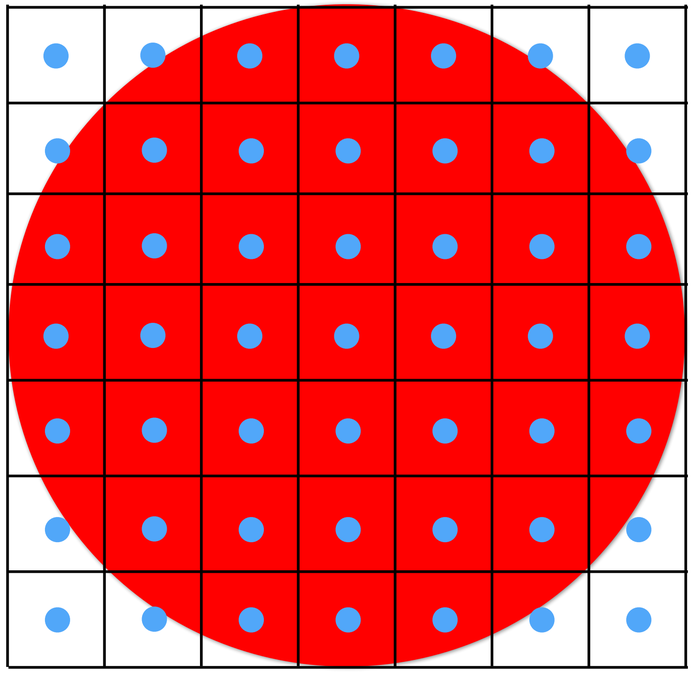


$$\frac{1}{5}\begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Pillbox Filter

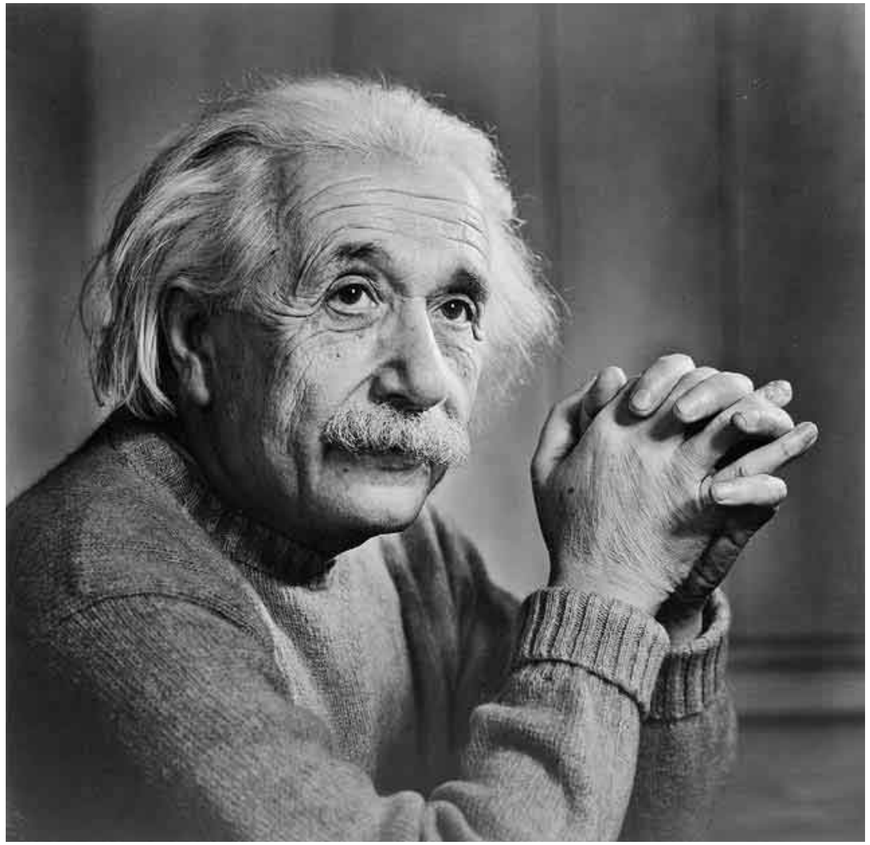


$$\frac{1}{5}\begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

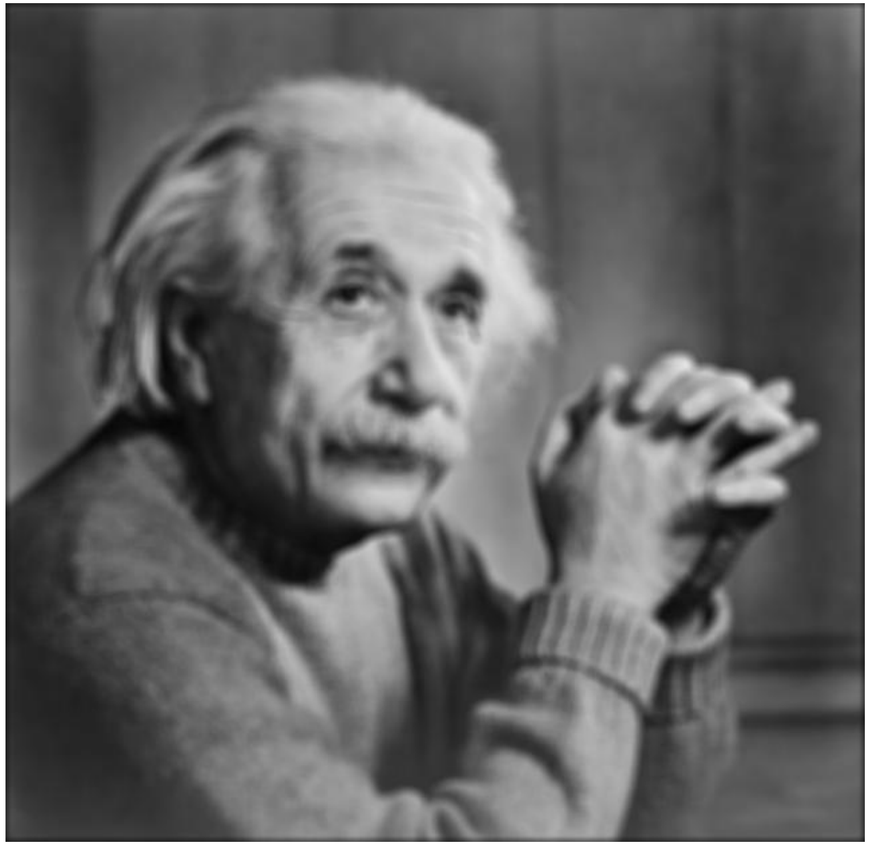


$$\frac{1}{37}\begin{bmatrix} 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 \end{bmatrix}$$

Pillbox Filter

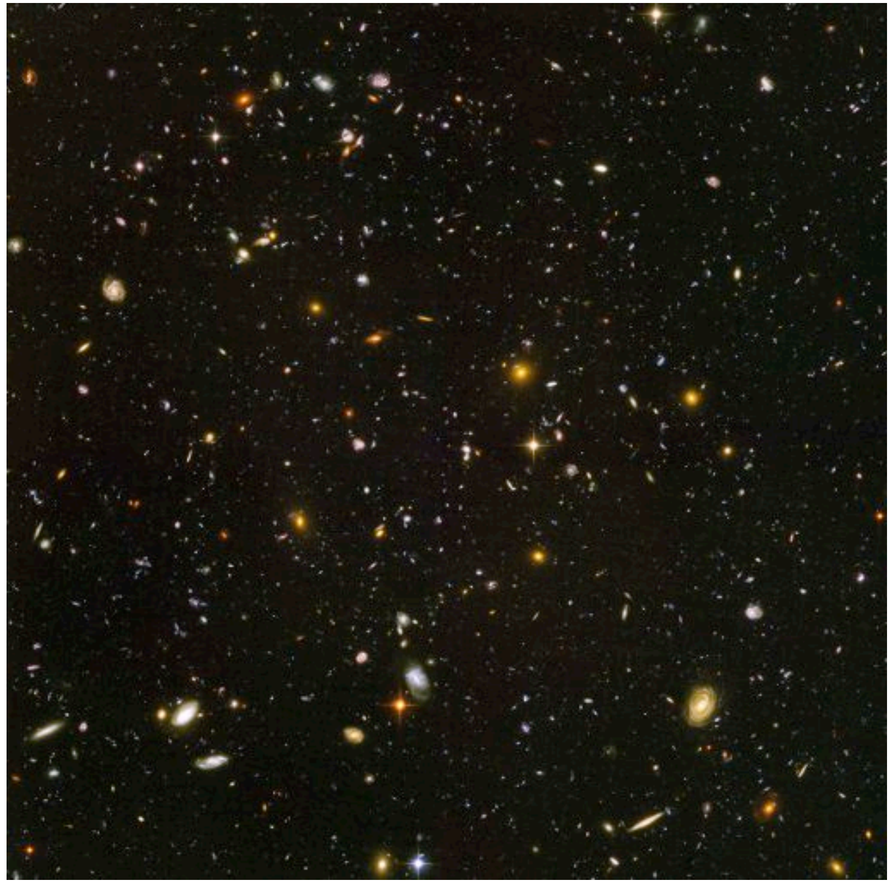


Original



11 x 11 Pillbox

Pillbox Filter



Hubble Deep View

With Circular Blur

Images: yehar.com

Smoothing

Smoothing with a box **doesn’t model lens defocus** well

— Smoothing with a box filter depends on direction

— Image in which the center point is 1 and every other point is 0

Smoothing with a (circular) **pillbox** is a better model for defocus (in geometric optics)

The **Gaussian** is a good general smoothing model

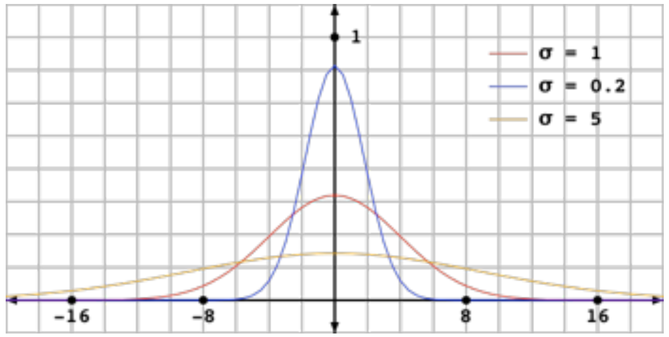
— for phenomena (that are the sum of other small effects)

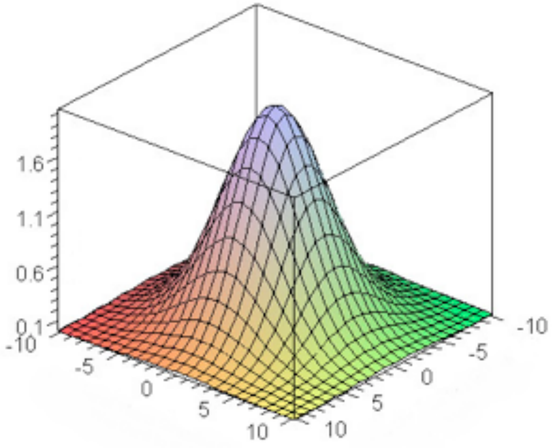
— whenever the Central Limit Theorem applies

Smoothing with a **Gaussian**

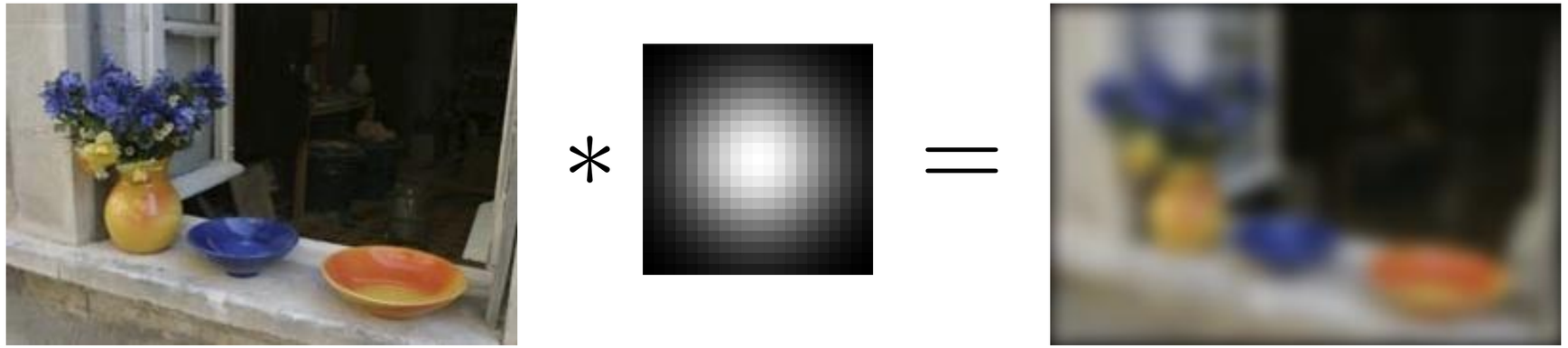
Gaussian kernels are often used for smoothing and resizing images

1D





2D

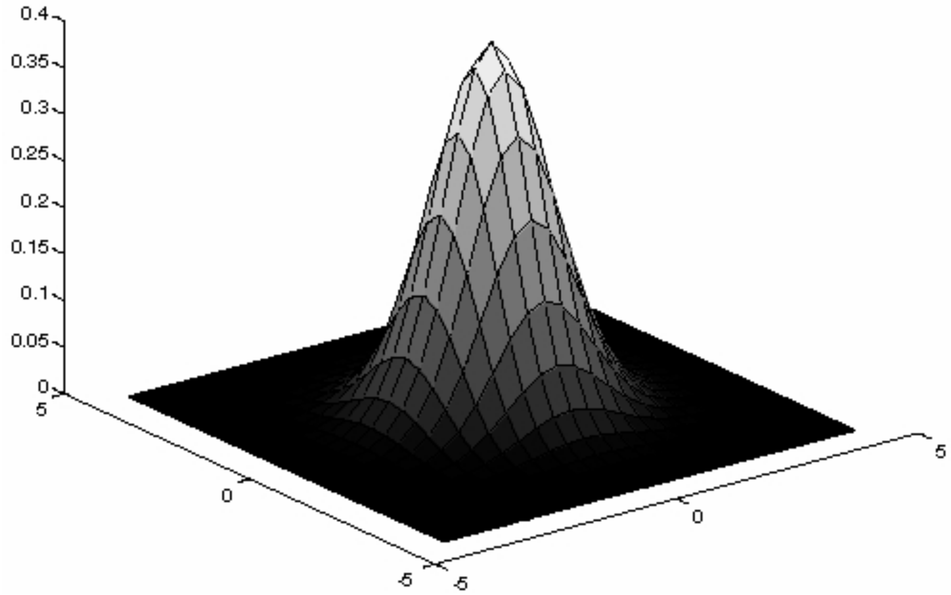


Smoothing with a **Gaussian**

Idea: Weight contributions of pixels by spatial proximity (nearness)

2D **Gaussian** (continuous case):

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



Forsyth & Ponce (2nd ed.)
Figure 4.2

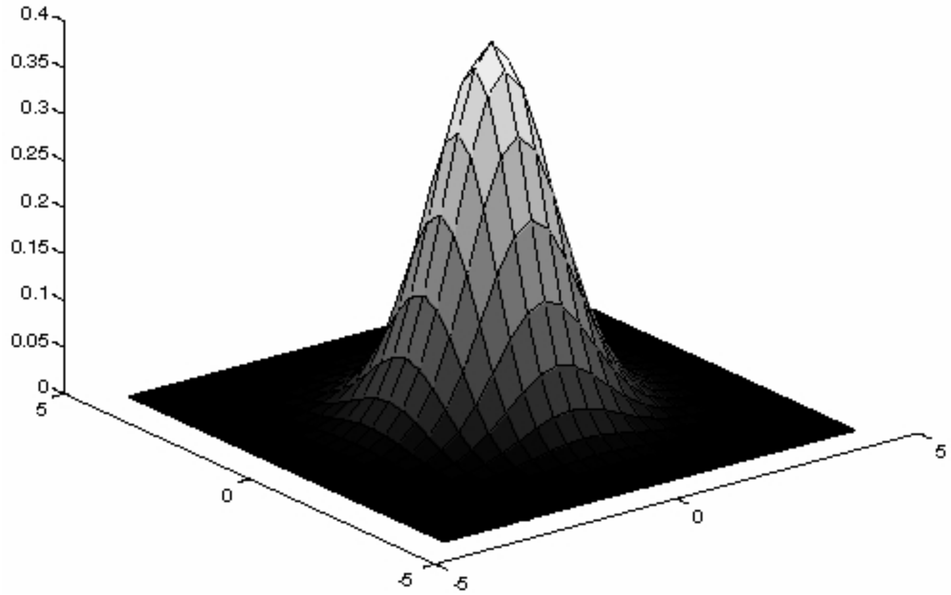
Smoothing with a **Gaussian**

Idea: Weight contributions of pixels by spatial proximity (nearness)

2D **Gaussian** (continuous case):

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

Standard Deviation



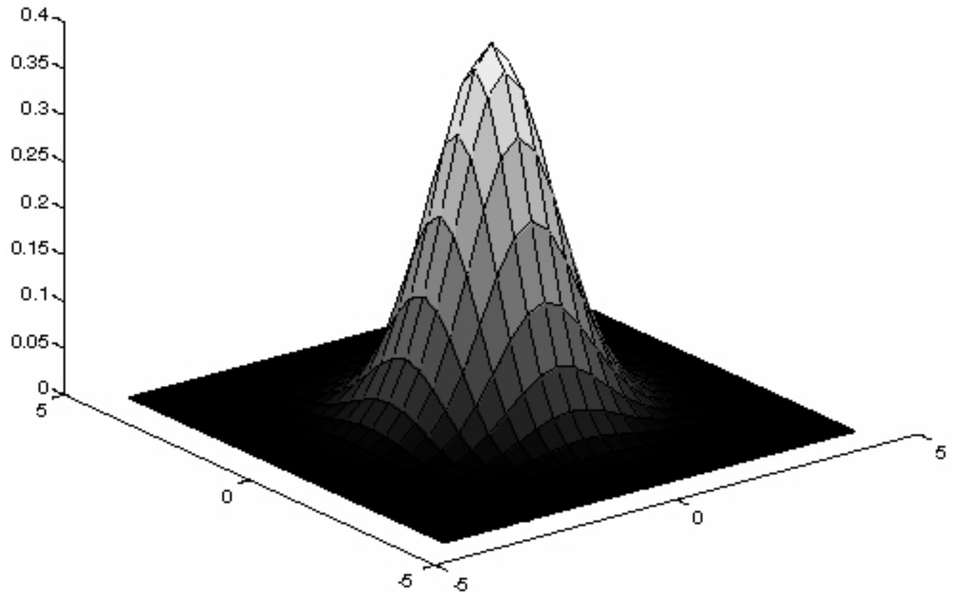
Forsyth & Ponce (2nd ed.)
Figure 4.2

Smoothing with a **Gaussian**

Idea: Weight contributions of pixels by spatial proximity (nearness)

2D **Gaussian** (continuous case):

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



Forsyth & Ponce (2nd ed.)
Figure 4.2

1. Define a continuous **2D function**

2. **Discretize it** by evaluating this function on the discrete pixel positions to obtain a filter

Smoothing with a **Gaussian**

Quantized an truncated **3x3 Gaussian** filter:

$G_\sigma(-1,1)$	$G_\sigma(0,1)$	$G_\sigma(1,1)$
$G_\sigma(-1,0)$	$G_\sigma(0,0)$	$G_\sigma(1,0)$
$G_\sigma(-1,-1)$	$G_\sigma(0,-1)$	$G_\sigma(1,-1)$

Smoothing with a **Gaussian**

Quantized an truncated **3x3 Gaussian** filter:

$G_\sigma(-1,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$
$G_\sigma(-1,0) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(0,0) = \frac{1}{2\pi\sigma^2}$	$G_\sigma(1,0) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$
$G_\sigma(-1,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$

Smoothing with a **Gaussian**

Quantized an truncated **3x3 Gaussian** filter:

$G_\sigma(-1,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$
$G_\sigma(-1,0) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(0,0) = \frac{1}{2\pi\sigma^2}$	$G_\sigma(1,0) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$
$G_\sigma(-1,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$

With $\sigma = 1$:

0.059	0.097	0.059
0.097	0.159	0.097
0.059	0.097	0.059

Smoothing with a **Gaussian**

Quantized an truncated **3x3 Gaussian** filter:

$G_\sigma(-1,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$
$G_\sigma(-1,0) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(0,0) = \frac{1}{2\pi\sigma^2}$	$G_\sigma(1,0) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$
$G_\sigma(-1,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$

With $\sigma = 1$:

0.059	0.097	0.059
0.097	0.159	0.097
0.059	0.097	0.059

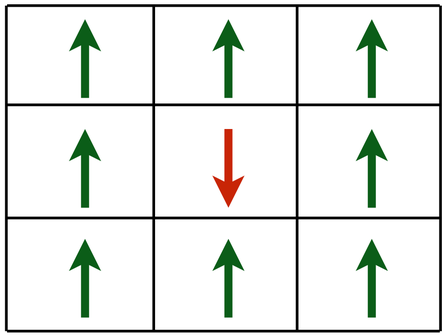
What happens if σ is larger?

Smoothing with a **Gaussian**

Quantized an truncated **3x3 Gaussian** filter:

$G_\sigma(-1,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$
$G_\sigma(-1,0) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(0,0) = \frac{1}{2\pi\sigma^2}$	$G_\sigma(1,0) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$
$G_\sigma(-1,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$

With $\sigma = 1$:



What happens if σ is larger?

— **More** blur

Smoothing with a **Gaussian**

Quantized an truncated **3x3 Gaussian** filter:

$G_\sigma(-1,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$
$G_\sigma(-1,0) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(0,0) = \frac{1}{2\pi\sigma^2}$	$G_\sigma(1,0) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$
$G_\sigma(-1,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$

With $\sigma = 1$:

0.059	0.097	0.059
0.097	0.159	0.097
0.059	0.097	0.059

What happens if σ is larger?

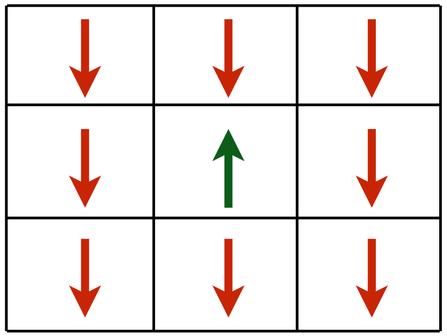
What happens if σ is smaller?

Smoothing with a **Gaussian**

Quantized an truncated **3x3 Gaussian** filter:

$G_\sigma(-1,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$
$G_\sigma(-1,0) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(0,0) = \frac{1}{2\pi\sigma^2}$	$G_\sigma(1,0) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$
$G_\sigma(-1,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$

With $\sigma = 1$:

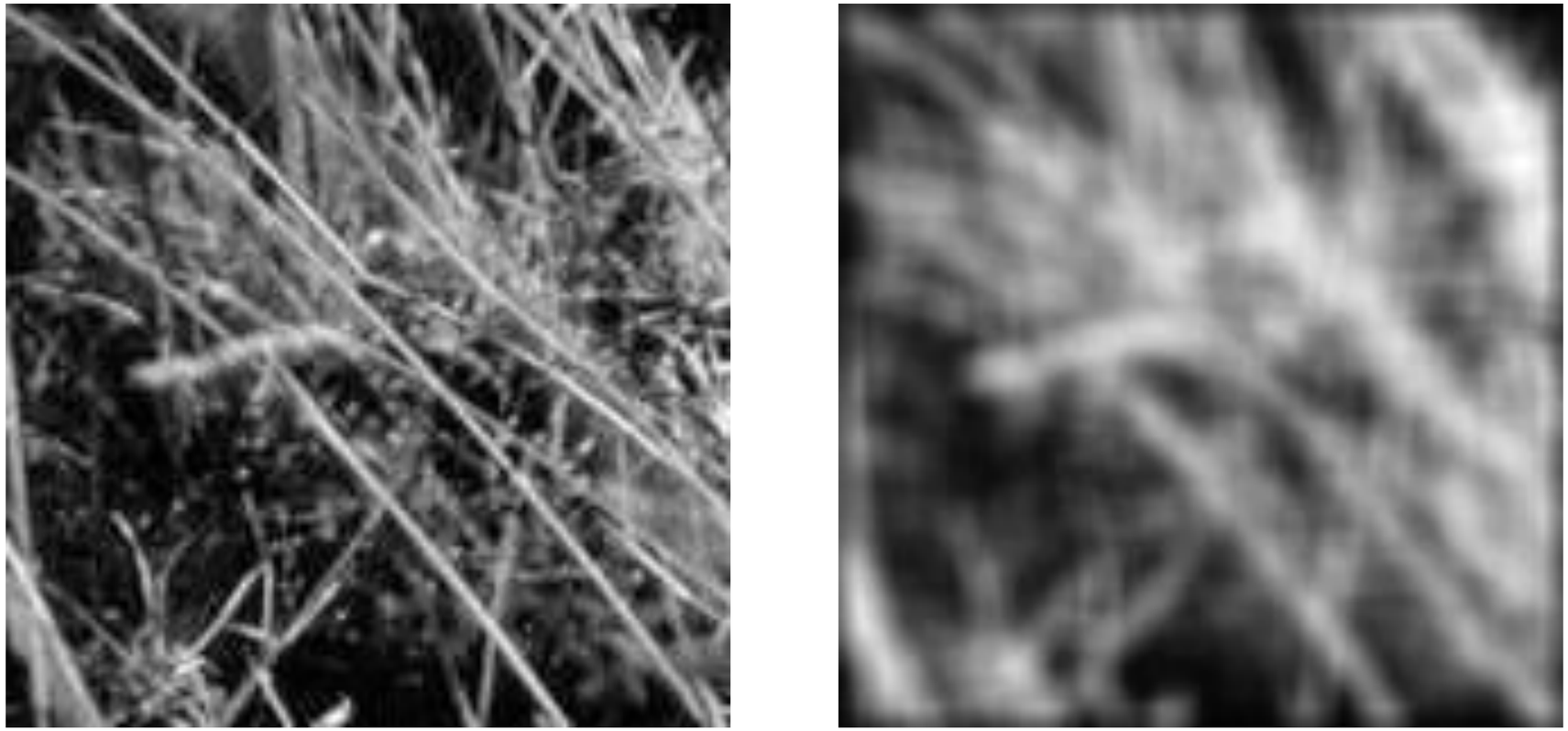


What happens if σ is larger?

What happens if σ is smaller?

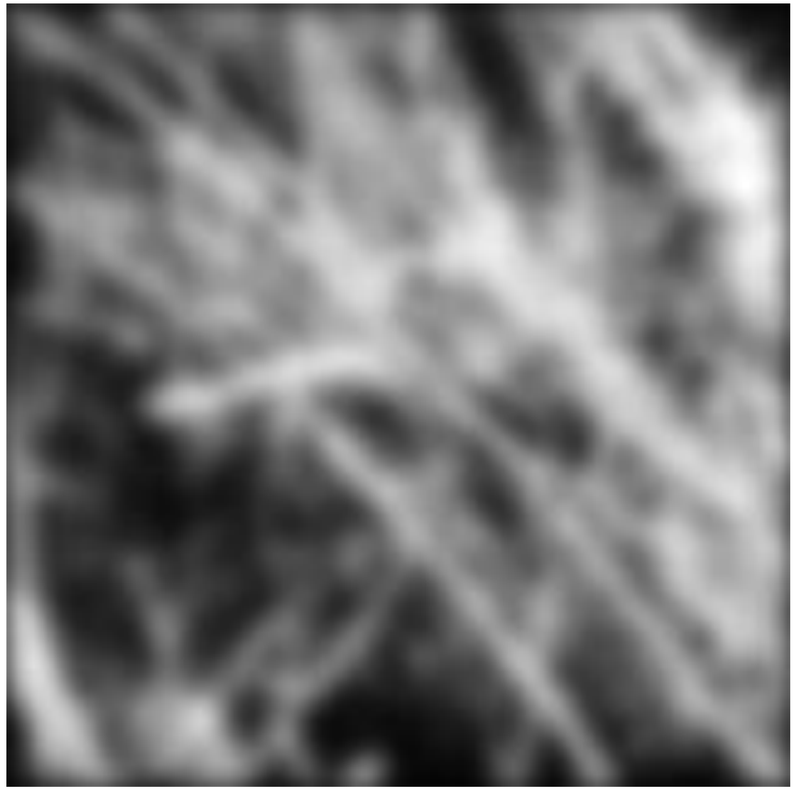
— **Less** blur

Smoothing with a **Box Filter**



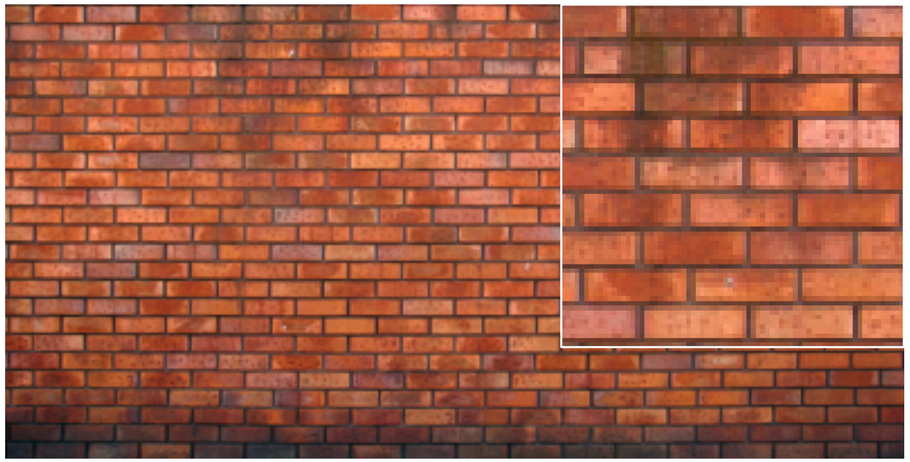
Forsyth & Ponce (2nd ed.) Figure 4.1 (left and middle)

Smoothing with a **Gaussian**

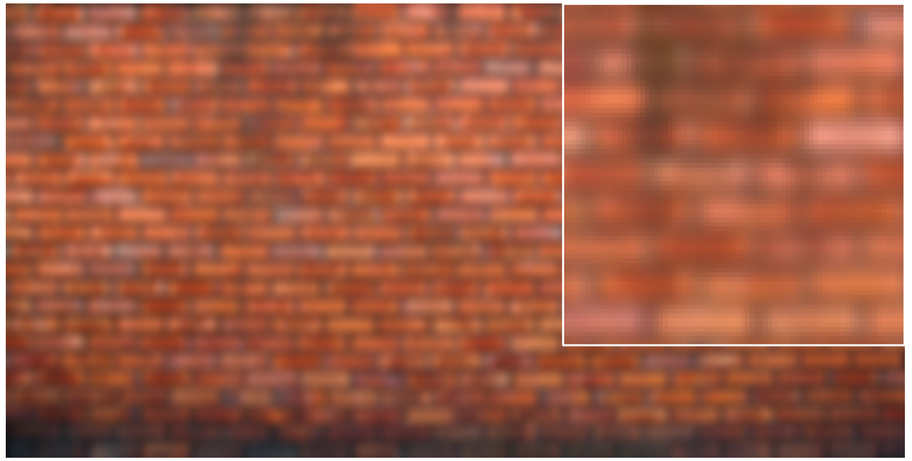


Forsyth & Ponce (2nd ed.) Figure 4.1 (left and right)

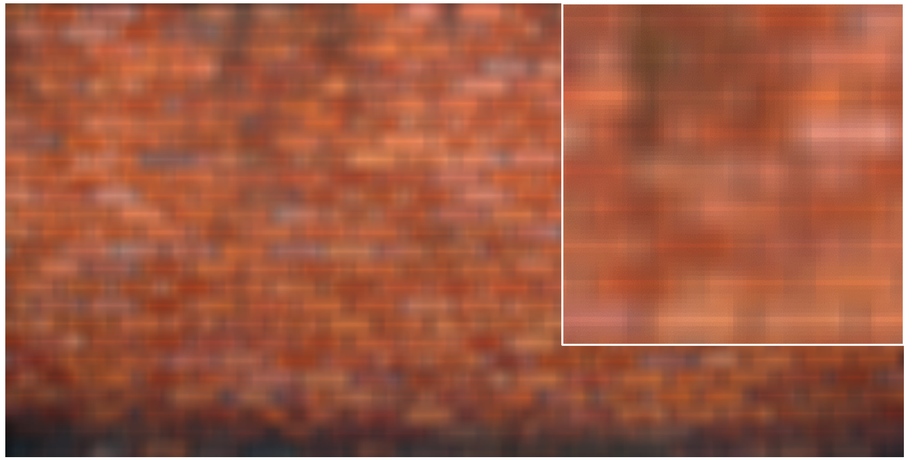
Box vs. **Gaussian** Filter



original



7x7 Gaussian



7x7 box

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)

Fun: How to get shadow effect?

University of British Columbia

Adopted from: Ioannis (Yannis) Gkioulekas (CMU)

Fun: How to get shadow effect?

University of British Columbia

Blur with a Gaussian kernel, then compose the blurred image with the original (with some offset)

Adopted from: Ioannis (Yannis) Gkioulekas (CMU)

Example 6: Smoothing with a Gaussian

Quantized an truncated **3x3 Gaussian** filter:

$G_\sigma(-1,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$
$G_\sigma(-1,0) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(0,0) = \frac{1}{2\pi\sigma^2}$	$G_\sigma(1,0) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$
$G_\sigma(-1,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$

With $\sigma = 1$:

0.059	0.097	0.059
0.097	0.159	0.097
0.059	0.097	0.059

What is the problem with this filter?

Example 6: Smoothing with a Gaussian

Quantized an truncated **3x3 Gaussian** filter:

$G_\sigma(-1,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$
$G_\sigma(-1,0) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(0,0) = \frac{1}{2\pi\sigma^2}$	$G_\sigma(1,0) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$
$G_\sigma(-1,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$	$G_\sigma(0,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{1}{2\sigma^2}}$	$G_\sigma(1,-1) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{2}{2\sigma^2}}$

With $\sigma = 1$:

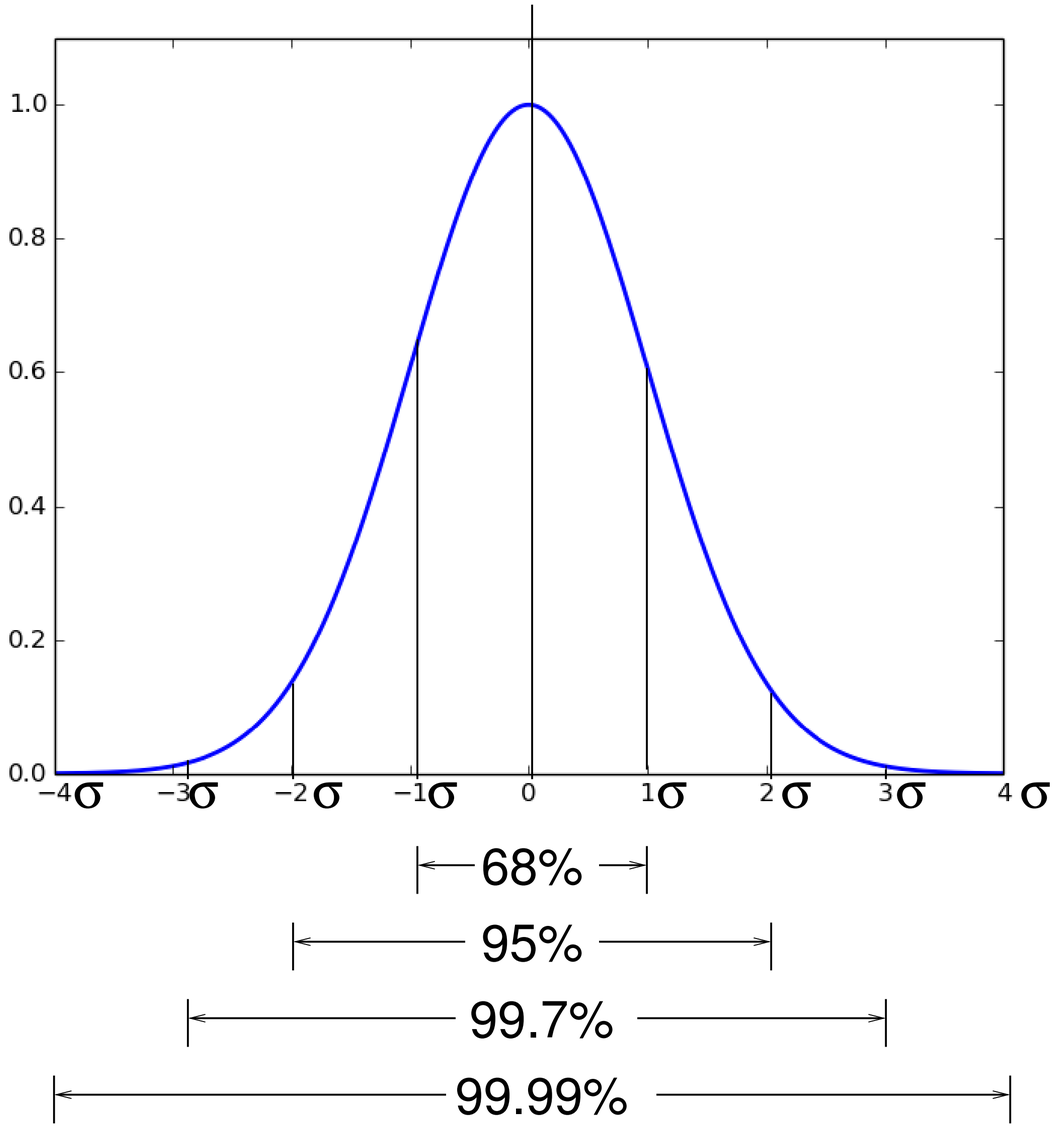
0.059	0.097	0.059
0.097	0.159	0.097
0.059	0.097	0.059

What is the problem with this filter?

does not sum to 1

truncated too much

Gaussian: Area Under the Curve



Smoothing with a **Gaussian**

With $\sigma = 1$:

0.059	0.097	0.059
0.097	0.159	0.097
0.059	0.097	0.059

Better version of the Gaussian filter:

- sums to 1 (normalized)
- captures $\pm 2\sigma$

$$\frac{1}{273}$$

1	4	7	4	1
4	16	26	16	4
7	26	41	26	7
4	16	26	16	4
1	4	7	4	1

In general, you want the Gaussian filter to capture $\pm 3\sigma$, for $\sigma = 1$ => 7x7 filter

Exercise

With $\sigma = 5$ what filter size would be appropriate?

Exercise

With $\sigma = 5$ what filter size would be appropriate?

$$\sigma * 6 = 5 * 6 = 30 => 31 \times 31$$

Smoothing **Summary**

Smoothing with a box **doesn't model lens defocus** well

— Smoothing with a box filter depends on direction

— Point spread function is a box

Smoothing with a (circular) **pillbox** is a better model for defocus (in geometric optics)

The **Gaussian** is a good general smoothing model

— for phenomena (that are the sum of other small effects)

— whenever the Central Limit Theorem applies (avg of many independent rvs → normal dist)

Lets talk about **efficiency**

Efficient Implementation: **Separability**

A 2D function of x and y is **separable** if it can be written as the product of two functions, one a function only of x and the other a function only of y

Both the **2D box filter** and the **2D Gaussian filter** are **separable**

Both can be implemented as two 1D convolutions:

- First, convolve each row with a 1D filter
- Then, convolve each column with a 1D filter
- Aside: or vice versa

The **2D Gaussian** is the only (non trivial) 2D function that is both separable and rotationally invariant.

Separability: Box Filter Example

Standard (3x3)

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

$$F(X,Y) = F(X)F(Y)$$

filter

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	0	10	20	30	30	30	20	10	
	10	10	10	10	0	0	0	0	
	10	30	10	10	0	0	0	0	

Separability: Box Filter Example

Standard (3x3)

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

$$F(X,Y) = F(X)F(Y)$$

filter

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	0	10	20	30	30	30	20	10	
	10	10	10	10	0	0	0	0	
	10	30	10	10	0	0	0	0	

Separable

image $I(X,Y)$

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

$F(X)$

filter

$\frac{1}{3}$

1	1	1

	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	
	0	30	60	90	90	90	60	30	
	0	30	60	90	90	90	60	30	
	0	30	30	60	60	90	60	30	
	0	30	60	90	90	90	60	30	
	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	
	30	30	30	30	0	0	0	0	
	0	0	0	0	0	0	0	0	

Separability: Box Filter Example

Standard (3x3)

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

$$F(X,Y) = F(X)F(Y)$$

filter

$\frac{1}{9}$

1	1	1
1	1	1
1	1	1

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	0	10	20	30	30	30	20	10	
	10	10	10	10	0	0	0	0	
	10	30	10	10	0	0	0	0	

Separable

image $I(X,Y)$

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

$F(X)$

filter

$\frac{1}{3}$

1	1	1

	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	
	0	30	60	90	90	90	60	30	
	0	30	60	90	90	90	60	30	
	0	30	30	60	60	90	60	30	
	0	30	60	90	90	90	60	30	
	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	
	30	30	30	30	0	0	0	0	
	0	0	0	0	0	0	0	0	

$F(Y)$

filter

$\frac{1}{3}$

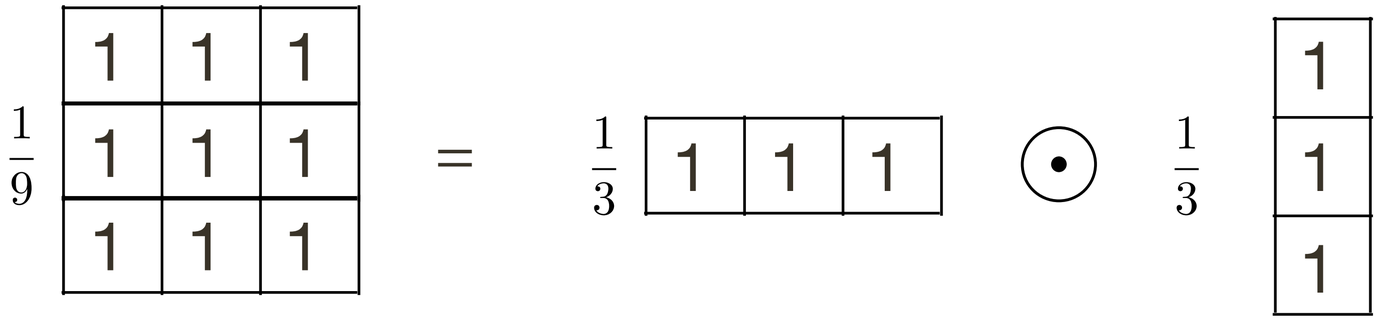
1
1
1

output $I'(X,Y)$

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	0	10	20	30	30	30	20	10	
	10	10	10	10	0	0	0	0	
	10	30	10	10	0	0	0	0	

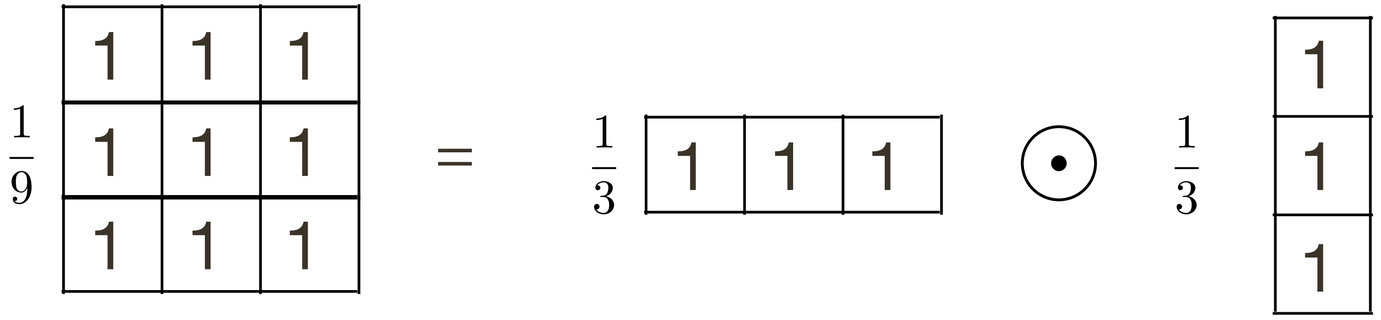
Separability: How do you know if filter is separable?

If a 2D filter can be expressed as an outer product of two 1D filters



Separability: How do you know if filter is separable?

Mathematically: Rank of filter matrix is 1 (recall rank is number of linearly independent row vectors)



Efficient Implementation: **Separability**

For example, recall the 2D **Gaussian**:

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

The 2D Gaussian can be expressed as a product of two functions, one a function of x and another a function of y

Efficient Implementation: **Separability**

For example, recall the 2D **Gaussian**:

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

$$= \left(\frac{1}{\sqrt{2\pi}\sigma} \exp^{-\frac{x^2}{2\sigma^2}} \right) \left(\frac{1}{\sqrt{2\pi}\sigma} \exp^{-\frac{y^2}{2\sigma^2}} \right)$$

function of x | function of y

The 2D Gaussian can be expressed as a product of two functions, one a function of x and another a function of y

Efficient Implementation: **Separability**

For example, recall the 2D **Gaussian**:

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

$$= \left(\frac{1}{\sqrt{2\pi}\sigma}\exp^{-\frac{x^2}{2\sigma^2}}\right)\left(\frac{1}{\sqrt{2\pi}\sigma}\exp^{-\frac{y^2}{2\sigma^2}}\right)$$

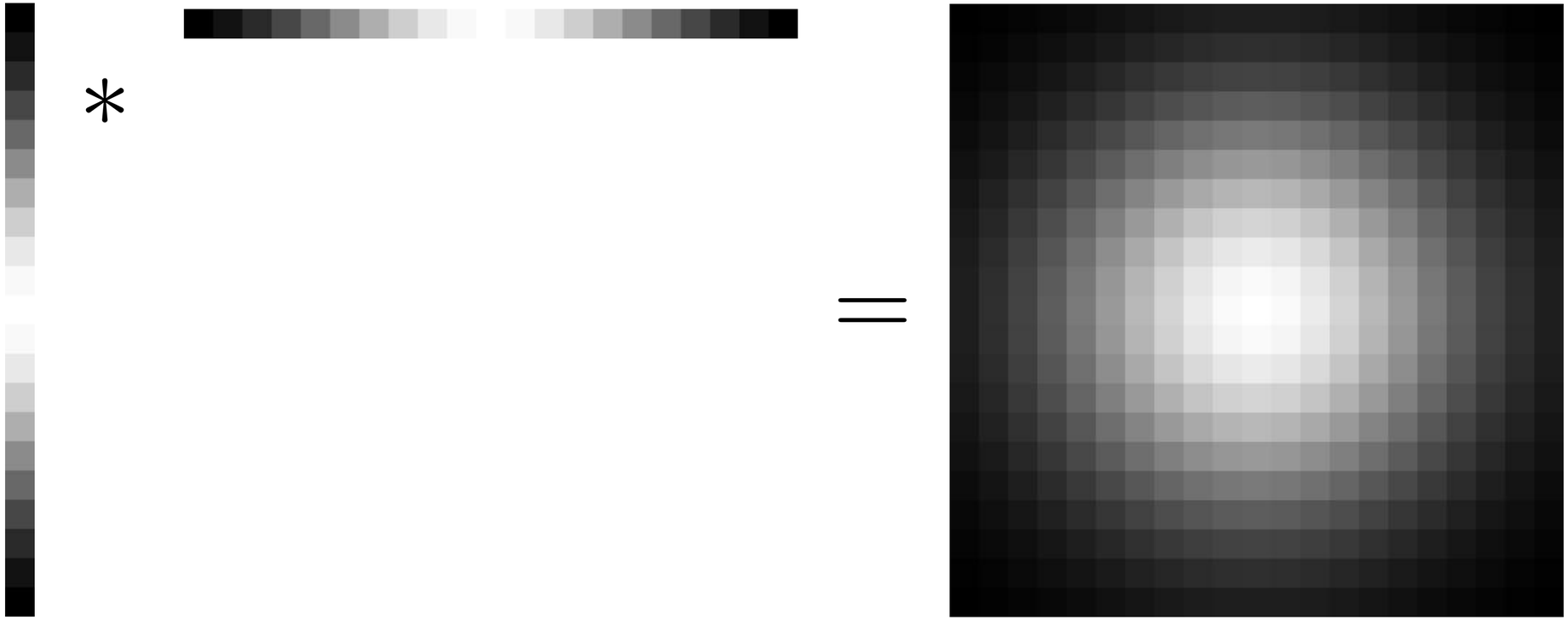
function of x | function of y

The 2D Gaussian can be expressed as a product of two functions, one a function of x and another a function of y

In this case the two functions are (identical) 1D Gaussians

Gaussian Blur

2D Gaussian filter can be thought of as an **outer product** or **convolution** of row and column filters



Example: Separable Gaussian Filter

$$\frac{1}{16}\begin{array}{|c|c|c|c|c|}\hline 1 & 4 & 6 & 4 & 1 \\ \hline \end{array} \otimes \frac{1}{16}\begin{array}{|c|}\hline 1 \\ \hline 4 \\ \hline 6 \\ \hline 4 \\ \hline 1 \\ \hline \end{array} = \frac{1}{256}\begin{array}{|c|c|c|c|c|}\hline 1 & 4 & 6 & 4 & 1 \\ \hline 4 & 16 & 24 & 16 & 4 \\ \hline 6 & 24 & 36 & 24 & 6 \\ \hline 4 & 16 & 24 & 16 & 4 \\ \hline 1 & 4 & 6 & 4 & 1 \\ \hline \end{array}$$

Example: Separable Gaussian Filter

$$\frac{1}{16}\begin{array}{|c|c|c|c|c|}\hline 1 & 4 & 6 & 4 & 1 \\ \hline \end{array} \otimes \frac{1}{16}\begin{array}{|c|}\hline 1 \\ \hline 4 \\ \hline 6 \\ \hline 4 \\ \hline 1 \\ \hline \end{array} = \frac{1}{256}\begin{array}{|c|c|c|c|c|}\hline 1 & 4 & 6 & 4 & 1 \\ \hline 4 & 16 & 24 & 16 & 4 \\ \hline 6 & 24 & 36 & 24 & 6 \\ \hline 4 & 16 & 24 & 16 & 4 \\ \hline 1 & 4 & 6 & 4 & 1 \\ \hline \end{array}$$

Efficient Implementation: **Separability**

Naive implementation of 2D **Gaussian**:

At each pixel, (X, Y), there are $m \times m$ multiplications

There are $n \times n$ pixels in (X, Y)

Total: $m^2 \times n^2$ multiplications

Efficient Implementation: **Separability**

Naive implementation of 2D **Gaussian**:

At each pixel, (X, Y), there are	$m \times m$ multiplications
There are	$n \times n$ pixels in (X, Y)
Total:	$m^2 \times n^2$ multiplications

Separable 2D **Gaussian**:

Efficient Implementation: **Separability**

Naive implementation of 2D **Gaussian**:

At each pixel, (X, Y), there are	$m \times m$ multiplications
There are	$n \times n$ pixels in (X, Y)
Total:	$m^2 \times n^2$ multiplications

Separable 2D **Gaussian**:

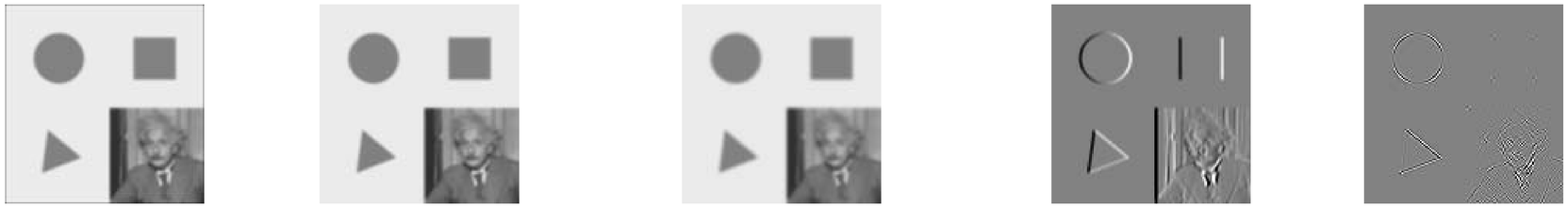
At each pixel, (X, Y), there are	$2m$ multiplications
There are	$n \times n$ pixels in (X, Y)
Total:	$2m \times n^2$ multiplications

Separable Filtering

Several useful filters can be applied as independent row and column operations

$$\frac{1}{K^2}\begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & 1 & \cdots & 1 \\ \vdots & \vdots & 1 & \vdots \\ 1 & 1 & \cdots & 1 \end{bmatrix} \quad \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \quad \frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix} \quad \frac{1}{8}\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad \frac{1}{4}\begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix}$$

$$\frac{1}{K}\begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix} \quad \frac{1}{4}\begin{bmatrix} 1 & 2 & 1 \end{bmatrix} \quad \frac{1}{16}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \end{bmatrix} \quad \frac{1}{2}\begin{bmatrix} -1 & 0 & 1 \end{bmatrix} \quad \frac{1}{2}\begin{bmatrix} 1 & -2 & 1 \end{bmatrix}$$



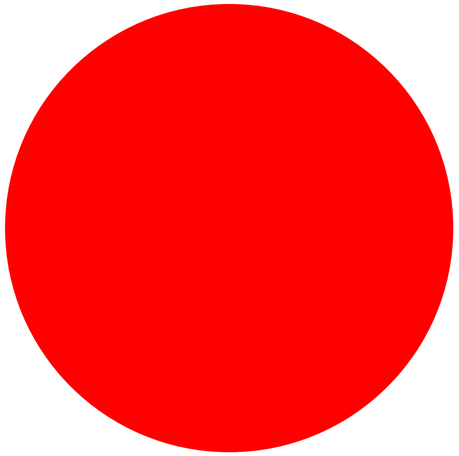
(a) box, $K = 5$ (b) bilinear (c) “Gaussian” (d) Sobel (e) corner

Sepprable?

Box Filter

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

Pillbox Filter



Gaussian Filter

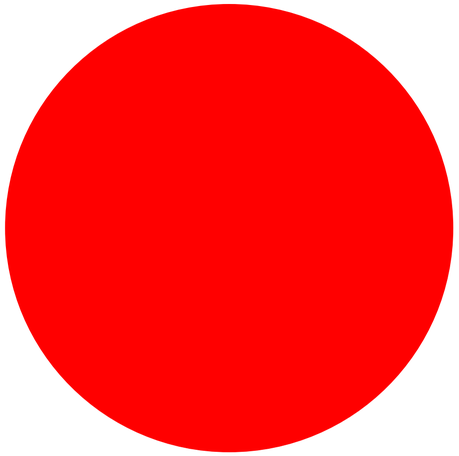
$$\frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

Rotationally Invariant?

Box Filter

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

Pillbox Filter



Gaussian Filter

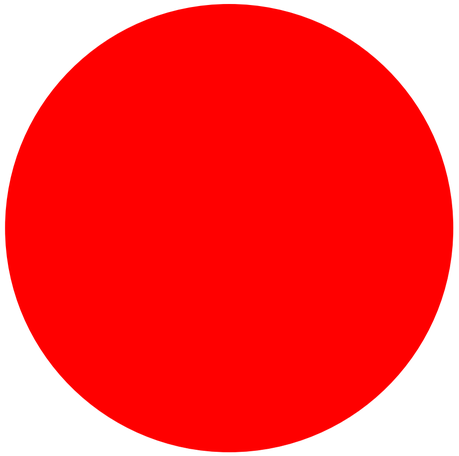
$$\frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

Low-pass Filtering = “Smoothing”

Box Filter

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

Pillbox Filter



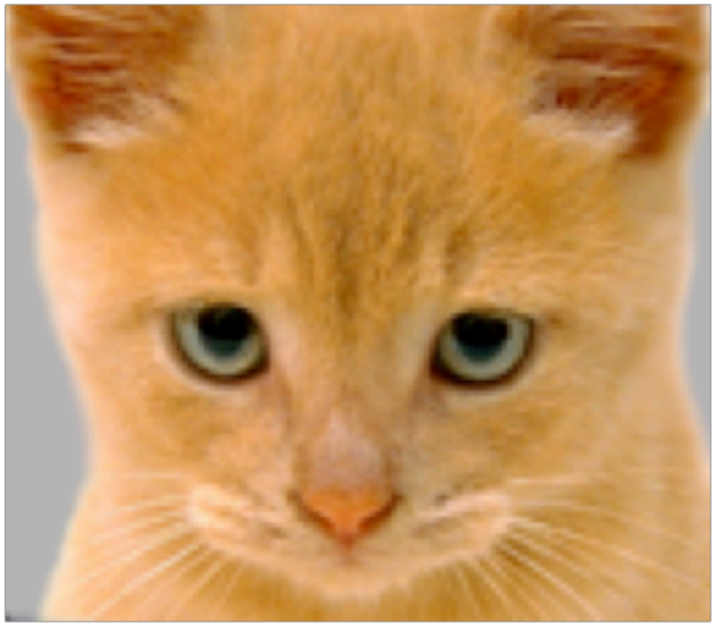
Gaussian Filter

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

All of these filters are **Low-pass Filters**

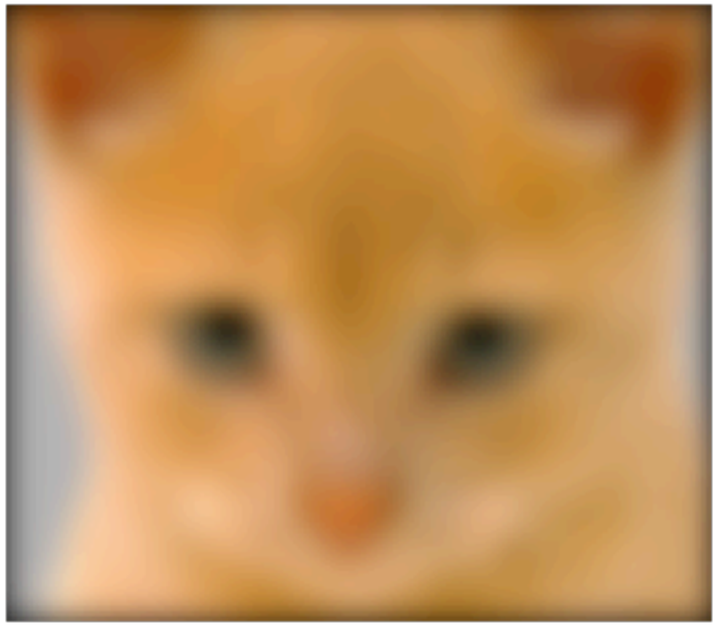
Low-pass filter: Low pass filter filters out all of the high frequency content of the image, only low frequencies remain

Assignment 1: **Low/High Pass** Filtering



Original

$$I(x, y)$$



Low-Pass Filter

$$I(x, y) * g(x, y)$$

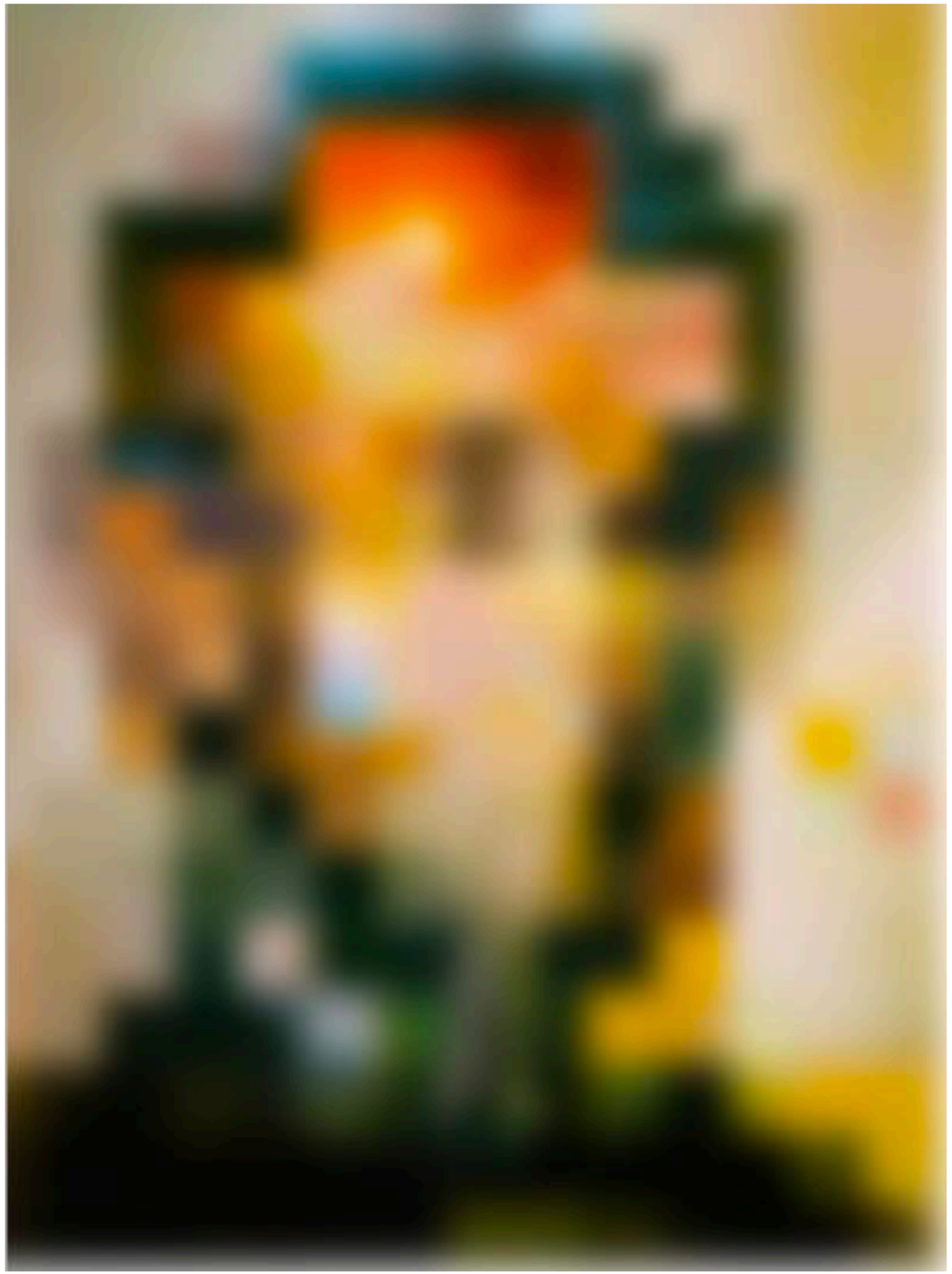
High-Pass Filter

$$I(x, y) - I(x, y) * g(x, y)$$

Gala Contemplating the Mediterranean Sea Which at Twenty Meters Becomes the Portrait of Abraham Lincoln (Homage to Rothko)

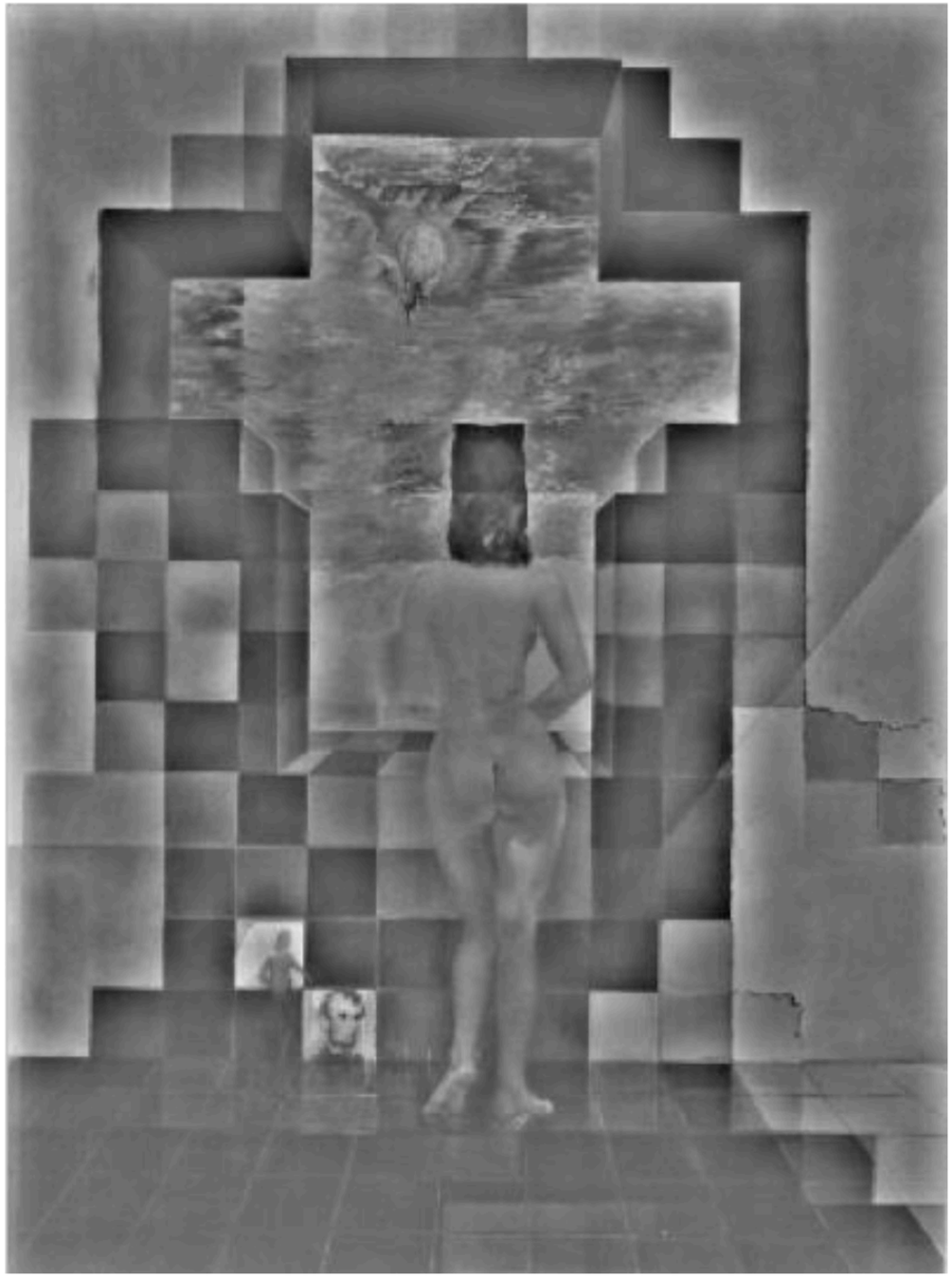
Salvador Dali, 1976

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)



Low-pass filtered version

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)



High-pass filtered version

Slide Credit: Ioannis (Yannis) Gkioulekas (CMU)