Georgia Institute of Technology

Computational Photography

Dr. Irfan Essa
Professor
School of Interactive Computing

Study the basics of computation and its impact on the entire workflow of photography, from capturing, manipulating and collaborating on, and sharing photographs.

Georgia Institute of Technology

Image Processing and Filtering, via Convolution and Cross-Correlation

Dr. Irfan Essa
Professor
School of Interactive Computing

Point-process and Neighboring Pixels Computations on an Image using Cross-Correlation and Convolution

Lesson Objectives

- ★ Describe in your own words the Cross-Correlation method for filtering images.
- ★ Describe in your own words the Convolution method for filtering images.
- ★ Identify at least two (2) differences between the Cross-Correlation and Convolution methods for filtering images.
- ★ Recall two (2) properties of Convolution methods for image filtering.

3

Review: Mathematical Representation for Smoothing

$$G[3,3] = \frac{1}{9}(A + B + C + D + E + F + G + H + I)$$

a	b	c
d	e	f
g	h	i

20	20	10	20	10	20	10	10	13
30	0	0	0	0	0	0	0	30
20	0	A	B	C	90	90	0	20
20	0	D	E	F	90	90	0	20
10	0	G	H	I	90	90	0	10
10	0	90	90	90	90	90	0	10
10	0	90	90	90	90	90	0	10
20	0	0	0	0	0	0	0	20
20	20	10	20	10	20	10	10	13

Review: Mathematical Representation for Smoothing

$$G[3,3] = \frac{1}{9}(A + B + C + D + E + F + G + H + I)$$

$$G[i,j] = \frac{1}{(2k+1)^2} \sum_{u=-k}^{k} \sum_{v=-k}^{k} F[i+u, j+v]$$

a	b	c
d	e	f
g	h	i

20	20	10	20	10	20	10	10	13
30	0	0	0	0	0	0	0	30
20	0	A	B	C	90	90	0	20
20	0	D	E	F	90	90	0	20
10	0	G	H	I	90	90	0	10
10	0	90	90	90	90	90	0	10
10	0	90	90	90	90	90	0	10
20	0	0	0	0	0	0	0	20
20	20	10	20	10	20	10	10	13

Review: Mathematical Representation for Smoothing

$$G[3,3] = \frac{1}{9}(A + B + C + D + E + F + G + H + I)$$

$$G[i,j] = \frac{1}{(2k+1)^2} \sum_{u=-k}^{k} \sum_{v=-k}^{k} F[i+u, j+v]$$

Loop over all pixels in neighborhood around image pixel *F[i,j]*

a	b	c
d	e	f
g	h	i

20	20	10	20	10	20	10	10	13
30	0	0	0	0	0	0	0	30
20	0	A	B	C	90	90	0	20
20	0	D	E	F	90	90	0	20
10	0	G	H	I	90	90	0	10
10	0	90	90	90	90	90	0	10
10	0	90	90	90	90	90	0	10
20	0	0	0	0	0	0	0	20
20	20	10	20	10	20	10	10	13

Review: Mathematical Representation for Smoothing

$$G[3,3] = \frac{1}{9}(A + B + C + D + E + F + G + H + I)$$

$$G[i,j] = \frac{1}{(2k+1)^2} \sum_{u=-k}^{k} \sum_{v=-k}^{k} F[i+u, j+v]$$

Attribute uniform weight to each pixel

Loop over all pixels in neighborhood around image pixel *F[i,j]*

a	b	c
d	e	f
g	h	i

20	20	10	20	10	20	10	10	13
30	0	0	0	0	0	0	0	30
20	0	A	B	C	90	90	0	20
20	0	D	E	F	90	90	0	20
10	0	G	H	I	90	90	0	10
10	0	90	90	90	90	90	0	10
10	0	90	90	90	90	90	0	10
20	0	0	0	0	0	0	0	20
20	20	10	20	10	20	10	10	13

Review: Mathematical Representation for Smoothing

$$G[3,3] = \frac{1}{9}(A + B + C + D + E + F + G + H + I)$$

$$G[i,j] = \frac{1}{(2k+1)^2} \sum_{u=-k}^{k} \sum_{v=-k}^{k} F[i+u, j+v]$$

Attribute uniform weight to each pixel

Loop over all pixels in neighborhood around image pixel *F[i,j]*

a	b	c
d	e	f
g	h	i

20	20	10	20	10	20	10	10	13
30	0	0	0	0	0	0	0	30
20	0	A	B	C	90	90	0	20
20	0	D	E	F	90	90	0	20
10	0	G	H	I	90	90	0	10
10	0	90	90	90	90	90	0	10
10	0	90	90	90	90	90	0	10
20	0	0	0	0	0	0	0	20
20	20	10	20	10	20	10	10	13

More Generally,

$$G[3,3] = a * A + b * B + c * C + d * D + e * E + f * F + h * H + i * I$$

4

Review: Mathematical Representation for Smoothing

$$G[3,3] = \frac{1}{9}(A + B + C + D + E + F + G + H + I)$$

$$G[i,j] = \frac{1}{(2k+1)^2} \sum_{u=-k}^{k} \sum_{v=-k}^{k} F[i+u, j+v]$$

Attribute uniform weight to each pixel

Loop over all pixels in neighborhood around image pixel *F[i,j]*

a	b	c
d	e	f
g	h	i

20	20	10	20	10	20	10	10	13
30	0	0	0	0	0	0	0	30
20	0	A	B	C	90	90	0	20
20	0	D	E	F	90	90	0	20
10	0	G	H	I	90	90	0	10
10	0	90	90	90	90	90	0	10
10	0	90	90	90	90	90	0	10
20	0	0	0	0	0	0	0	20
20	20	10	20	10	20	10	10	13

More Generally,

$$G[3,3] = a*A + b*B + c*C + d*D + e*E + f*F + h*H + i*I$$

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

Review: Mathematical Representation for Smoothing

$$G[3,3] = \frac{1}{9}(A + B + C + D + E + F + G + H + I)$$

$$G[i,j] = \frac{1}{(2k+1)^2} \sum_{u=-k}^{k} \sum_{v=-k}^{k} F[i+u, j+v]$$

Attribute uniform weight to each pixel

Loop over all pixels in neighborhood around image pixel *F[i,j]*

a	b	c
d	e	f
g	h	i

20	20	10	20	10	20	10	10	13
30	0	0	0	0	0	0	0	30
20	0	A	B	C	90	90	0	20
20	0	D	E	F	90	90	0	20
10	0	G	H	I	90	90	0	10
10	0	90	90	90	90	90	0	10
10	0	90	90	90	90	90	0	10
20	0	0	0	0	0	0	0	20
20	20	10	20	10	20	10	10	13

More Generally,

$$G[3,3] = a*A + b*B + c*C + d*D + e*E + f*F + h*H + i*I$$

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

Attribute non-uniform weights

Cross-Correlation Method

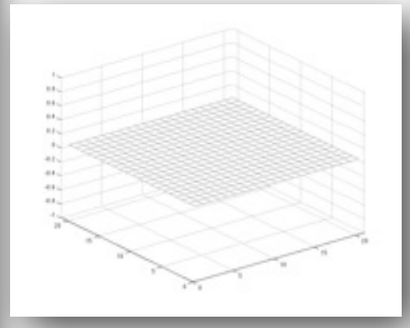
$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} h[u,v]F[i+u, j+v]$$

- ★ Denoted by $G = h \otimes F$
- ★ Filtering an image: replace each pixel with a linear combination of its neighbors.
- ★ The filter "kernel" or "mask" *h[u,v]* is the prescription for the weights in the linear combination.

In signal processing, **cross-correlation** is a measure of similarity of two waveforms as a function of a time-lag applied to one of them. This is also known as a *sliding dot product* or *sliding inner-product*.

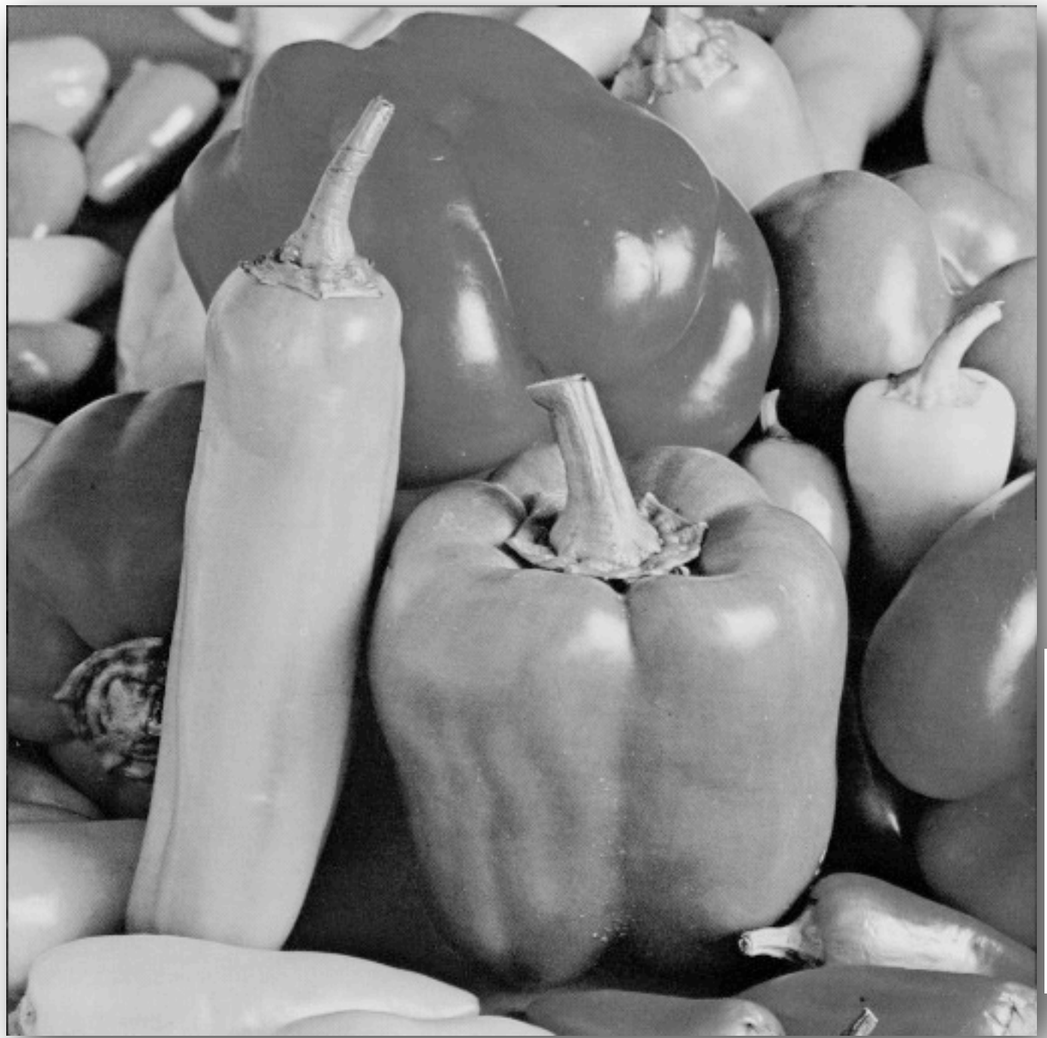
Example: Box Filter

Box/Average
Filter
21x21

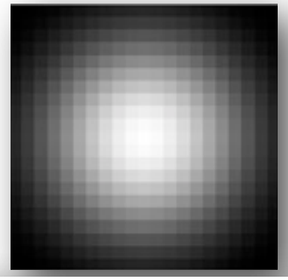


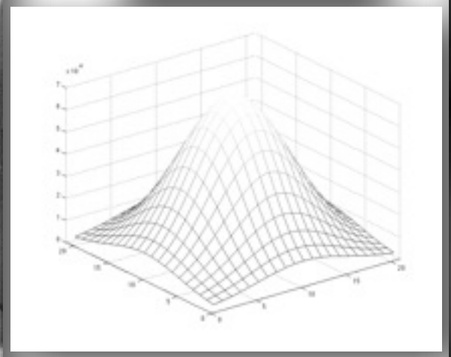
6

Example: Gaussian Filter



Gaussian
Filter
21x21





Filtering by a Kernel (defining Convolution)

0	0	0	0	0	0	0
0	0	0	0	0	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	0	0	0	0	0
0	0	0	0	0	0	0

F[i,j]

$\otimes$

a	b	c
d	e	f
g	h	i

h[i,j]

$=$

G[i,j]

- Impulse function
- 'Filter' means to slide the kernel over the image which results in a reversed response.

Slide Adapted from Aaron Bobick

Filtering by a Kernel (defining Convolution)

0	0	0	0	0	0	0
0	0	0	0	0	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	0	0	0	0	0
0	0	0	0	0	0	0

F[i,j]

$\otimes$

a	b	c
d	e	f
g	h	i

h[i,j]

=

G[i,j]

- ★ Impulse function
- ★ 'Filter' means to slide the kernel over the image which results in a reversed response.

Slide Adapted from Aaron Bobick

Filtering by a Kernel (defining Convolution)

0	0	0	0	0	0	0
0	0	0	0	0	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	0	0	0	0	0
0	0	0	0	0	0	0

F[i,j]

$\otimes$

a	b	c
d	e	f
g	h	i

h[i,j]

=

			0			

G[i,j]

- Impulse function
- 'Filter' means to slide the kernel over the image which results in a reversed response.

Slide Adapted from Aaron Bobick

Filtering by a Kernel (defining Convolution)

0	0	0	0	0	0	0
0	0	0	0	0	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	0	0	0	0	0
0	0	0	0	0	0	0

F[i,j]

$\otimes$

a	b	c
d	e	f
g	h	i

h[i,j]

=

	0	f

G[i,j]

- ★ Impulse function
- ★ ‘Filter’ means to slide the kernel over the image which results in a reversed response.

Slide Adapted from Aaron Bobick

Filtering by a Kernel (defining Convolution)

0	0	0	0	0	0	0
0	0	0	0	0	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	0	0	0	0	0
0	0	0	0	0	0	0

F[i,j]

$\otimes$

a	b	c
d	e	f
g	h	i

h[i,j]

=

0	f	e

G[i,j]

- Impulse function
- 'Filter' means to slide the kernel over the image which results in a reversed response.

Slide Adapted from Aaron Bobick

Filtering by a Kernel (defining Convolution)

0	0	0	0	0	0	0
0	0	0	0	0	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	0	0	0	0	0
0	0	0	0	0	0	0

$F[i,j]$

$\otimes$

a	b	c
d	e	f
g	h	i

$h[i,j]$

$=$

	0	f	e	d

$G[i,j]$

- ★ Impulse function
- ★ ‘Filter’ means to slide the kernel over the image which results in a reversed response.

Slide Adapted from Aaron Bobick

Filtering by a Kernel (defining Convolution)

0	0	0	0	0	0	0
0	0	0	0	0	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	0	0	0	0	0
0	0	0	0	0	0	0

F[i,j]

$\otimes$

a	b	c
d	e	f
g	h	i

h[i,j]

=

0	f	e	d	0

G[i,j]

- ★ Impulse function
- ★ 'Filter' means to slide the kernel over the image which results in a reversed response.

Slide Adapted from Aaron Bobick

Filtering by a Kernel (defining Convolution)

0	0	0	0	0	0	0
0	0	0	0	0	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	0	0	0	0	0
0	0	0	0	0	0	0

F[i,j]

$\otimes$

a	b	c
d	e	f
g	h	i

h[i,j]

$=$

0	i	h	g	0
0	f	e	d	0
0	c	b	a	0

G[i,j]

- ★ Impulse function
- ★ 'Filter' means to slide the kernel over the image which results in a reversed response.

Slide Adapted from Aaron Bobick

Filtering by a Kernel (defining Convolution)

0	0	0	0	0	0	0
0	0	0	0	0	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	0	0	0	0	0
0	0	0	0	0	0	0

$F[i,j]$

$\otimes$

a	b	c
d	e	f
g	h	i

$h[i,j]$

$=$

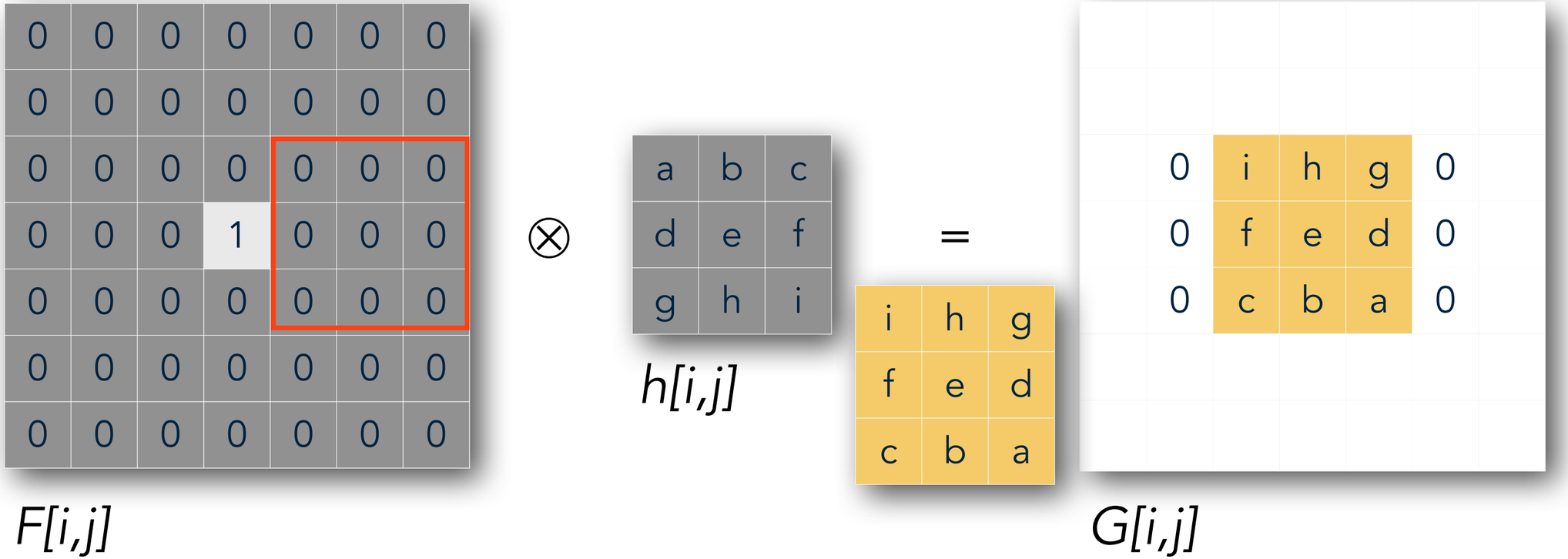
	0	i	h	g	0
	0	f	e	d	0
	0	c	b	a	0

$G[i,j]$

- Impulse function
- 'Filter' means to slide the kernel over the image which results in a reversed response.

Slide Adapted from Aaron Bobick

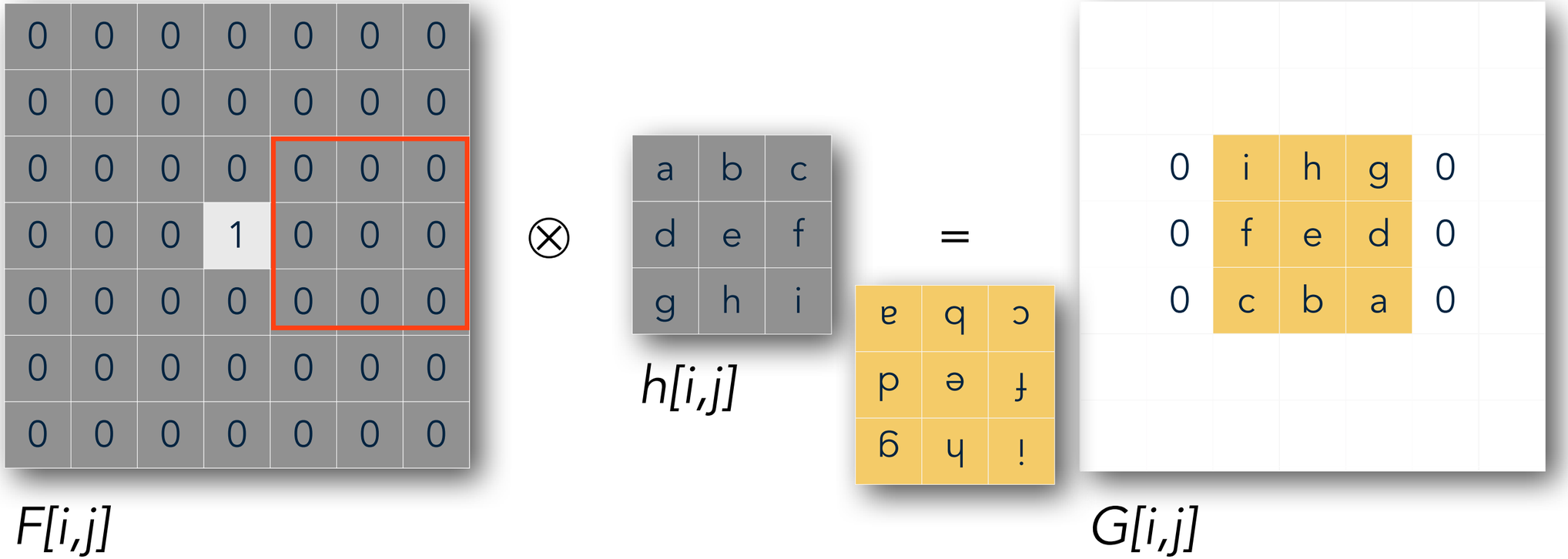
Filtering by a Kernel (defining Convolution)



- ★ Impulse function
- ★ 'Filter' means to slide the kernel over the image which results in a reversed response.

Slide Adapted from Aaron Bobick

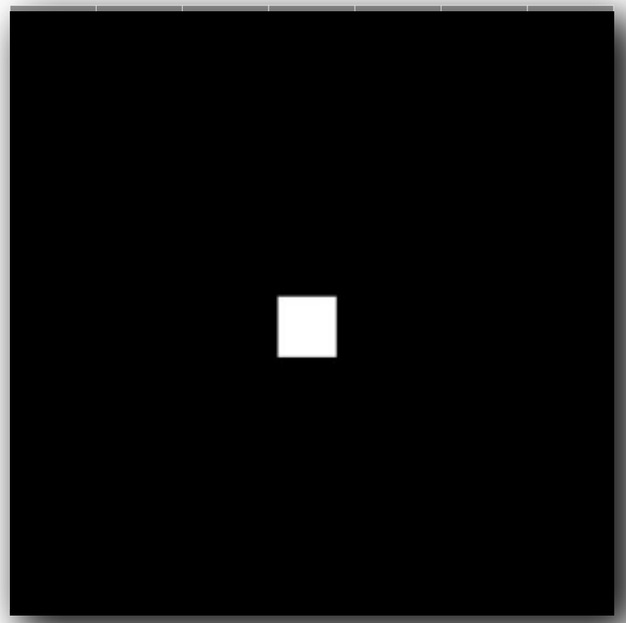
Filtering by a Kernel (defining Convolution)



- ★ Impulse function
- ★ 'Filter' means to slide the kernel over the image which results in a reversed response.

Slide Adapted from Aaron Bobick

Filtering by a Kernel (defining Convolution)



F[i,j]

$\otimes$

a	b	c
d	e	f
g	h	i

h[i,j]

=

ɐ	q	ɔ
p	ə	ɟ
ɓ	ɥ	¡

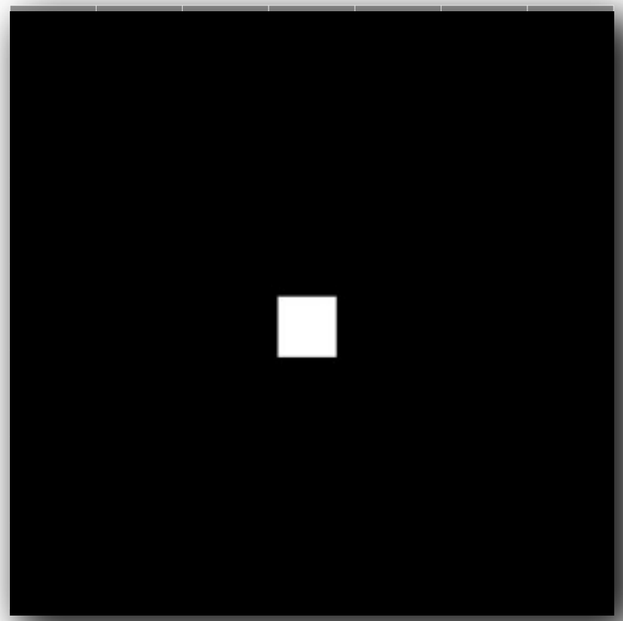
0	i	h	g	0
0	f	e	d	0
0	c	b	a	0

G[i,j]

- Impulse function
- 'Filter' means to slide the kernel over the image which results in a reversed response.

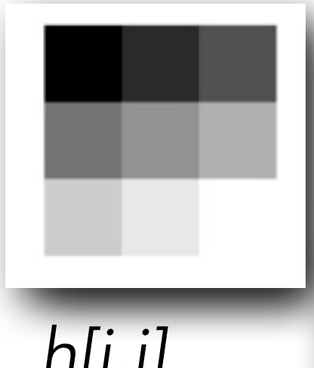
Slide Adapted from Aaron Bobick

Filtering by a Kernel (defining Convolution)



F[i,j]

$\otimes$



h[i,j]

$=$

ɐ	q	ɔ
p	ə	ɟ
ɓ	ɥ	¡

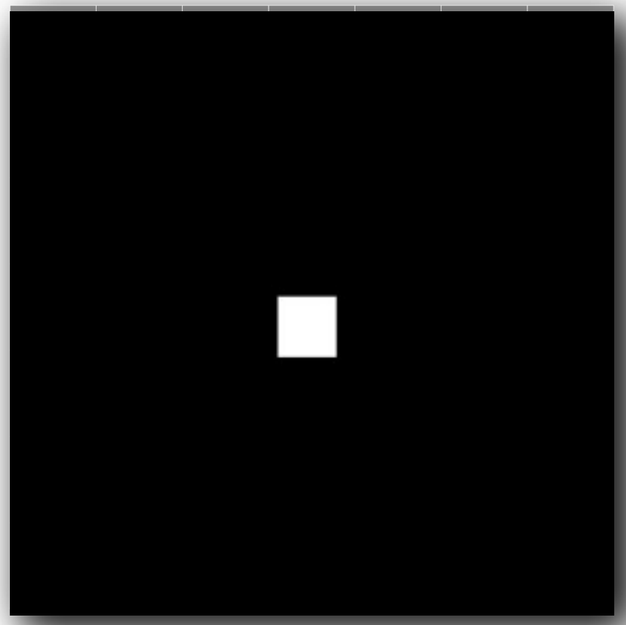
0	i	h	g	0
0	f	e	d	0
0	c	b	a	0

G[i,j]

- Impulse function
- 'Filter' means to slide the kernel over the image which results in a reversed response.

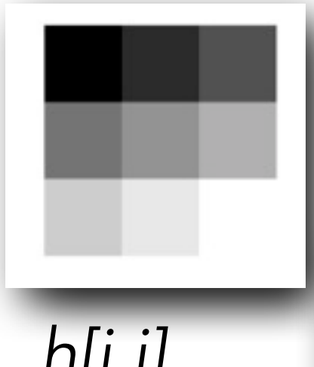
Slide Adapted from Aaron Bobick

Filtering by a Kernel (defining Convolution)



F[i,j]

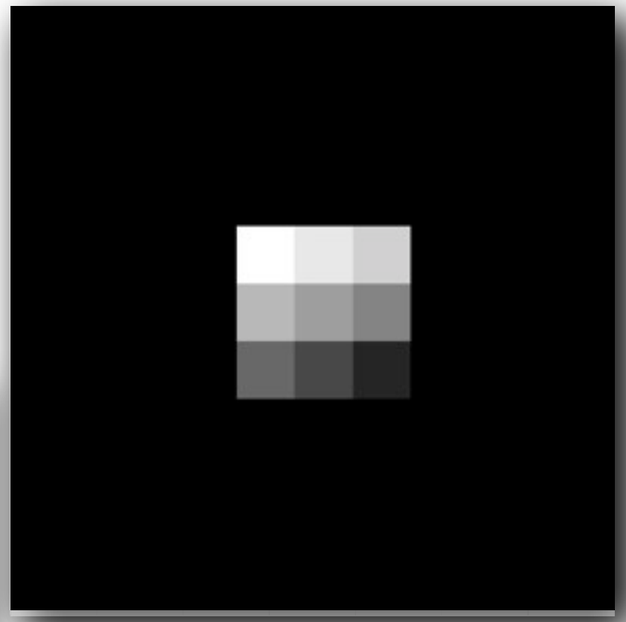
$\otimes$



h[i,j]

=

a	b	c
d	e	f
g	h	i



G[i,j]

- Impulse function
- 'Filter' means to slide the kernel over the image which results in a reversed response.

Slide Adapted from Aaron Bobick

Convolution Method

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

Convolution is a mathematical operation on two functions *F* and *h*, producing a third function that is typically viewed as a modified version of one of the original functions, giving the area overlap between the two functions as a function of the amount that one of the original functions is translated.

- Denoted by $G = h * F$
- Flip the filter in both dimensions (bottom to top, right to left)
- Then apply cross-correlation

Convolution vs. Cross-Correlation Methods

- For a both a Gaussian and a box filter, how will the outputs differ for each of the methods?
- If the input is an impulse signal, how will the outputs differ for each of the methods?

10

Convolution vs. Cross-Correlation Methods

Cross-Correlation: $G = h \otimes F$ $\quad G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} h[u,v] F[i+u, j+v]$

- ★ For a both a Gaussian and a box filter, how will the outputs differ for each of the methods?
- ★ If the input is an impulse signal, how will the outputs differ for each of the methods?

10

Convolution vs. Cross-Correlation Methods

Cross-Correlation: $G = h \otimes F$ $\quad G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} h[u,v] F[i+u, j+v]$

Convolution: $G = h * F$ $\quad G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} h[u,v] F[i-u, j-v]$

- For a both a Gaussian and a box filter, how will the outputs differ for each of the methods?
- If the input is an impulse signal, how will the outputs differ for each of the methods?

Properties of Convolution

Properties of Convolution

- ★ Linear and Shift Invariants: Behaves the same everywhere (i.e., the value of the output depends on the pattern in the image neighborhood, not the position of the neighborhood).

Properties of Convolution

- ★ Linear and Shift Invariants: Behaves the same everywhere (i.e., the value of the output depends on the pattern in the image neighborhood, not the position of the neighborhood).
- ★ Commutative: $F * G = G * F$

11

Properties of Convolution

- ★ Linear and Shift Invariants: Behaves the same everywhere (i.e., the value of the output depends on the pattern in the image neighborhood, not the position of the neighborhood).
- ★ Commutative: $F * G = G * F$
- ★ Associative: $(F * G) * H = F * (G * H)$

11

Properties of Convolution

- ★ Linear and Shift Invariants: Behaves the same everywhere (i.e., the value of the output depends on the pattern in the image neighborhood, not the position of the neighborhood).
- ★ Commutative: $F * G = G * F$
- ★ Associative: $(F * G) * H = F * (G * H)$
- ★ Identity: Unit Impulse $E = [....0,0,1,0,0....]$, $F * E = F$
  - True of Cross-Correlation?

11

Properties of Convolution

- ★ Linear and Shift Invariants: Behaves the same everywhere (i.e., the value of the output depends on the pattern in the image neighborhood, not the position of the neighborhood).
- ★ Commutative: $F * G = G * F$
- ★ Associative: $(F * G) * H = F * (G * H)$
- ★ Identity: Unit Impulse $E = [....0,0,1,0,0....]$, $F * E = F$
  - True of Cross-Correlation?
- ★ Separable: If the filter is separable, convolve all rows, then convolve all columns.

11

Linear Filters

Linear Filters

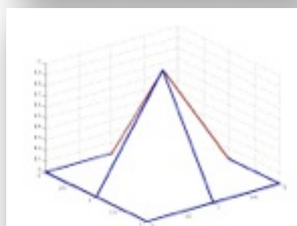
original, 64x64

12

Linear Filters

original, 64x64

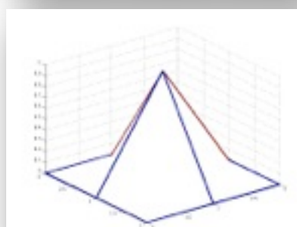
0	0	0
0	1	0
0	0	0



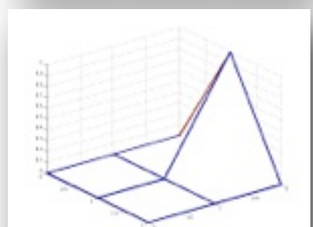
Linear Filters

original, 64x64

0	0	0
0	1	0
0	0	0



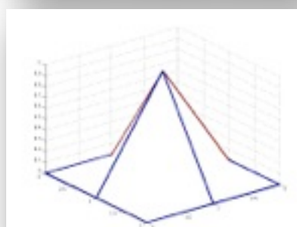
0	0	0
0	0	1
0	0	0



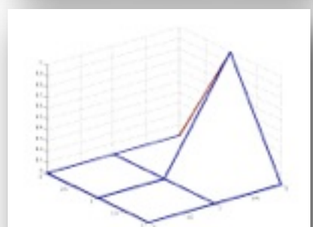
Linear Filters

original, 64x64

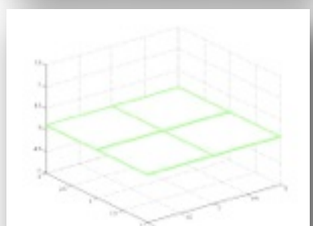
0	0	0
0	1	0
0	0	0



0	0	0
0	0	1
0	0	0



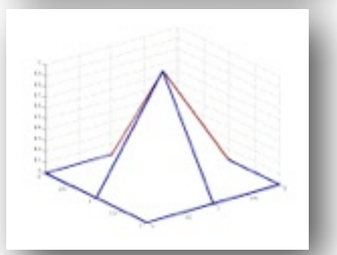
1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9



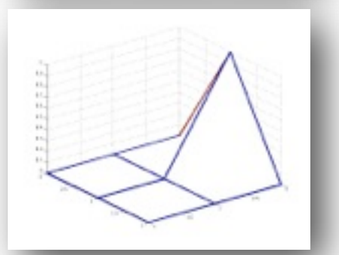
Linear Filters

original, 64x64

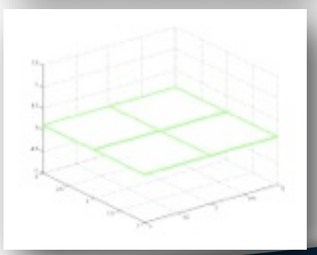
0	0	0
0	1	0
0	0	0



0	0	0
0	0	1
0	0	0



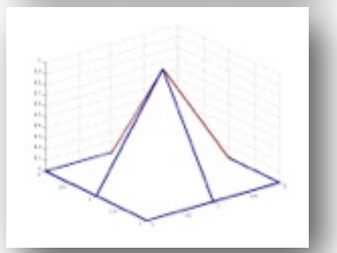
1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9



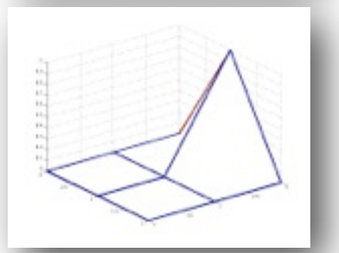
Linear Filters

original, 64x64

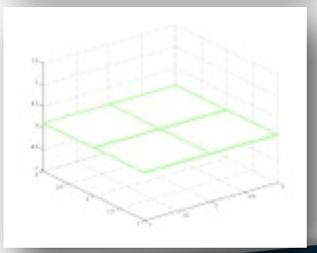
0	0	0
0	1	0
0	0	0



0	0	0
0	0	1
0	0	0

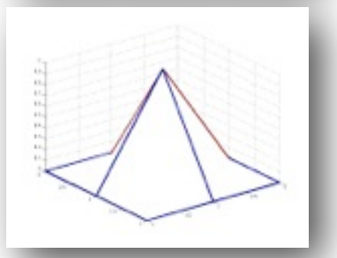


1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

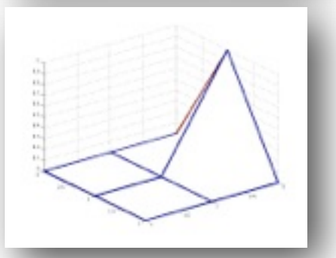


Linear Filters

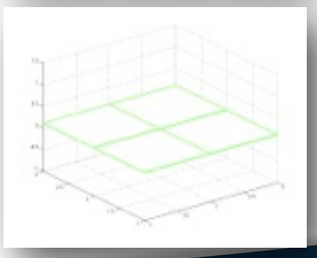
0	0	0
0	1	0
0	0	0



0	0	0
0	0	1
0	0	0



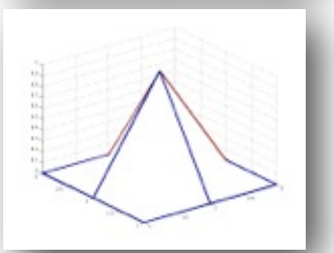
1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9



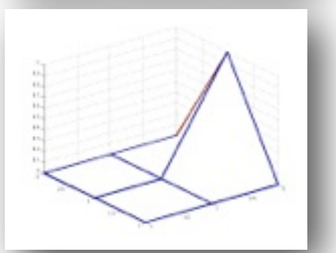
original, 64x64

Linear Filters

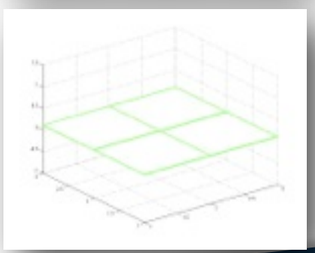
0	0	0
0	1	0
0	0	0



0	0	0
0	0	1
0	0	0



1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9



original, 64x64

Linear Filters

Linear Filters

original, 64x64

Linear Filters

0	0	0
0	1	0
0	0	0

X2 -

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

=

original, 64x64

Linear Filters

0	0	0
0	1	0
0	0	0

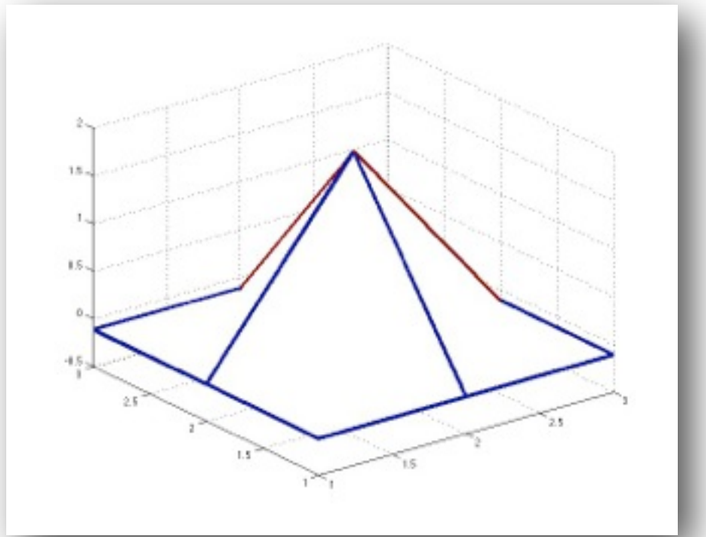
X2 -

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

=

original, 64x64

-0.11	-0.11	-0.11
-0.11	1.9	-0.11
-0.11	-0.11	-0.11



Linear Filters

0	0	0
0	1	0
0	0	0

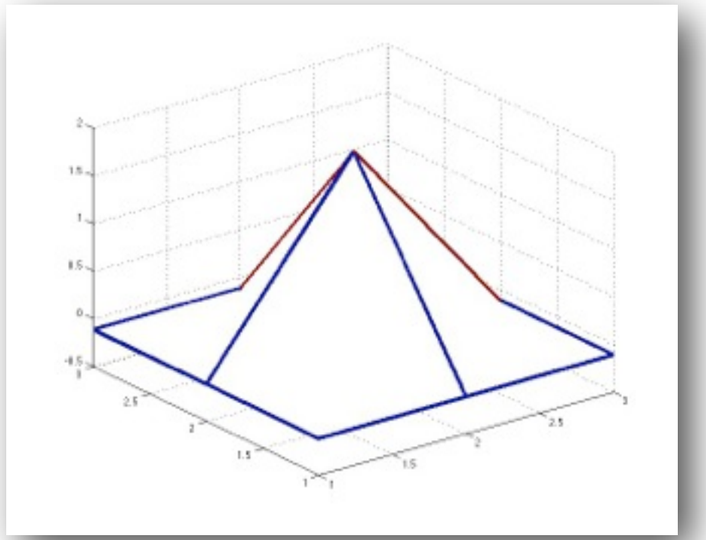
X2 -

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

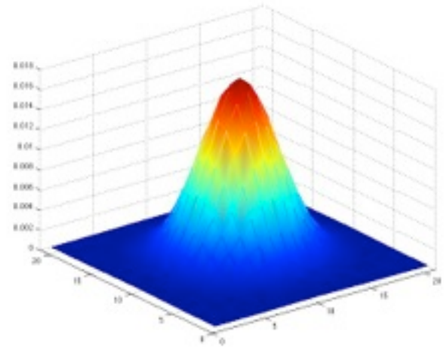
=

original, 64x64

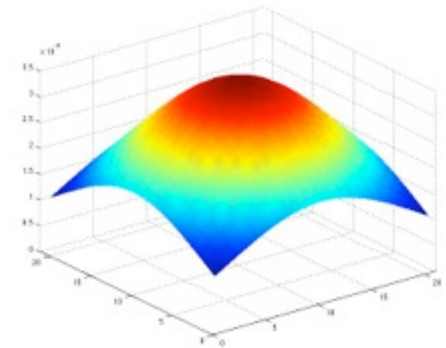
-0.11	-0.11	-0.11
-0.11	1.9	-0.11
-0.11	-0.11	-0.11



Using Gaussian Filters?



21X21, σ=3



21X21, σ=9

- ★ Square kernels are NOT smooth
- ★ Smoothing with an average actually doesn't compare well with a defocussed lens
- ★ Most obvious difference: A single point of light viewed in a defocussed lens looks like a fuzzy blob; the averaging process is square.
- ★ Gaussian function in 2D, with σ is the Variance of the Gaussian

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

Slide Adapted from Aaron Bobick and David Forsyth

14

Using Gaussian Filters for Smoothing

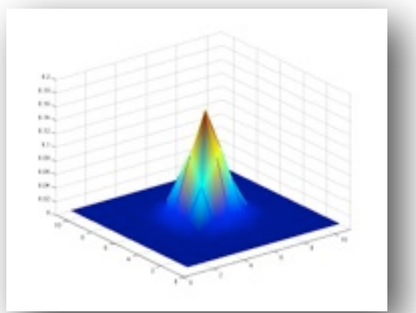
Original, 256x256

σ determines extent of smoothing

Using Gaussian Filters for Smoothing

Original, 256x256

Kernel = 11x11



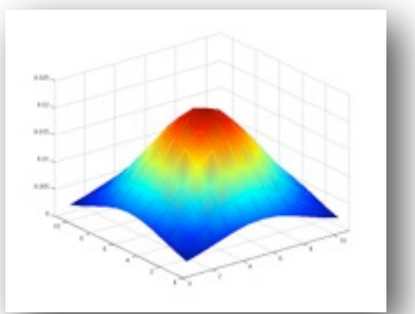
$\sigma=1$

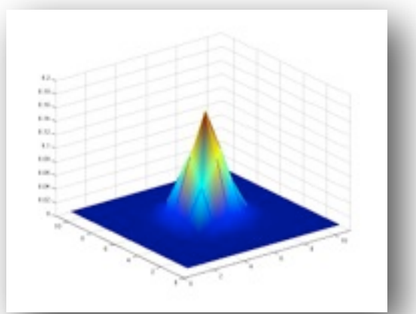
σ determines extent of smoothing

Using Gaussian Filters for Smoothing

Original, 256x256

Kernel = 11x11





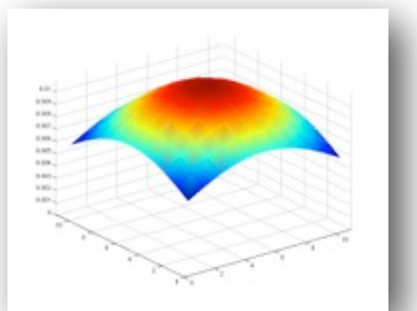
σ=3 σ=1

σ determines extent of smoothing

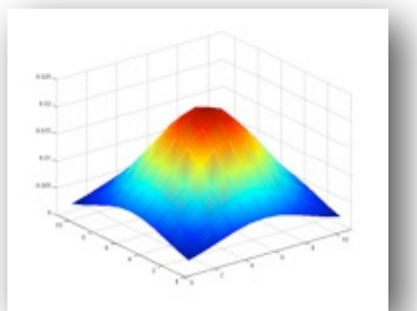
Using Gaussian Filters for Smoothing

Original, 256x256

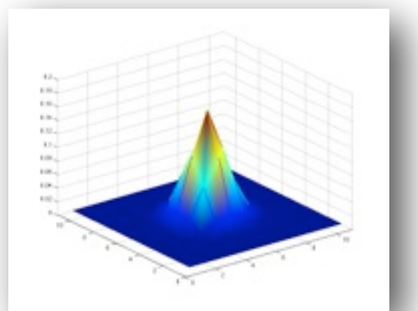
Kernel = 11x11



σ=6



σ=3



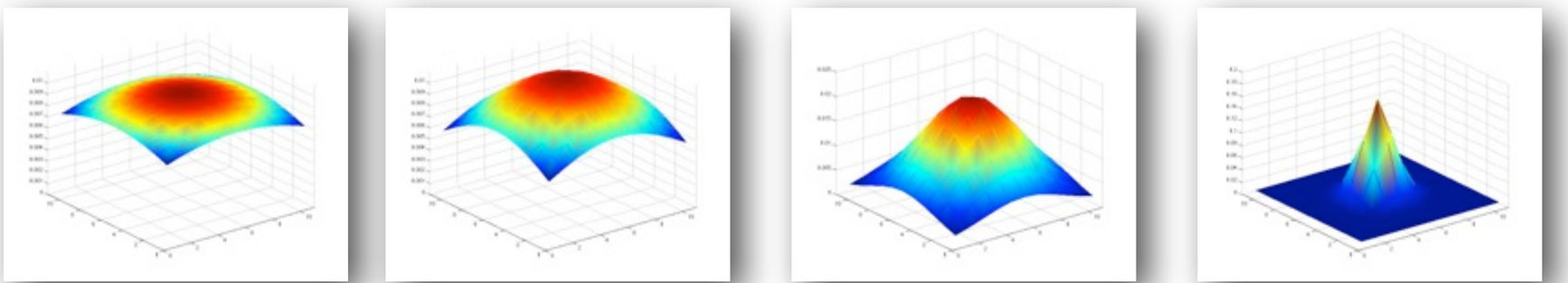
σ=1

σ determines extent of smoothing

Using Gaussian Filters for Smoothing

Original, 256x256

Kernel = 11x11



σ=9 σ=6 σ=3 σ=1

σ determines extent of smoothing

Summary

- ★ Presented Cross-Correlation method for filtering images.
- ★ Presented Convolution method for filtering images.
- ★ Discussed differences between the Cross-Correlation and Convolution methods for filtering images.
- ★ Discussed Properties of the Convolution method for filtering images.

Next Class

- ★ Image Analysis: Edge Detection

17

Credits

- ★ Matlab™ software by Mathworks Inc.
- ★ Some Slides adapted from Aaron Bobick, Steve Seitz, Steve Marschner.
- ★ Images used from USC's Signal and Image Processing Institute's Image Database

Georgia Institute of Technology

Computational Photography

Dr. Irfan Essa
Professor
School of Interactive Computing

Study the basics of computation and its impact on the entire workflow of photography, from capturing, manipulating and collaborating on, and sharing photographs.