CPSC 4040/6040 Computer Graphics Images

Joshua Levine
levinej@clemson.edu

Lecture 12
Region Processing

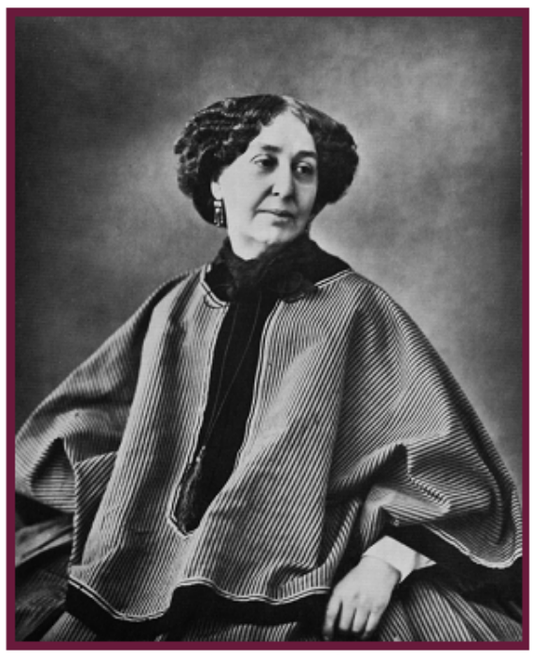
Sept. 29, 2015

Slide Credits:
Ross T. Whitaker
Frédo Durand

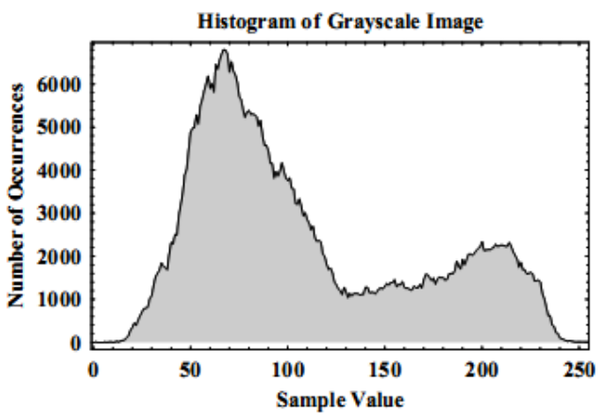
Agenda

- PA02 Results
- Q02 Due on Thurs.
- PA03 Questions?

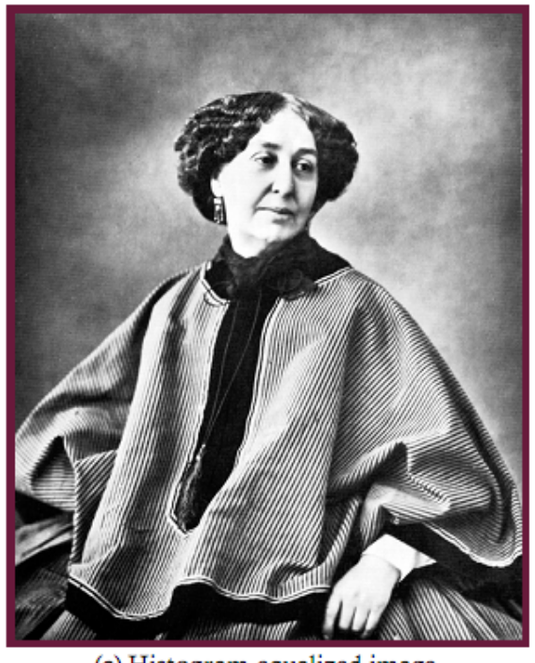
Last Time



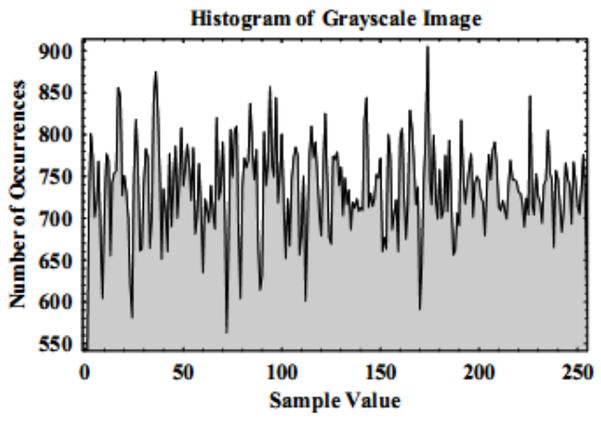
(a) Underexposed image.



(b) Histogram of the underexposed image.

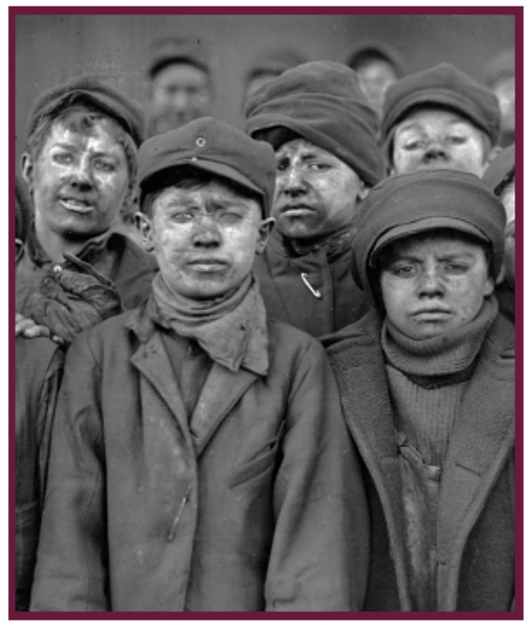


(c) Histogram-equalized image.

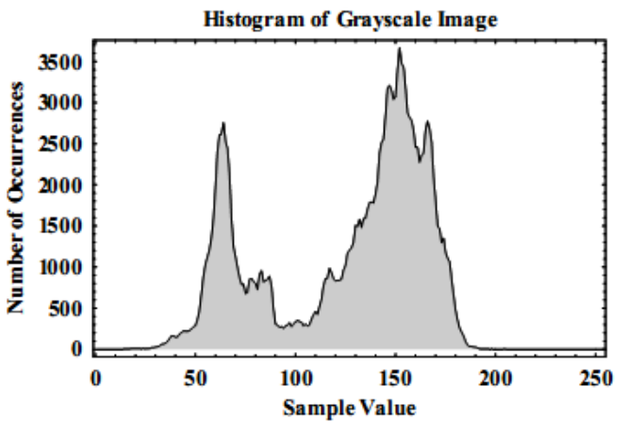


(d) Histogram of the equalized image.

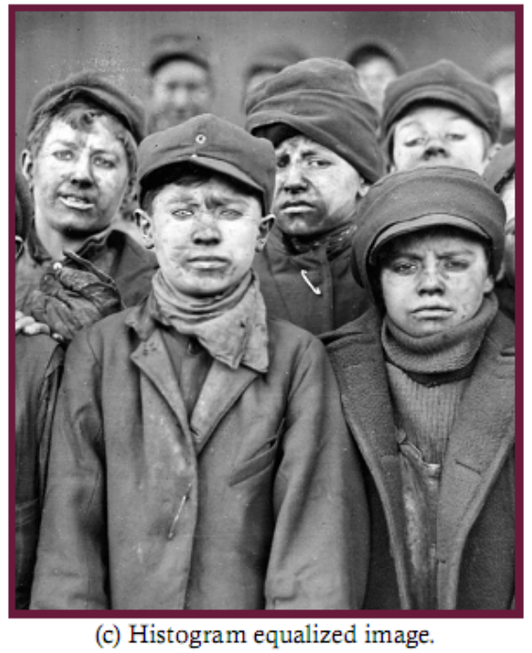
Figure 5.10. Histogram equalization of an underexposed image.



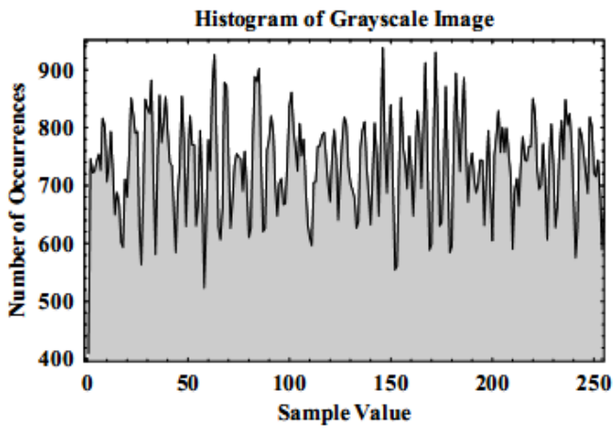
(a) Low-contrast image.



(b) Histogram of the low-contrast image.



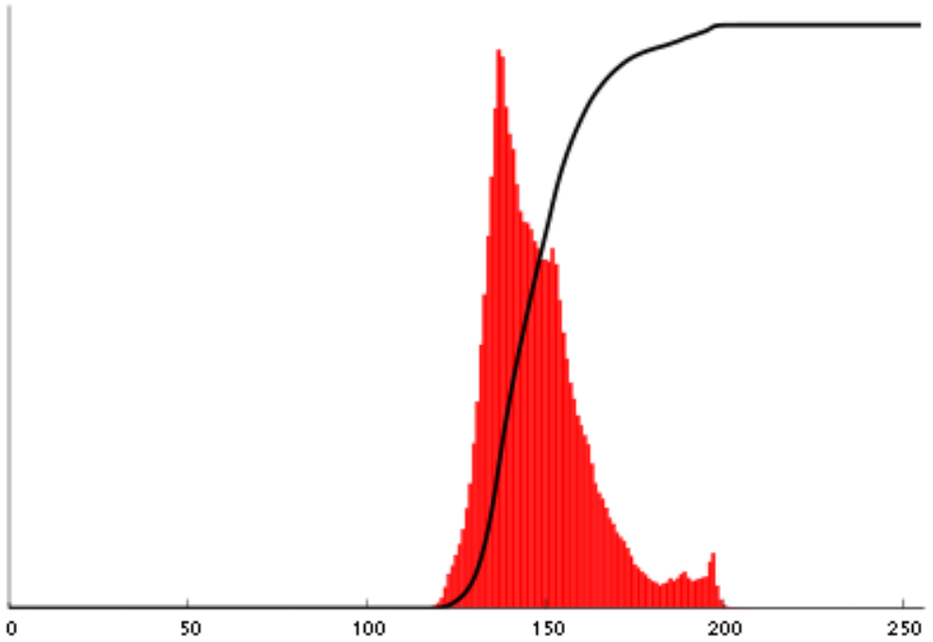
(c) Histogram equalized image.

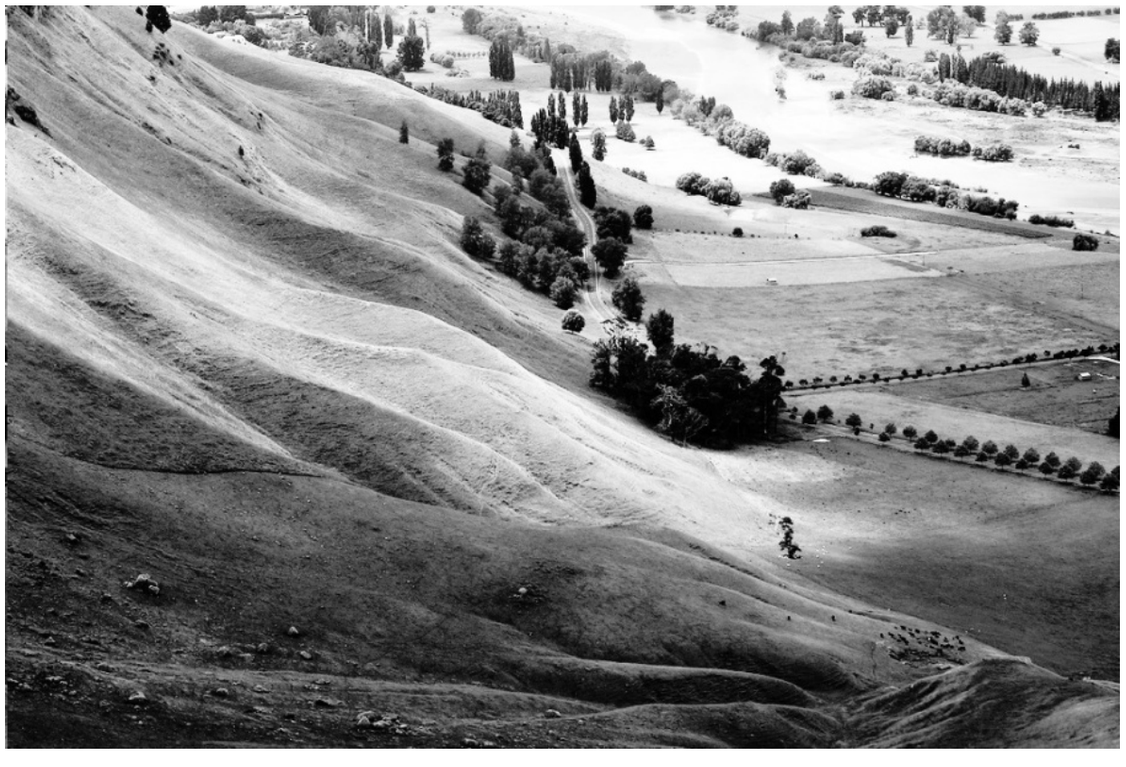


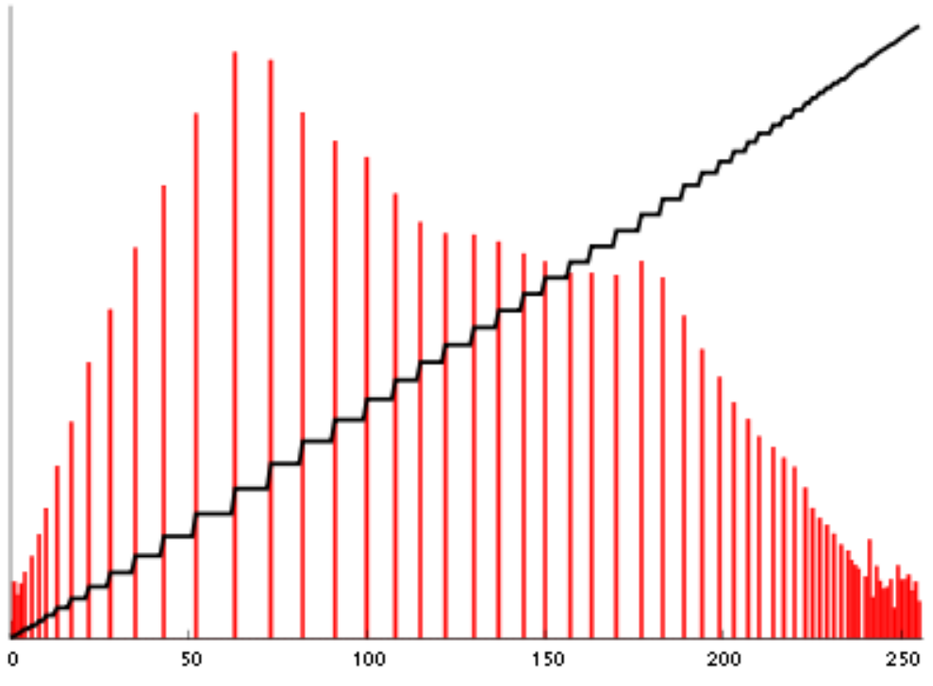
(d) Histogram of the equalized image.

Figure 5.11. Histogram equalization of a low-contrast image.

The Black Line is the CDF

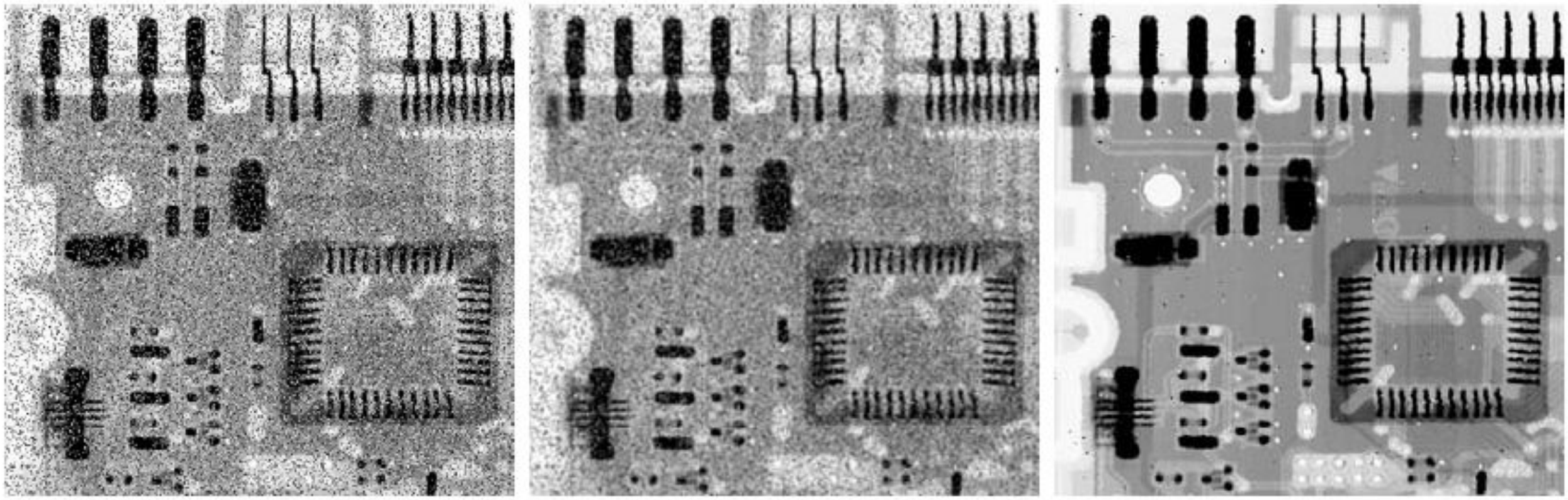






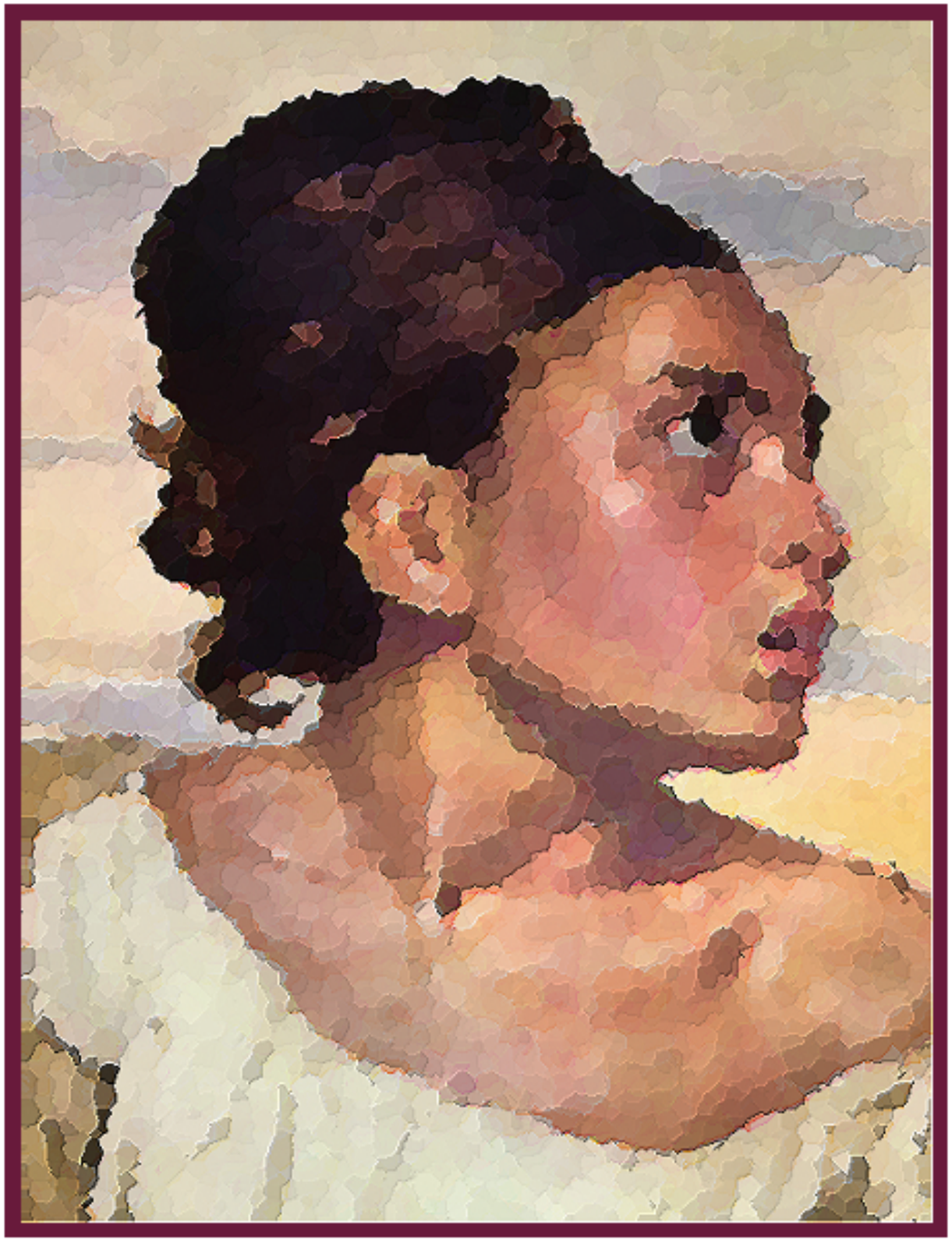
Median vs. Mean Filtering

Output = mean(N_i)



a b c

FIGURE 3.35 (a) X-ray image of circuit board corrupted by salt-and-pepper noise. (b) Noise reduction with a 3×3 averaging mask. (c) Noise reduction with a 3×3 median filter. (Original image courtesy of Mr. Joseph E. Pascente, Lixi, Inc.)



(a) Most common sample.

(b) Range filtering.

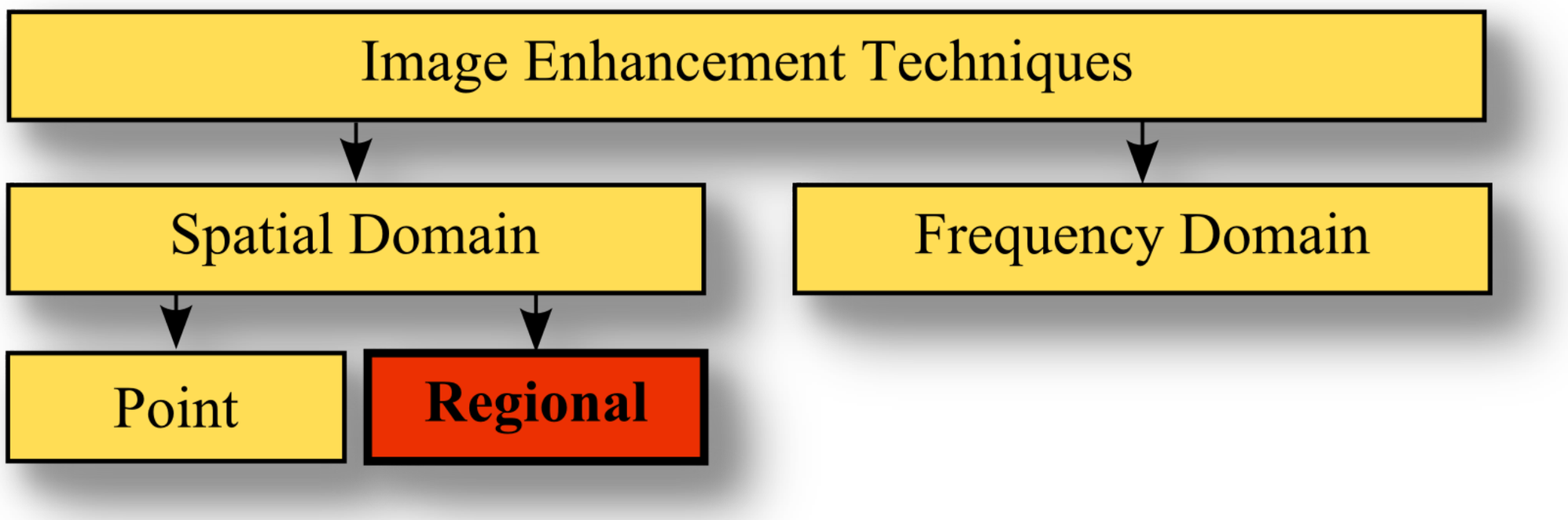
Figure 6.26. Other statistically-based filters.

Convolution

Overview of Region Processing

- Under point processing a single input sample is processed to produce a single output sample.
- In regional processing the value of the output sample is dependent on the values of samples within close proximity to the input sample.
- Exactly how the region is selected and how samples within the region affect the output depends upon the desired effect.

- **Convolution** and **Correlation** are two important regional operations



An Example: Mean Filtering

- Mean filters sum all of the pixels in a local neighborhood N_i and divide by the total number, computing the average pixel.

- Said another way, we replace each pixel as a linear combination of its neighbors (with equal weights!)

$$f(N_i) = 1 / |N_i| \, \Sigma C_j, \text{ for pixel } j \text{ in } N_i$$

- Where the N_i is a square, we call these **box** filters

Box Filtering

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

$$1/9 * \begin{matrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{matrix}$$

Box Filtering

1/9 *

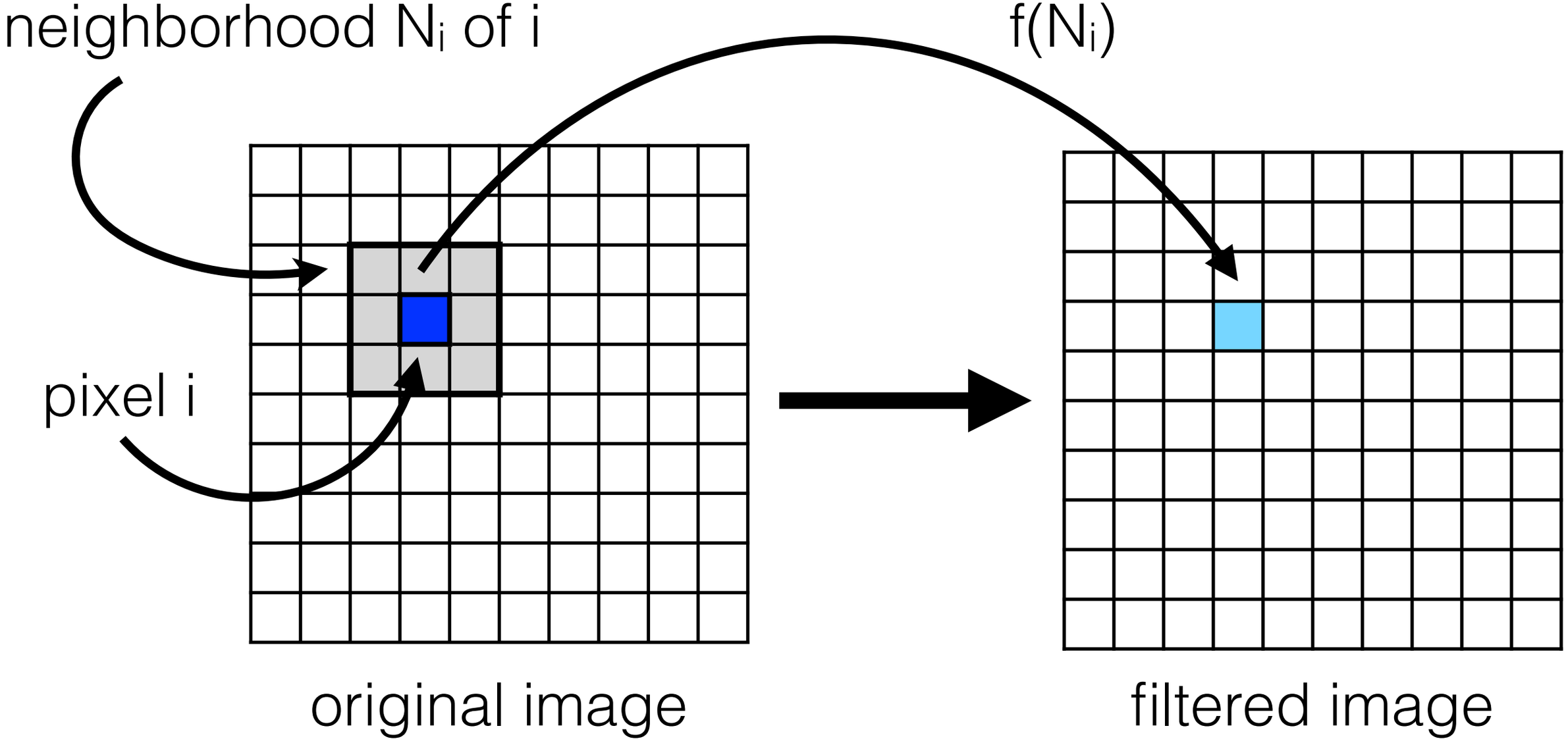
1	1	1
1	1	1
1	1	1

1/25 *

1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1

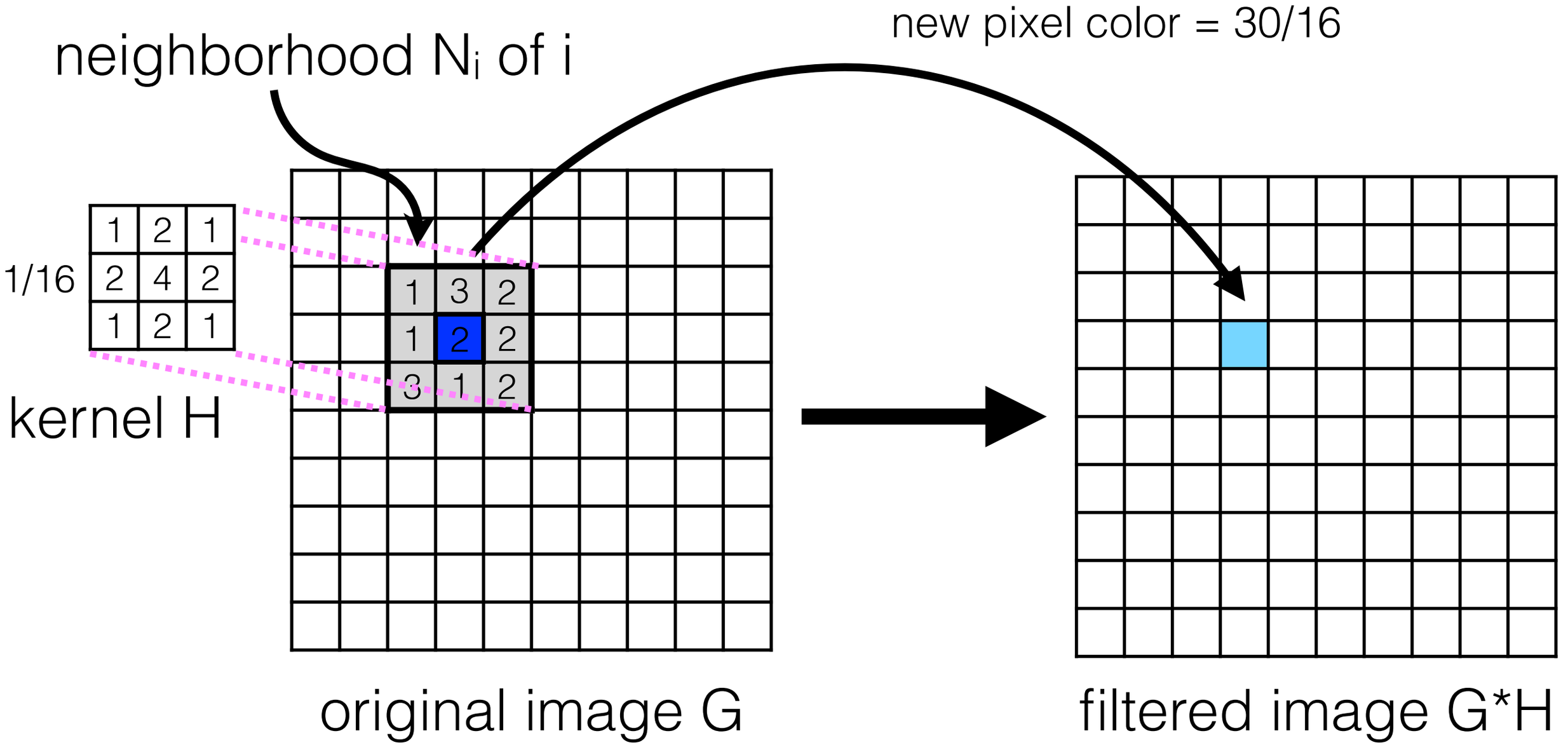
Recall: Filtering (Schematic)

$$C_{out} = f(N_{in})$$



Convolution

- This process of adding up pixels multiplied by various weights is called **convolution**



Kernels

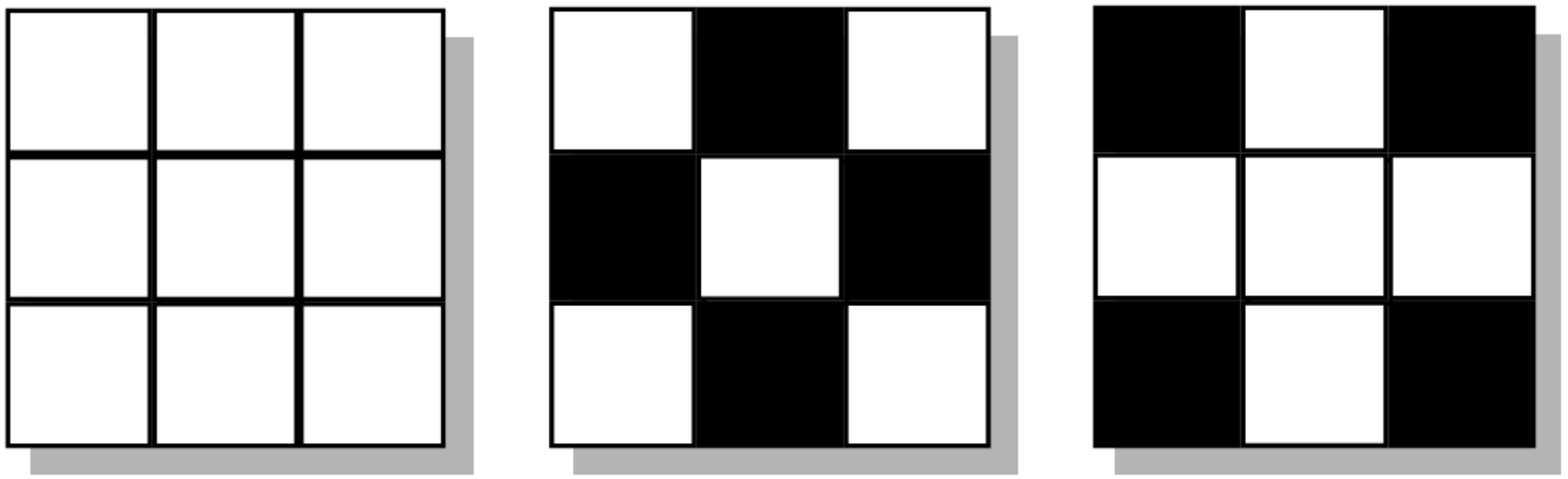
- Convolution employs a rectangular grid of coefficients, known as a **kernel**

- Kernels are like a neighborhood mask, they specify which elements of the image are in the neighborhood and their relative weights.

- A kernel is a set of weights that is applied to corresponding input samples that are summed to produce the output sample.

Recall: Rank Filtering w/ Masks

- Rank filtering can also use non-rectangular regions:
  - The most common of such regions are in the shape of either a plus (+) or a cross (x).
- A **mask** is used to specify non-rectangular regions where certain elements in the region are marked as either included (white) or excluded (black) from the region.



(a) Square. (b) An x mask. (c) A + mask.

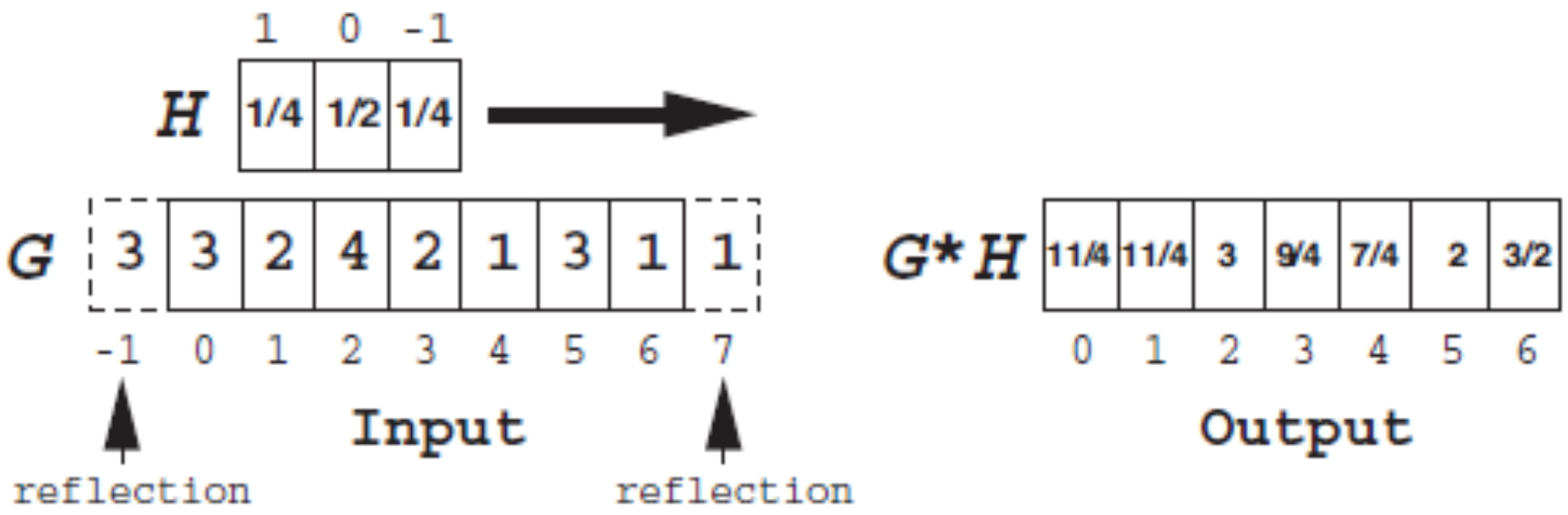
Figure 6.23. Rank filtering masks for a 3×3 square, 3×3 x, and 3×3 +.

One-dimensional Convolution

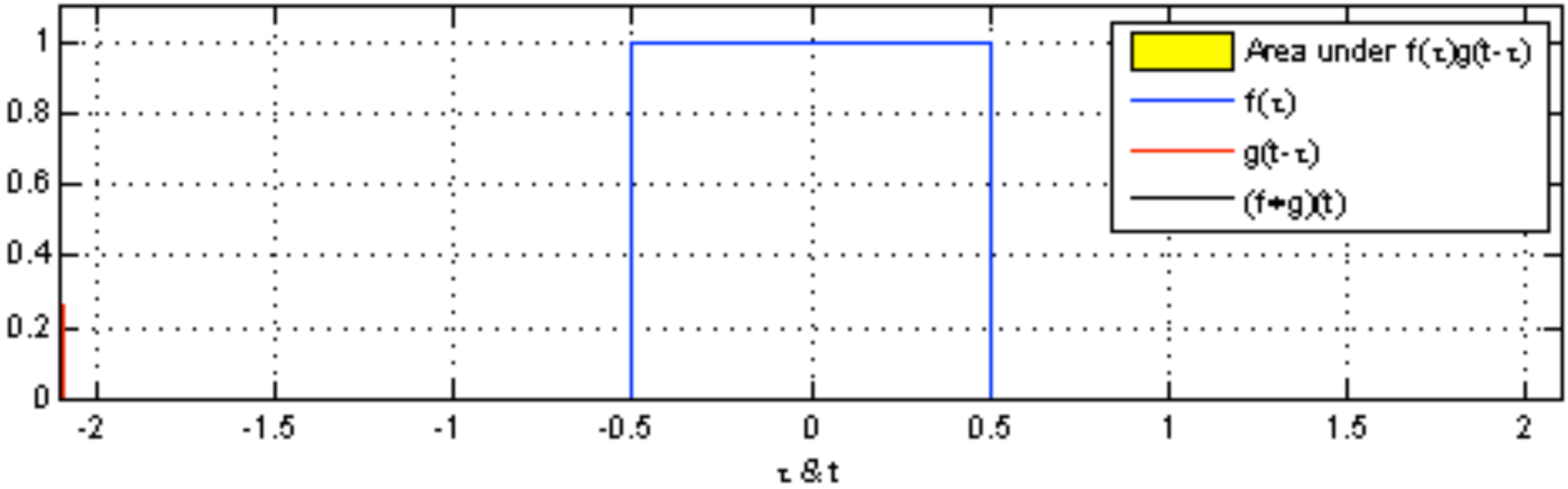
- Can be expressed by the following equation, which takes a filter H and convolves it with G:

$$\hat{G}[i] = (G * H)[i] = \sum_{j=i-n}^{i+n} G[j]H[i-j],\ i \in [0, N-1]$$

- Equivalent to sliding a window



Box Filters (Animated)



- The above is continuous, but a discrete version could be imagined as:

H = 111

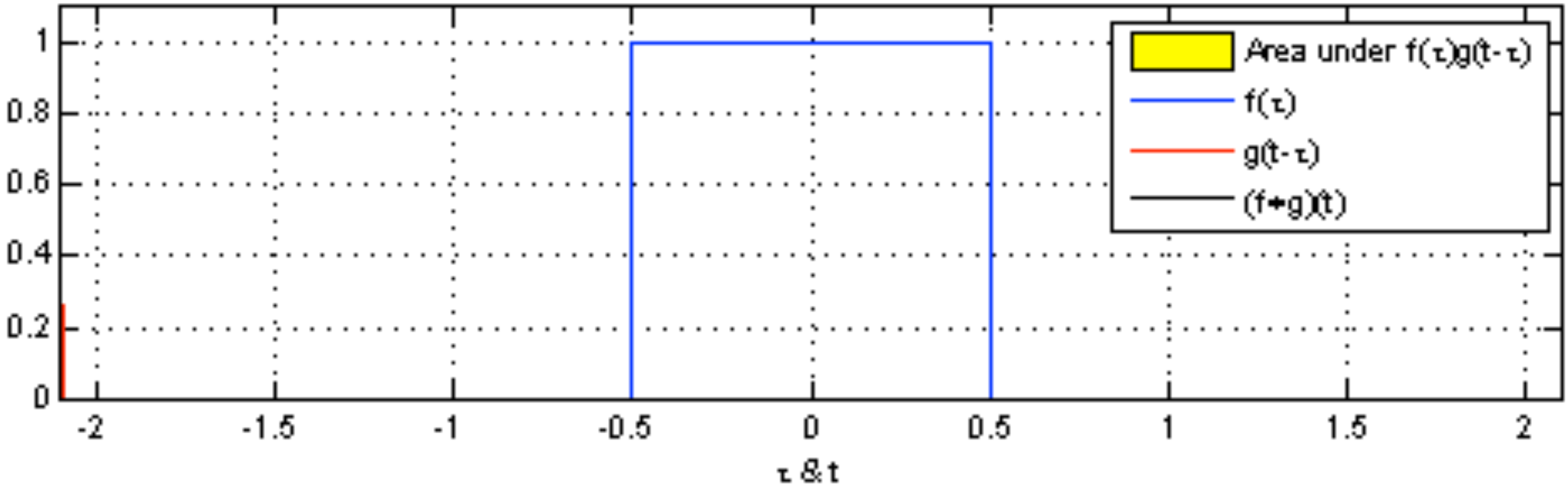
G = ...0000011100000...

G*H = ...0000012100000...

http://en.wikipedia.org/wiki/Convolution

More examples: http://math.mit.edu/daimp/ConvFlipDrag.html

Box Filters (Animated)



- The above is continuous, but a discrete version could be imagined as:

H = 111

G = ...0000011100000...

G*H = ...0000012100000...

http://en.wikipedia.org/wiki/Convolution

More examples: http://math.mit.edu/daimp/ConvFlipDrag.html

2-Dimensional Version

- Given a WxH grayscale image I and an MxN kernel K such that M and N are odd the convolution of I with K is given below where x is in [0,W-1] and y is in [0, H-1].

$$(I \otimes K)(x,y) = I'(x,y) = \sum_{j=-\lfloor M/2 \rfloor}^{\lfloor M/2 \rfloor} \sum_{k=-\lfloor N/2 \rfloor}^{\lfloor N/2 \rfloor} K(j,k) \cdot I(x-j, y-k). \quad (6.1)$$

- The (x-j) and (y-k) terms can be understood as reflections of the kernel about the central vertical and horizontal axes.

- The kernel weights are multiplied by the corresponding image samples and then summed together.

A note on indexing

- Convolution **reflects** the filter to preserve orientation.
- Correlation does **not** have this reflection.
  - But we often use them interchangeably since most kernels are symmetric.

Convolution **reflects** and **shifts** the kernel

Given kernel H =

1	2	3
4	5	6
7	8	9

9	8	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	5	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2	3	0
3	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	5	6	0
0	0	0	0	0	0	0	0	0	0	0	0	1	2	3	0	0	0	7	8	9	0
0	0	0	0	1	0	0	0	0	0	0	0	4	5	6	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	7	8	9	0	0	0				
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				

G⊗H

An Illustration

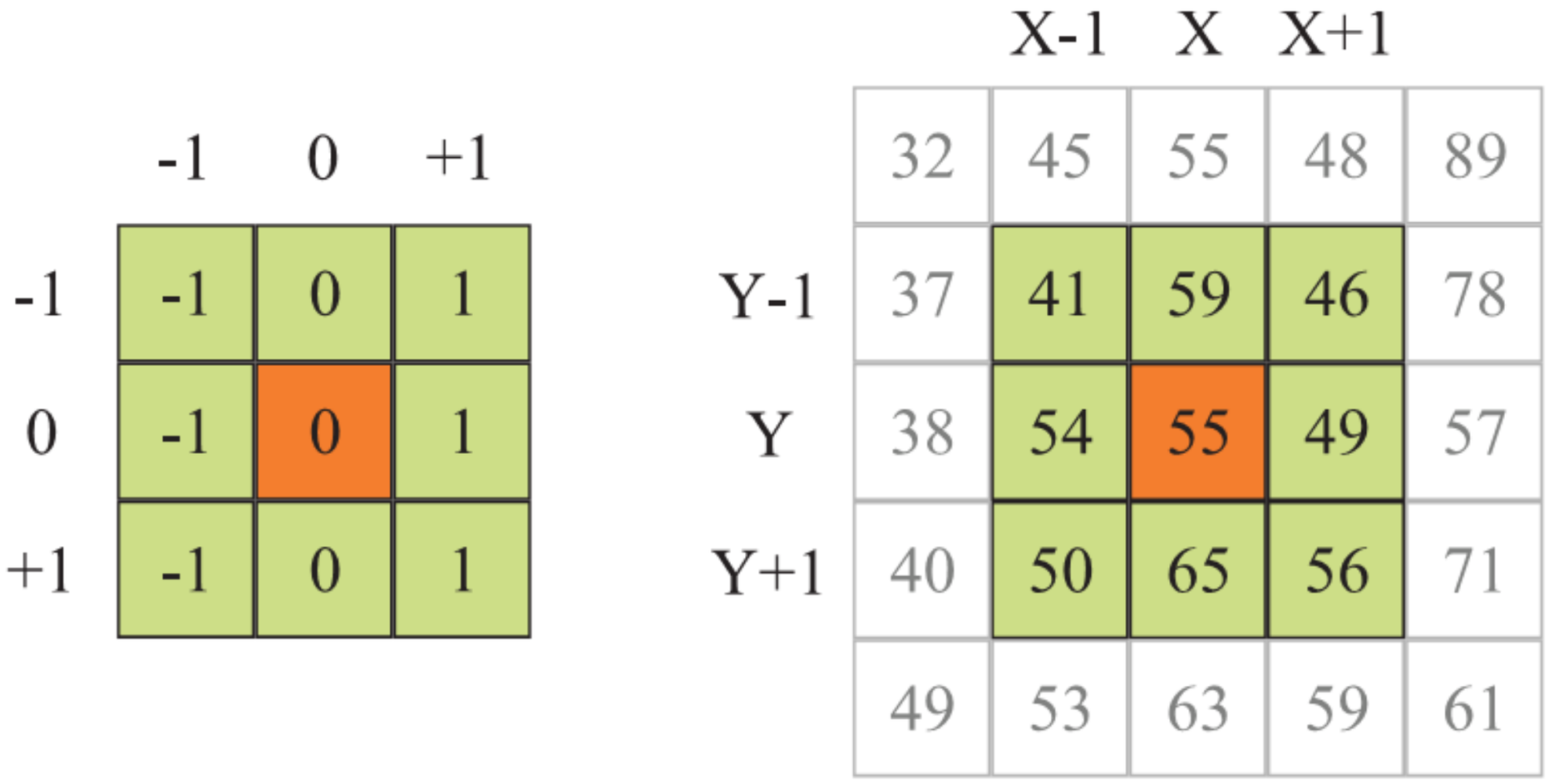


Figure 6.2. A 3×3 kernel is centered over sample $I(x, y)$.

Illustration

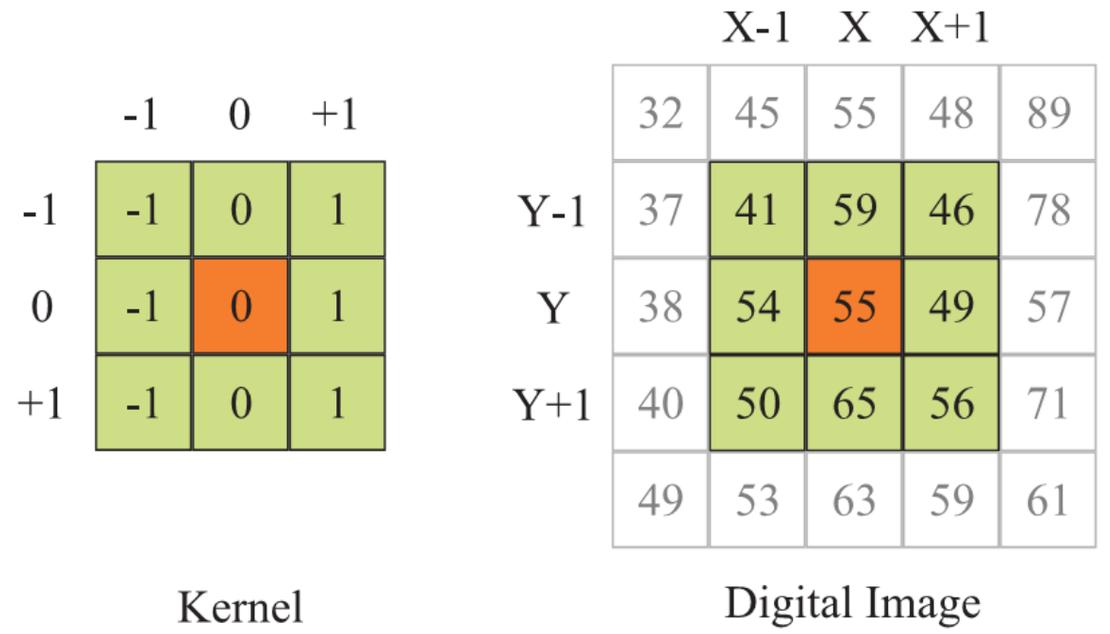


Figure 6.2. A 3×3 kernel is centered over sample $I(x, y)$.

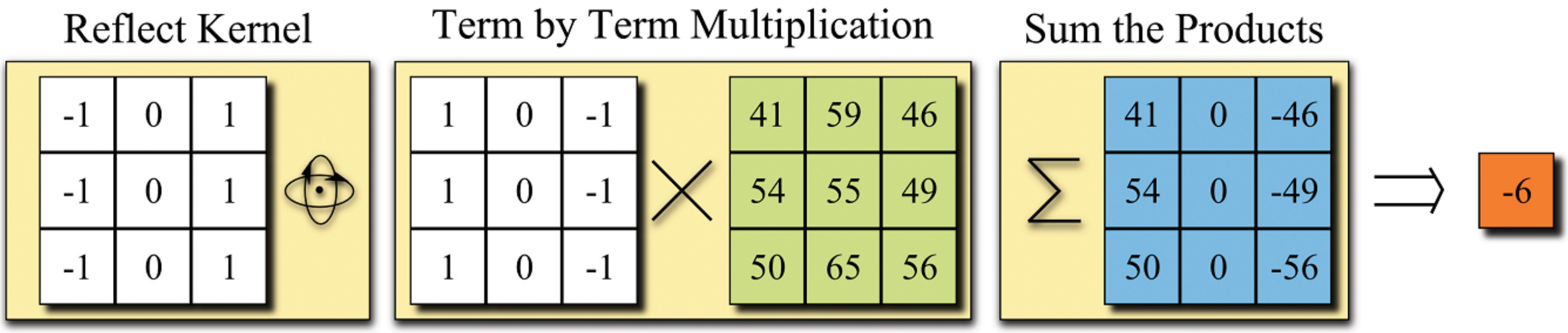
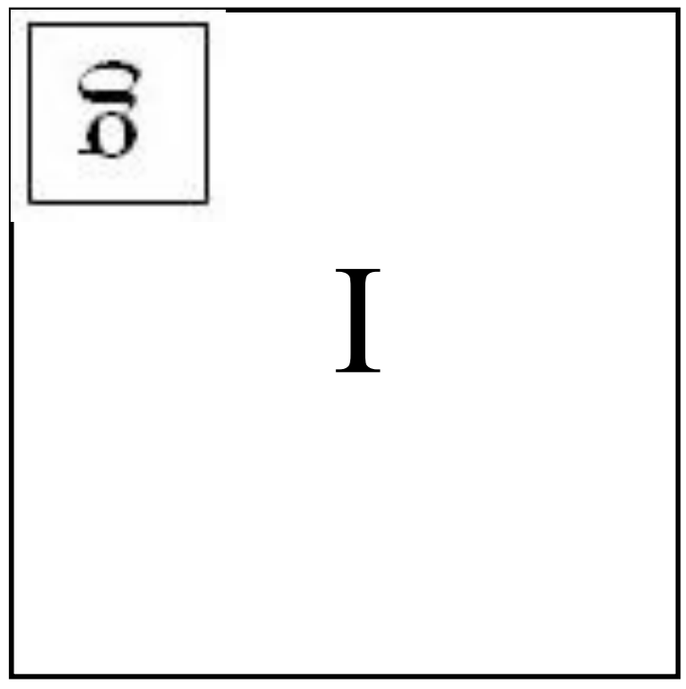


Figure 6.3. Convolution steps.

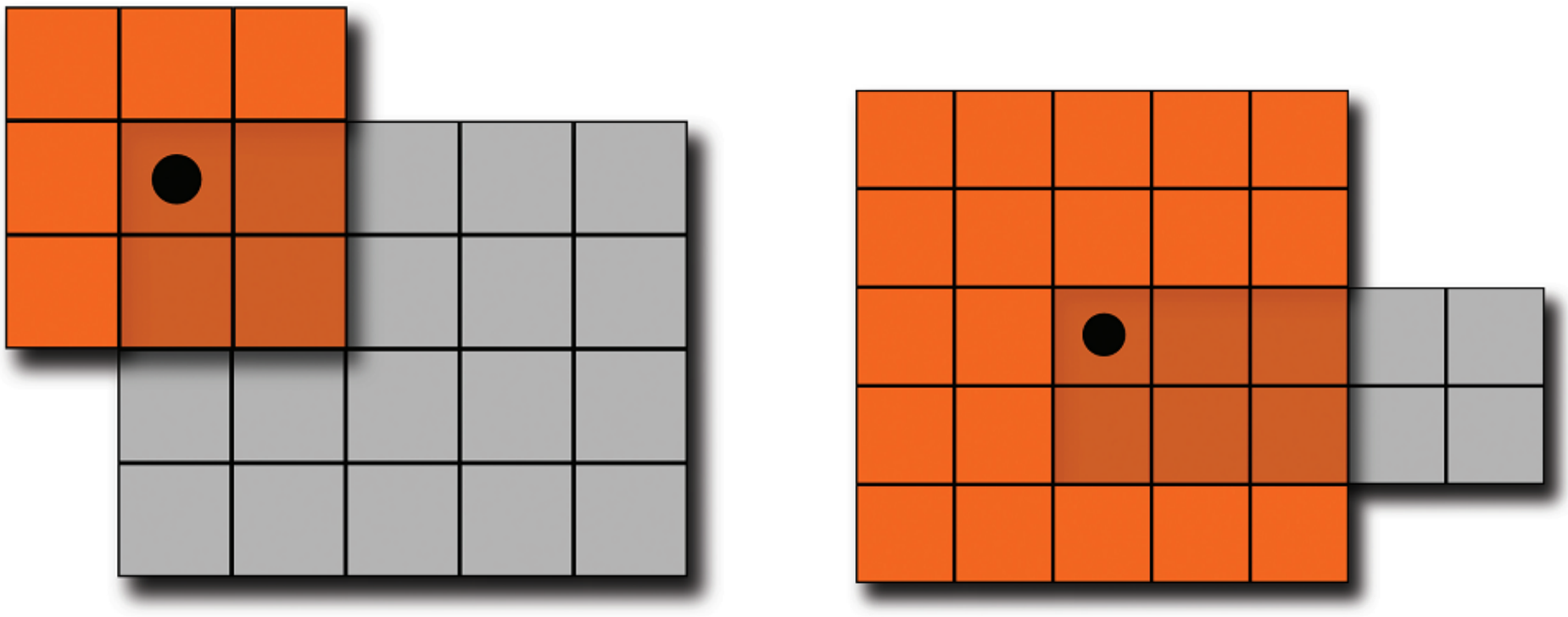
More Formally

$$I \otimes g(x) = \int_{x'} I(x')g(x - x')$$



Reminder: Boundaries

- Consider how to handle boundaries. What should be done if the kernel falls off of the boundary of the source image as shown in the illustrations below?



(a) Kernel at $I(0,0)$. (b) Kernel larger than the source.

Figure 6.4. Illustration of the edge handling problem.

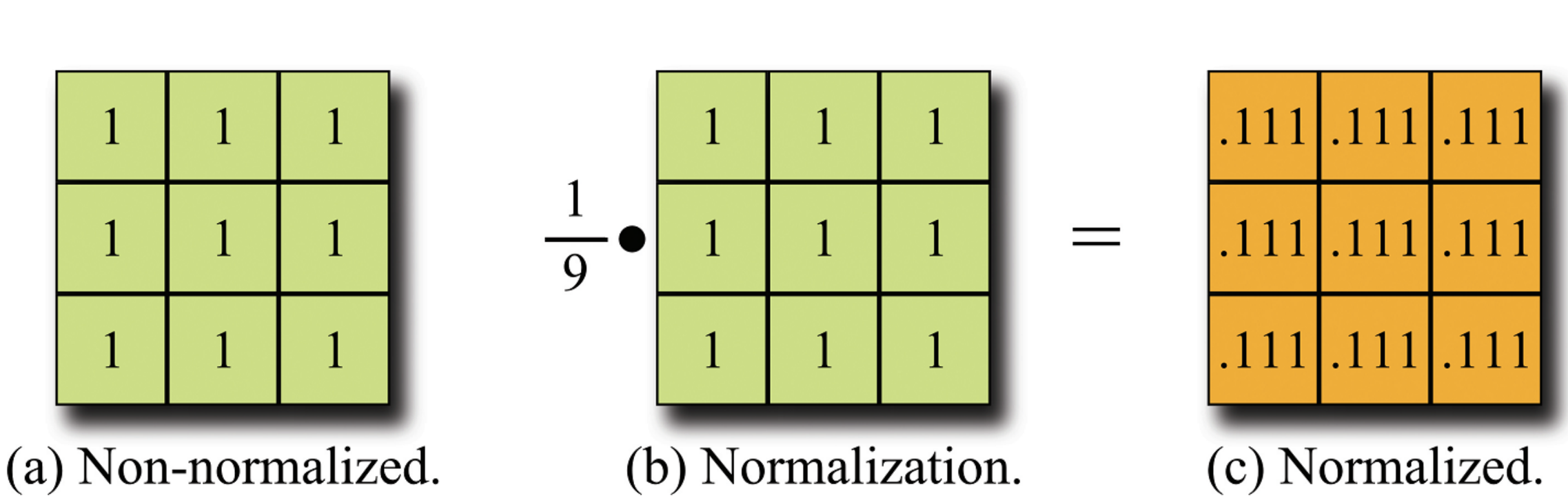
Boundary Padding

- Either we redefine convolution to:
  - Produce zero when the kernel falls off of the boundary. If the kernel extends beyond the source image when centered on a sample I(x, y) then the output sample is set to zero.
  - Produce I(x, y) when the kernel falls off the boundary. If the kernel extends beyond the source image when centered on a sample I(x,y) then the output sample is defined as I(x, y).
- Or else we use boundary padding:

Figure 6.5. (a) Zero padding, (b) circular indexing, and (c) reflected indexing.

Kernel Rescaling

- Important: rescale the kernel by making the coefficients sum to 1.
  - The effect of the kernel is unchanged
  - Rescaling the kernel is equivalent to rescaling the convolved image.



(a) Non-normalized. (b) Normalization. (c) Normalized.

Figure 6.6. Normalizing a kernel.

Smoothing Filters

Smoothing Spatial Filters

- Any weighted filter with positive values will smooth in some way, examples:

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

$$\frac{1}{16}\times\begin{array}{|c|c|c|}\hline 1 & 2 & 1\\\hline 2 & 4 & 2\\\hline 1 & 2 & 1\\\hline\end{array}$$

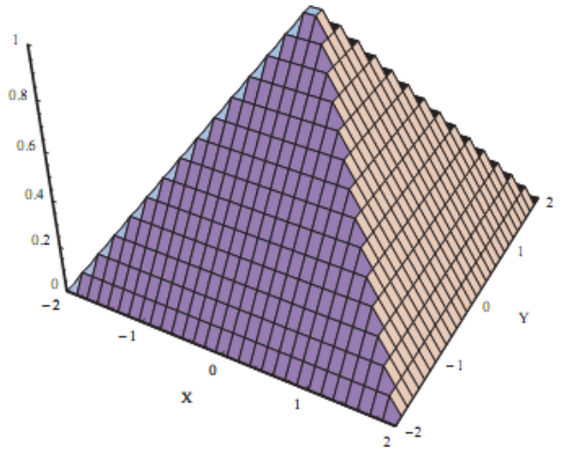
- Normally, we use integers in the filter, and then divide by the sum (computationally more efficient)

- These are also called **blurring** or **low-pass** filters

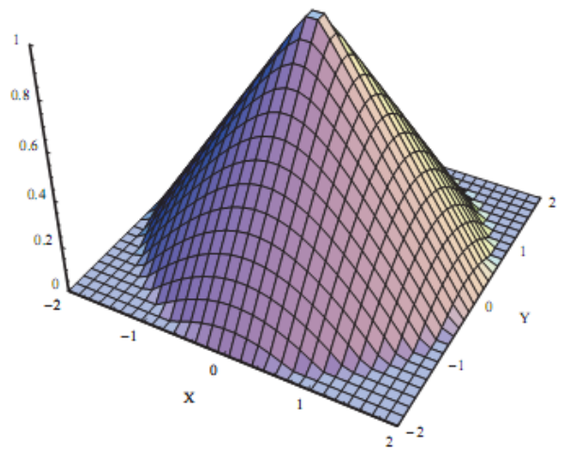
Smoothing Kernels

$$f(x,y) = -\alpha \cdot \max(|x|,|y|)$$

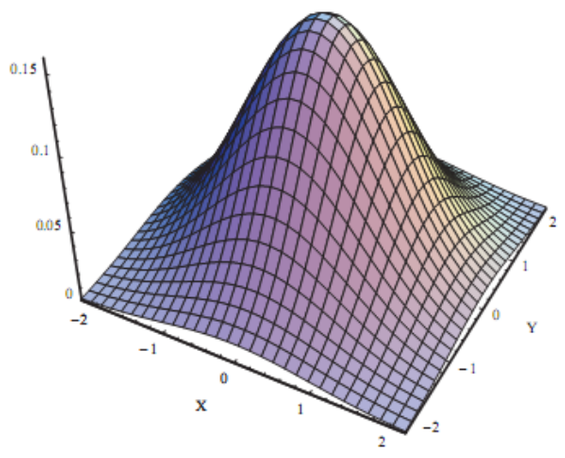
$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{\frac{-(x^2+y^2)}{2\sigma^2}}$$



(a) Pyramid.



(b) Cone.



(c) Gaussian.

$$f(x,y) = -\alpha \cdot \sqrt{x^2+y^2}$$

1	2	3	2	1
2	4	6	4	2
3	6	9	6	3
2	4	6	4	2
1	2	3	2	1

(a) Pyramid.

0	0	1	0	0
0	2	2	2	0
1	2	5	2	1
0	2	2	2	0
0	0	1	0	0

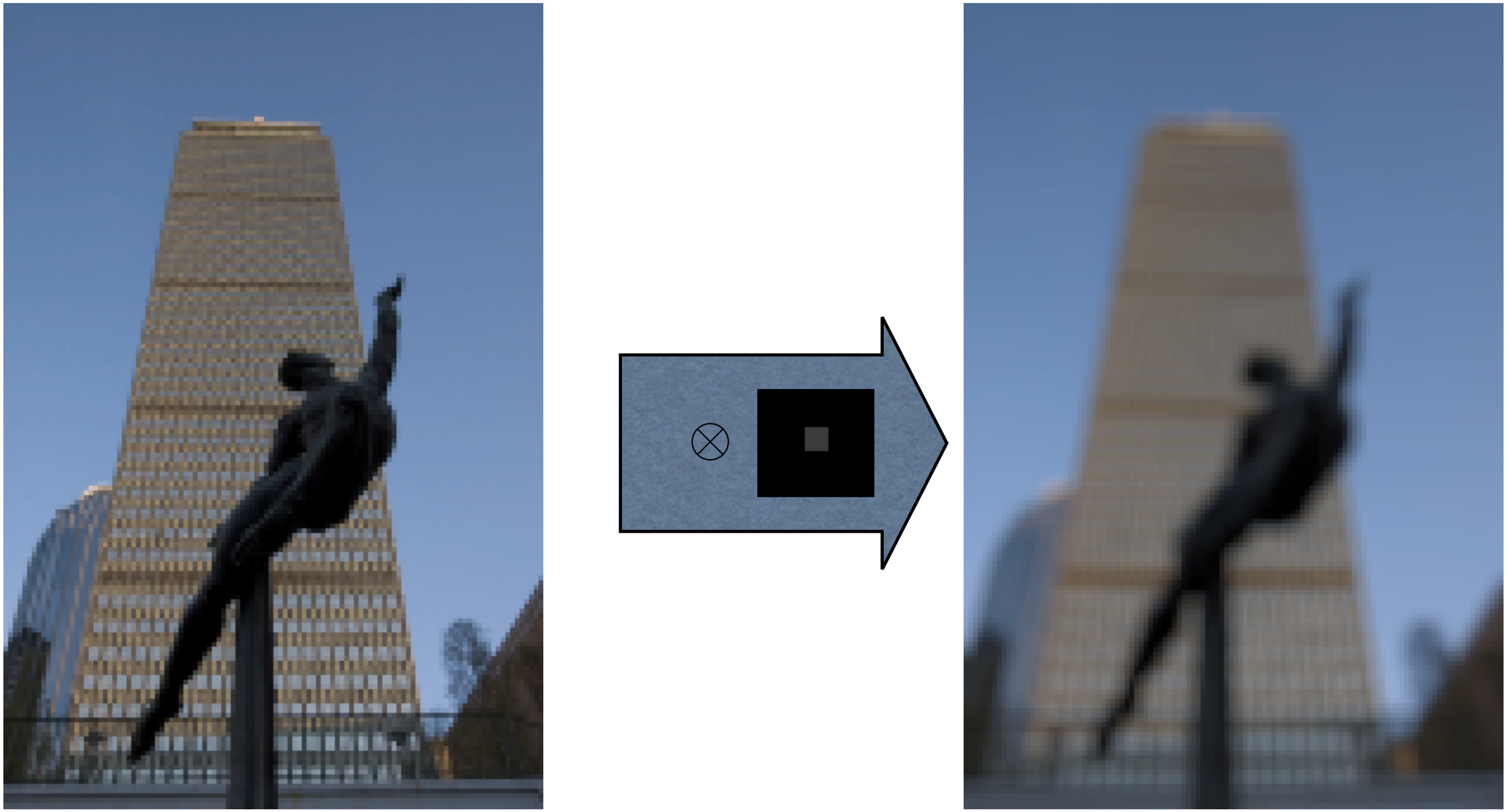
(b) Cone.

1	4	7	4	1
4	16	28	16	4
7	28	49	28	7
4	16	28	16	4
1	4	7	4	1

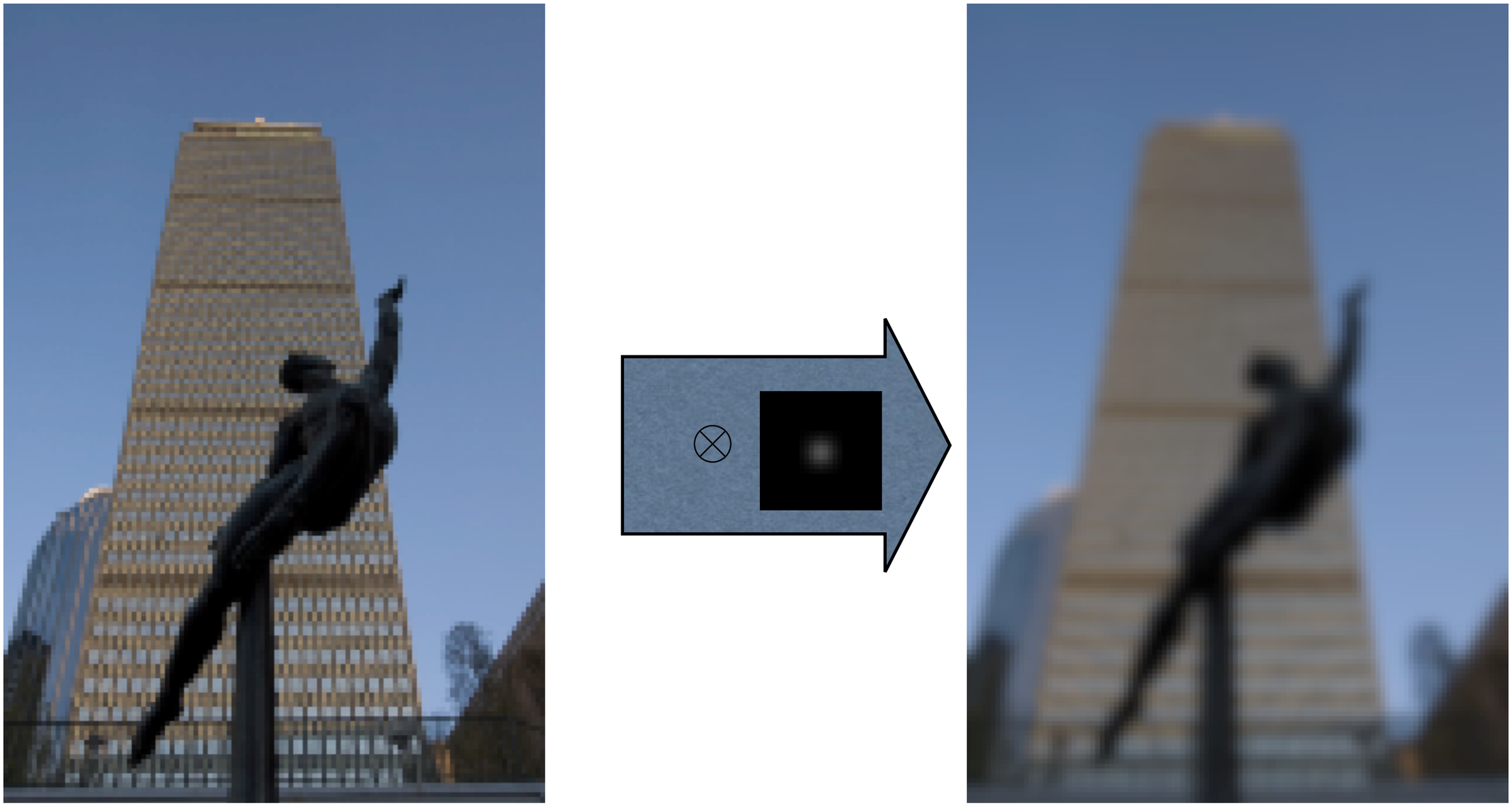
(c) Gaussian.

Table 6.1. Discretized kernels.

Box Filter



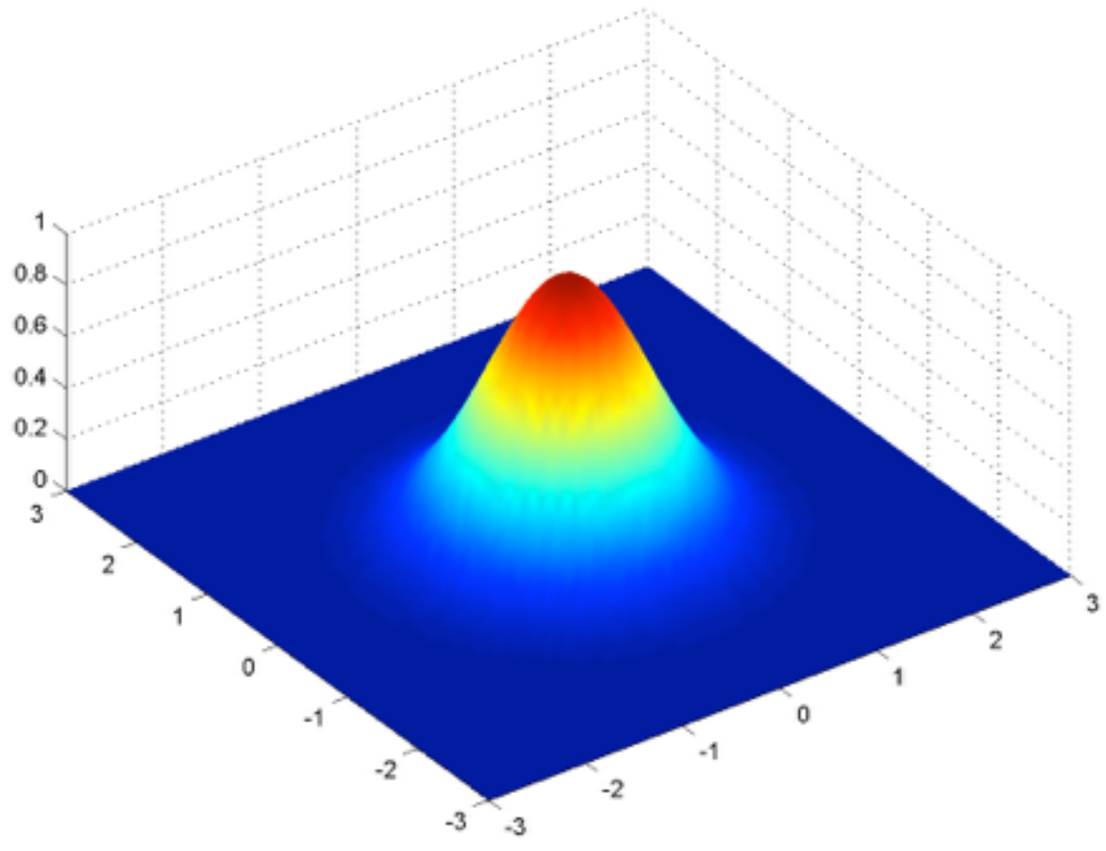
Gaussian Filter



Gaussians

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

- Gaussian kernel is parameterized on the standard deviation σ
- Large σ's reduce the center peak and spread the information across a larger area
- Smaller σ's create a thinner and taller peak
- Gaussians are smooth everywhere.
- Gaussians have infinite **support**
  - >0 everywhere
- But often truncate to 2σ or 3σ



http://en.wikipedia.org/wiki/Gaussian_function

Smoothing Example

(a) Source image.

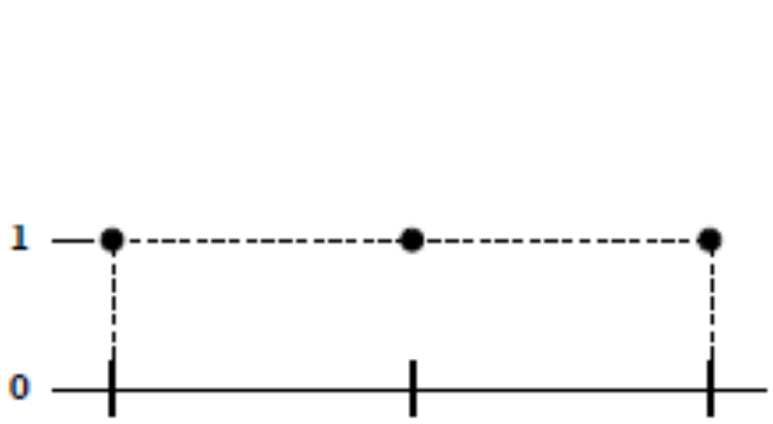
(b) 17 × 17 Box.

(c) 17 × 17 Gaussian.

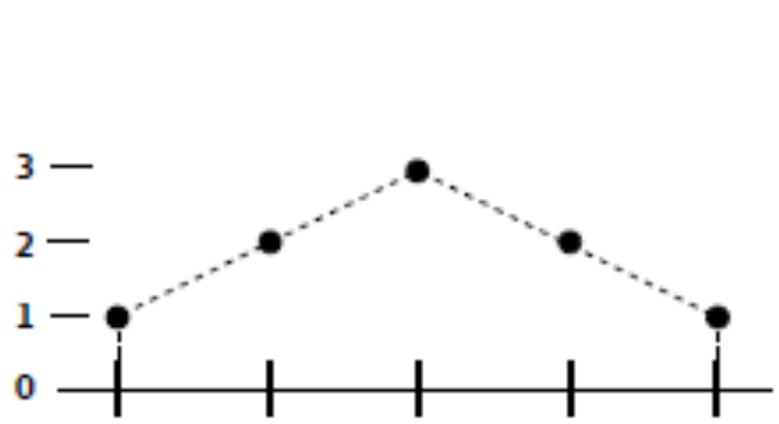
Figure 6.10. Smoothing examples.

Smoothing Smoothing Filters

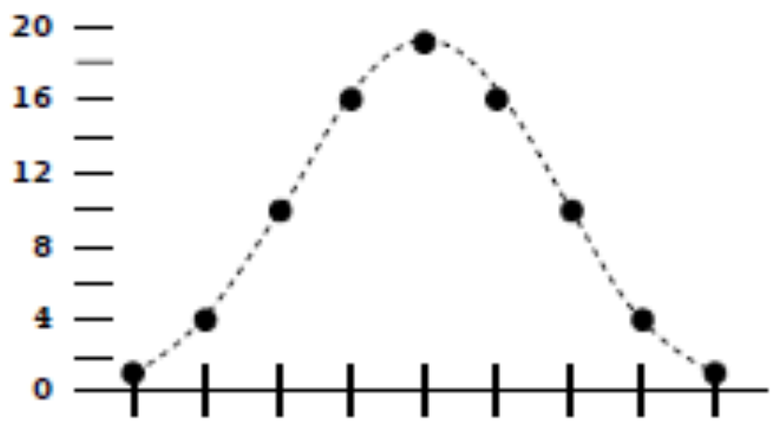
- Box ⊗ Box = Tent (Pyramid)
- Tent ⊗ Tent = Bell (Gaussian)



a) box filter



b) tent filter



c) bell filter

Sharpening Filters

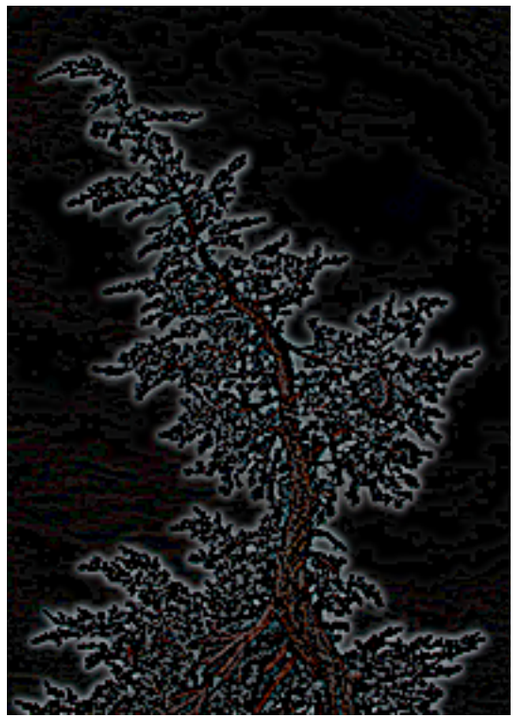
Sharpening (Idea)

Input

-

blurred

=

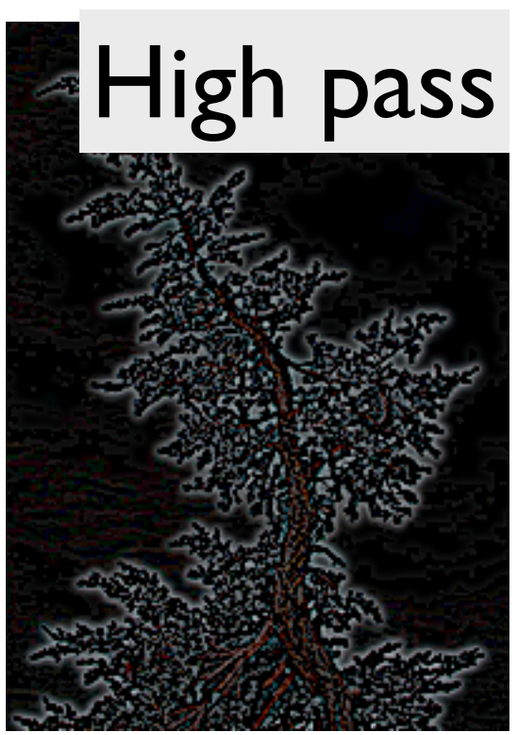


High pass

Input

+k*

High pass



=

Sharpened image

Sharpening is a Convolution

- This equation can then be used to generate appropriate sharpening kernels
  - Assume that $I = I \otimes K_{identity}$ and $I_{low} = I \otimes K_{low}$

$$
\begin{aligned}
I_{\text{sharp}} &= (1+\alpha)I - \alpha I_{\text{low}}, \\
&= (1+\alpha)(I \otimes K_{\text{identity}}) - \alpha(I \otimes K_{\text{low}}), \\
&= I \otimes (1+\alpha)K_{\text{identity}} - I \otimes (\alpha K_{\text{low}}), \\
&= I \otimes ((1+\alpha)K_{\text{identity}} - \alpha K_{\text{low}}).
\end{aligned}
$$

Sharpening is a Convolution

$$
\begin{aligned}
I_{\text{sharp}} &= (1+\alpha)I - \alpha I_{\text{low}}, \\
&= (1+\alpha)(I \otimes K_{\text{identity}}) - \alpha(I \otimes K_{\text{low}}), \\
&= I \otimes (1+\alpha)K_{\text{identity}} - I \otimes (\alpha K_{\text{low}}), \\
&= I \otimes ((1+\alpha)K_{\text{identity}} - \alpha K_{\text{low}}).
\end{aligned}
$$

$$
K_{\text{identity}} = \frac{1}{9} \times \begin{bmatrix} 0 & 0 & 0 \\ 0 & 9 & 0 \\ 0 & 0 & 0 \end{bmatrix},
$$

$$
K_{\text{low}} = \frac{1}{9} \times \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix},
$$

$$
(1+\alpha)K_{\text{identity}} - \alpha K_{\text{low}} = \frac{1}{9} \times \begin{bmatrix} -\alpha & -\alpha & -\alpha \\ -\alpha & (9+8\alpha) & -\alpha \\ -\alpha & -\alpha & -\alpha \end{bmatrix}
$$

Lec13 Required Reading

- Hunt — 3.4, 7.1.3, 7.1.4
- Szeliski — 2.3, 3.5:

http://link.springer.com/book/10.1007/978-1-84882-935-0/page/1