Filtering

D.A. Forsyth,

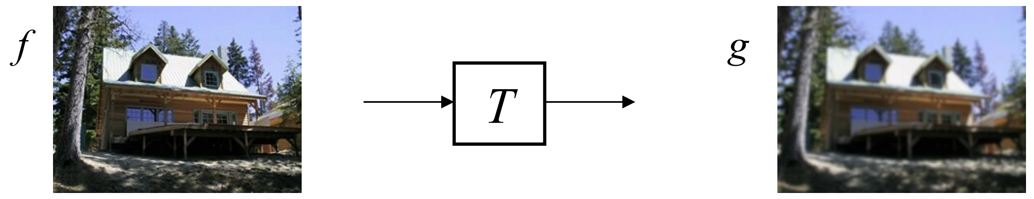
University of Illinois at Urbana Champaign

Some slides adapted from
Svetlana Lazebnik, who adapted from Alyosha Efros, Derek Hoiem

Image filtering

- Roughly speaking, replace image value at x with some function of values in its spatial neighborhood $N(x)$:

$$g(x) = T(f(N(x)))$$



- Examples: smoothing, sharpening, edge detection, etc.

Sliding window operations

- Slide a fixed-size window over the image and perform the same simple computation at each window location
- Example: reduce image noise
  - Take the *average* of pixel values in each window
  - More generally, we can take a *weighted sum* where the weights are given by a *filter kernel*

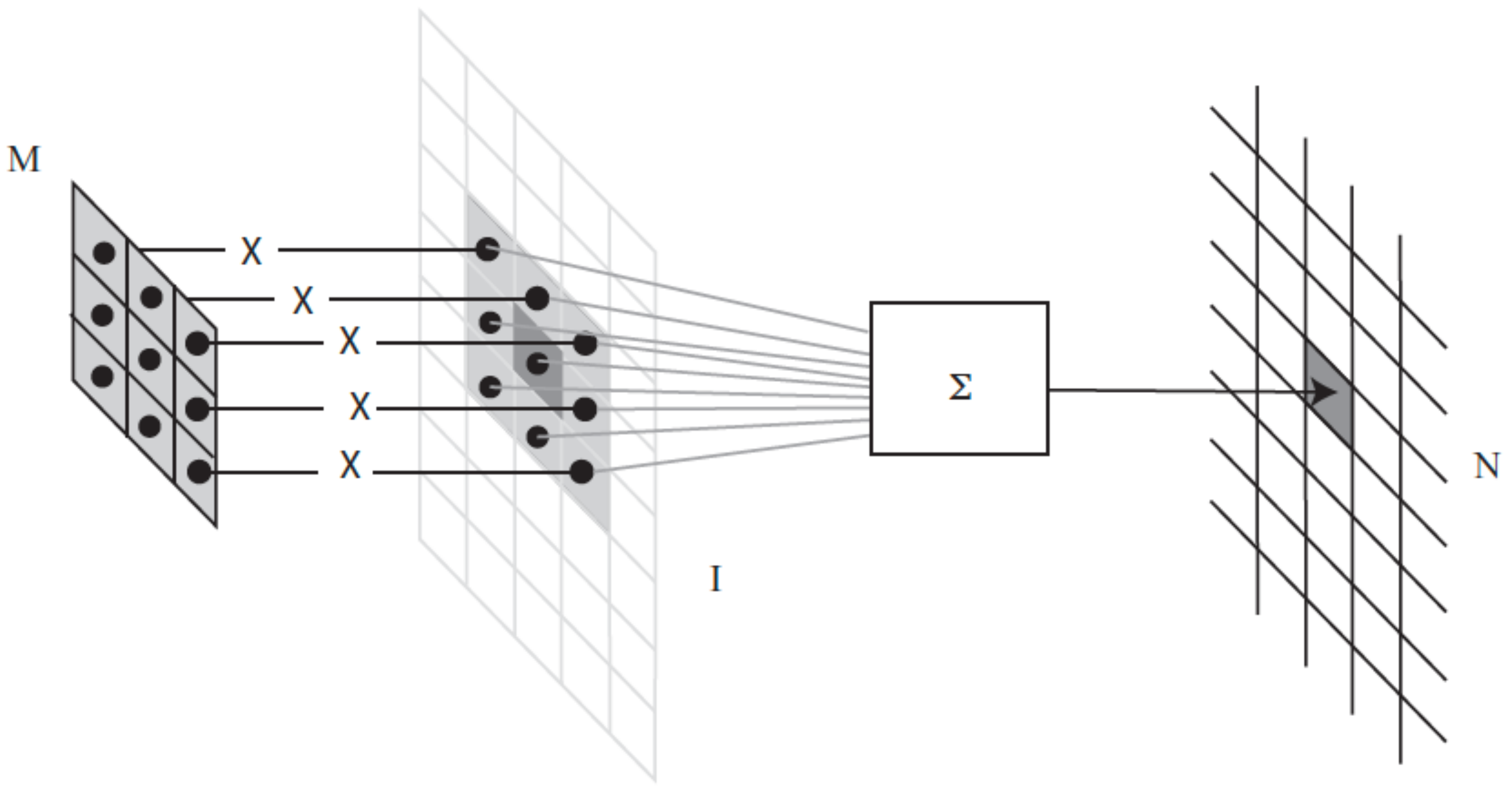
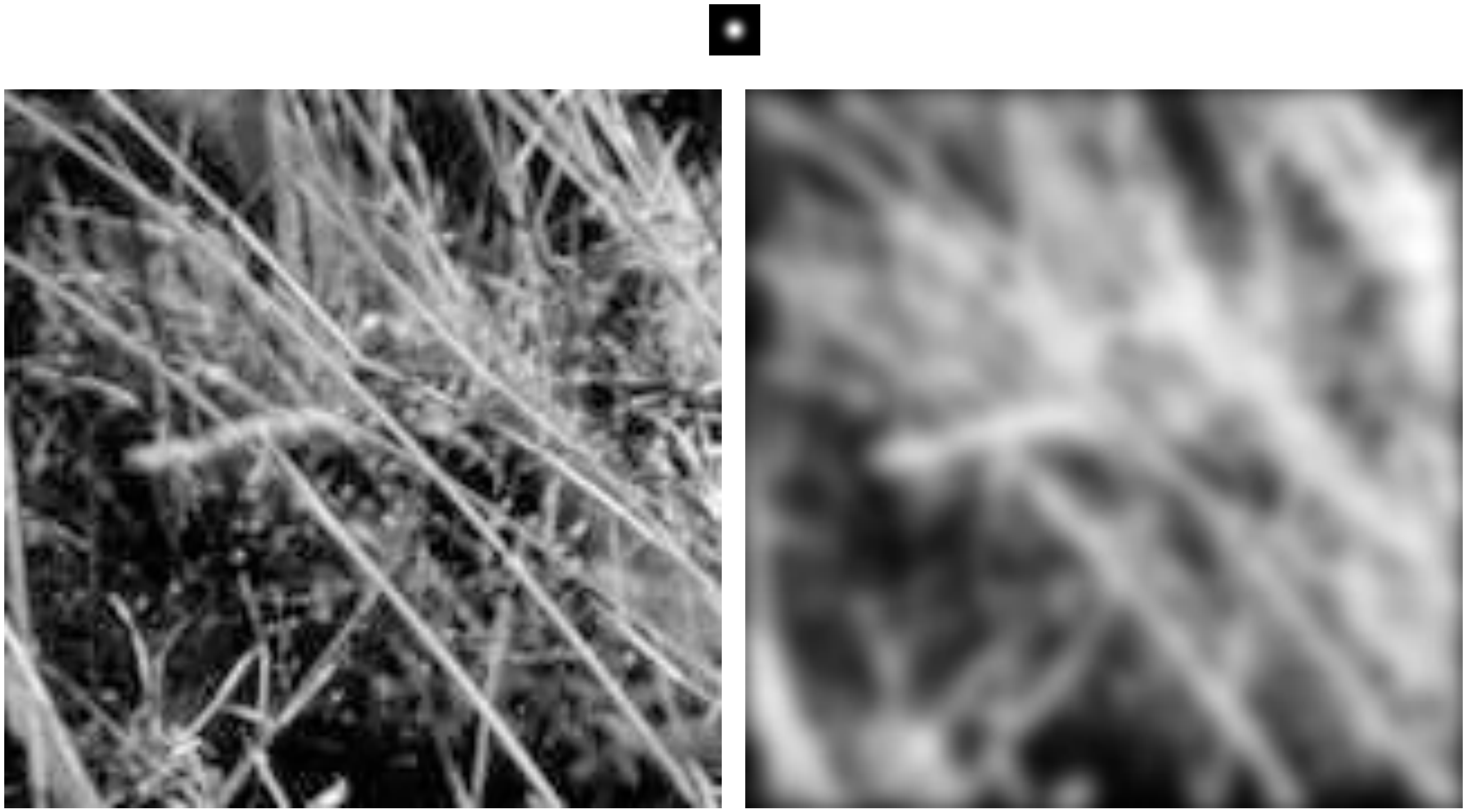
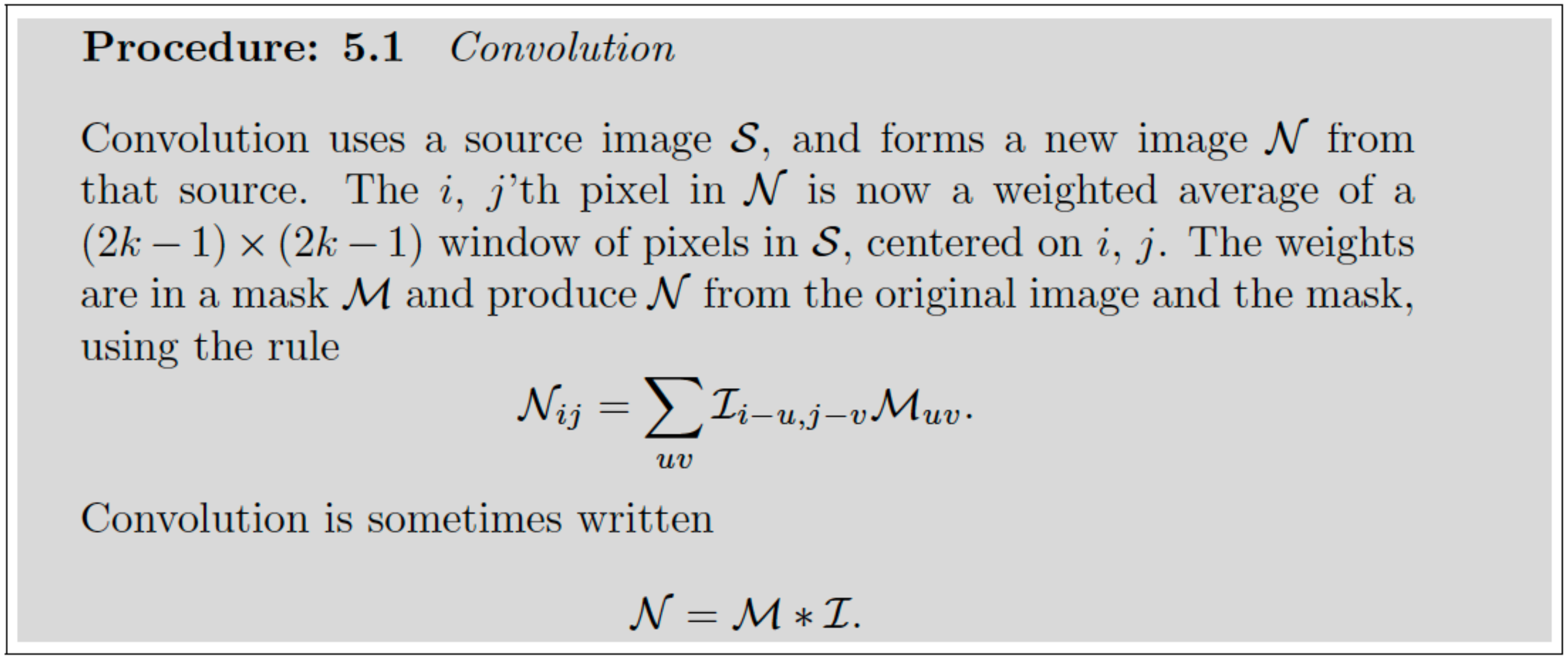


FIGURE 3.1: *To compute the value of $\mathcal{N}$ at some location, you shift a copy of $\mathcal{M}$ (the flipped version of $\mathcal{W}$) to lie over that location in $\mathcal{I}$; you multiply together the non-zero elements of $\mathcal{M}$ and $\mathcal{I}$ that lie on top of one another; and you sum the results.*

Image filtering example



Convolution



Procedure: 5.1 *Convolution*

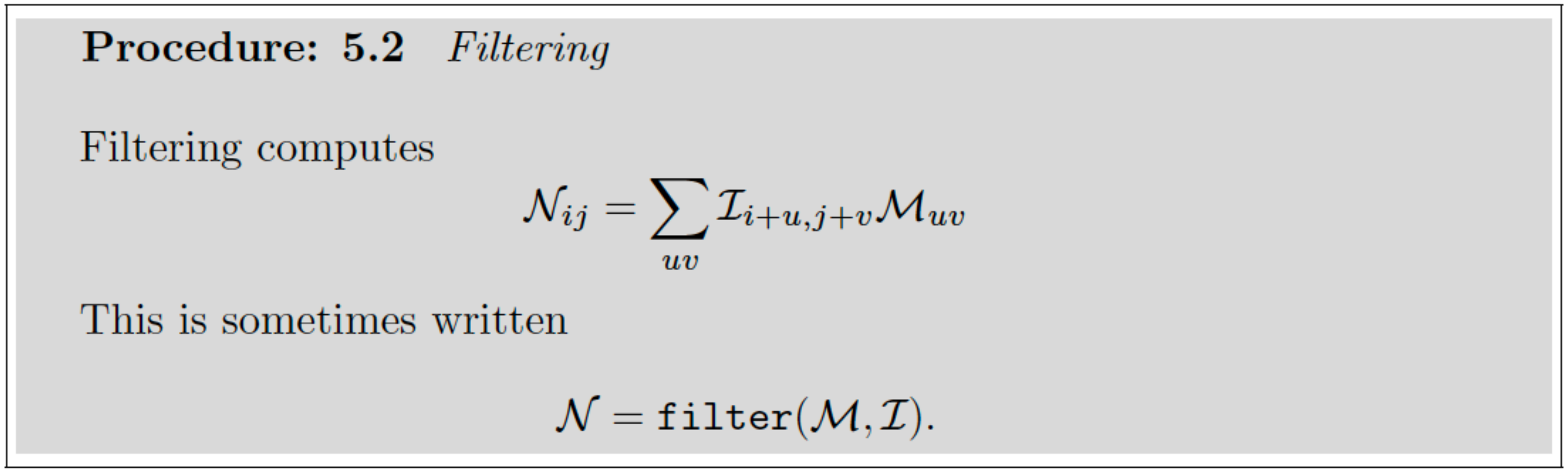
Convolution uses a source image $\mathcal{S}$, and forms a new image $\mathcal{N}$ from that source. The i, j'th pixel in $\mathcal{N}$ is now a weighted average of a $(2k-1)\times(2k-1)$ window of pixels in $\mathcal{S}$, centered on i, j. The weights are in a mask $\mathcal{M}$ and produce $\mathcal{N}$ from the original image and the mask, using the rule

$$\mathcal{N}_{ij} = \sum_{uv} \mathcal{I}_{i-u,j-v}\mathcal{M}_{uv}.$$

Convolution is sometimes written

$$\mathcal{N} = \mathcal{M} * \mathcal{I}.$$

Filtering



Procedure: 5.2 *Filtering*

Filtering computes

$$\mathcal{N}_{ij} = \sum_{uv} \mathcal{I}_{i+u,j+v} \mathcal{M}_{uv}$$

This is sometimes written

$$\mathcal{N} = \texttt{filter}(\mathcal{M}, \mathcal{I}).$$

Convolution and filtering differ only by how the mask is indexed. The difference matters in some contexts, but not (much) to us (and is widely ignored).

Shifting

Define the operation $\texttt{shift}(\mathcal{I}, m, n)$ which shifts an image so that the i, j'th pixel is moved to the $i-m$, $j-n$'th pixel, so

$$\texttt{shift}(\mathcal{I}, m, n)_{ij} = \mathcal{I}_{i-m, j-n}.$$

Ignore the question of the range, as **`shift`** just relabels pixel locations.

Shift invariant linear systems

- **Superposition:** the response to the sum of stimuli is the sum of the individual responses, so

$$R(f+g) = R(f) + R(g);$$

- **Scaling:** the response to a scaled stimulus is a scaled version of the response to the original stimulus, so

$$R(kf) = kR(f).$$

  An operation that exihibits superposition and scaling is *linear*.

- **Shift invariance:** In a shift invariant linear system, the response to a translated stimulus is just a translation of the response to the stimulus. This means that, for example, if a view of a small light aimed at the center of the camera is a small, bright blob, then if the light is moved to the periphery, the response is same small, bright blob, only translated.

Shift invariant linear systems

A device that is linear and shift invariant is known as a *shift invariant linear system*. The operation represented by the device is a *shift invariant linear operation*.

Some systems accept a continuous signal and produce a continuous signal. A natural example is a lens, which takes a pattern of light and produces a pattern of light. Others accept a discrete signal (a vector; an array) and produce a discrete signal. A natural example would be smoothing with a Gaussian. Either kind of system can be linear, and either kind of system can be shift invariant. It turns out that any operation that is shift invariant and linear can be represented by a convolution, and this is true for either continuous or discrete signals.

Properties of convolution

- Linear in
  - image
  - mask (or kernel)
- Shift invariant
  - assuming image is infinite or infinitely padded w/ zeros
- Associative
  - $(f*g)*h=f*(g*h)$

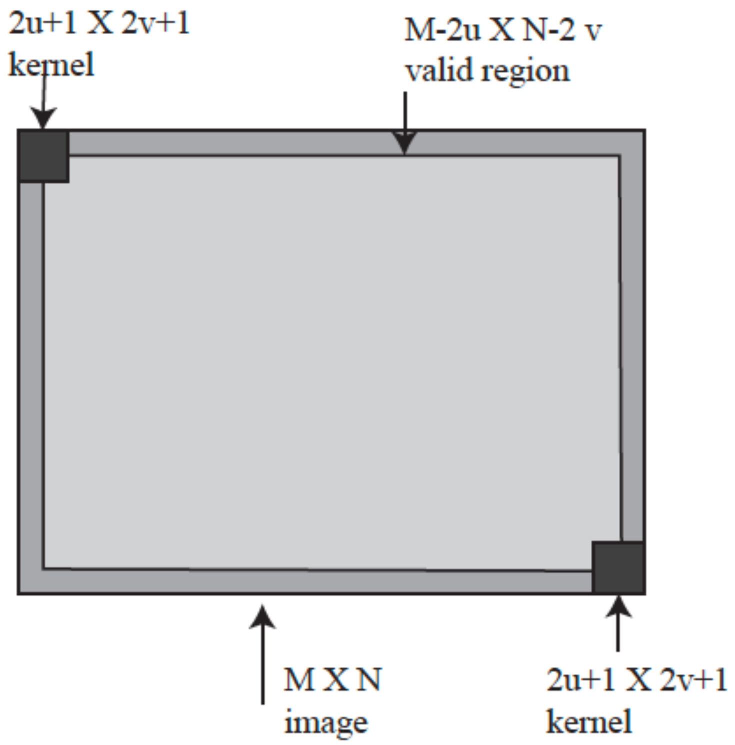
Properties of convolution

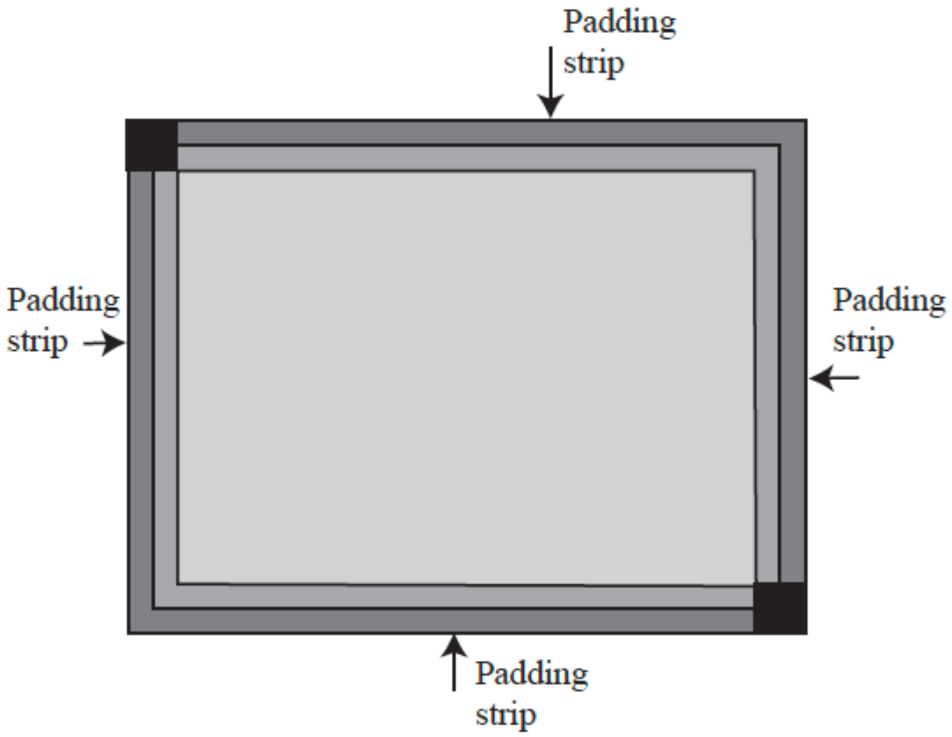
Define the operation $\texttt{shift}(\mathcal{I}, m, n)$ which shifts an image so that the i, j'th pixel is moved to the $i - m$, $j - n$'th pixel, so

$$\texttt{shift}(\mathcal{I}, m, n)_{ij} = \mathcal{I}_{i-m,j-n}.$$

Ignore the question of the range, as `shift` just relabels pixel locations.

Padding and edges





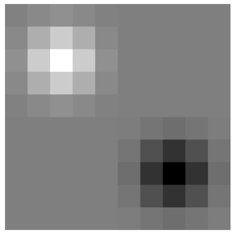
Padding options

- Pad with:
  - Zeros
  - Wrap around
  - Copy edge
  - Reflect across edge

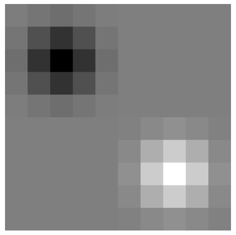
Note: Filtering vs. “convolution”

- In classical signal processing terminology, convolution is filtering with a *flipped* kernel, and filtering with an upright kernel is known as *cross-correlation*
  - Check convention of filtering function you plan to use!

Filtering or “cross-correlation”
(Kernel in original orientation)



“Convolution”
(Kernel flipped in x and y)



Adapted from D. Fouhey and J. Johnson

Practice with linear filters

Original

0	0	0
0	1	0
0	0	0

?

Source: D. Lowe

Practice with linear filters

Original

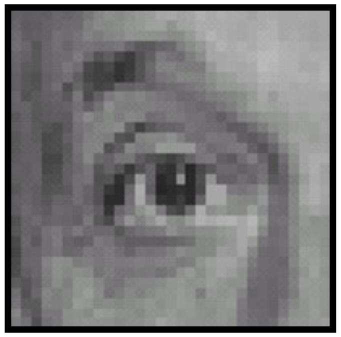
0	0	0
0	1	0
0	0	0

One surrounded by zeros is the *identity filter*

Filtered
(no change)

Source: D. Lowe

Practice with linear filters


Original

0	0	0
0	0	1
0	0	0

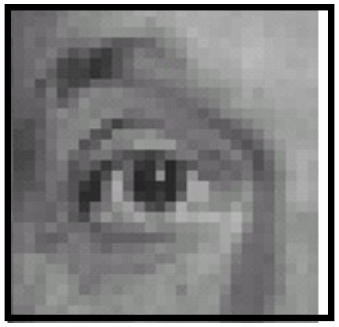
?

Source: D. Lowe

Practice with linear filters

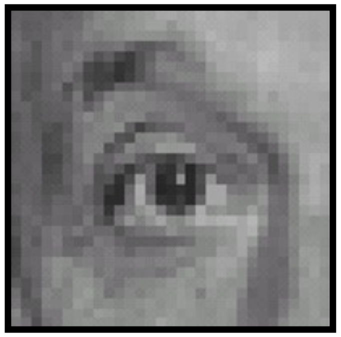
Original

0	0	0
0	0	1
0	0	0


Shifted *left*
By one pixel

Source: D. Lowe

Practice with linear filters


Original

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

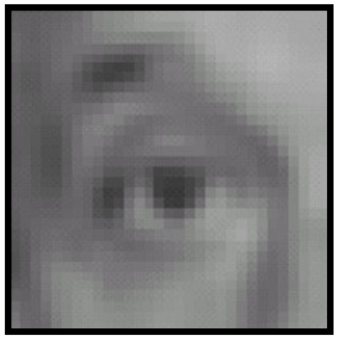
?

Source: D. Lowe

Practice with linear filters

Original

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



Blur (with a box filter)

Source: D. Lowe

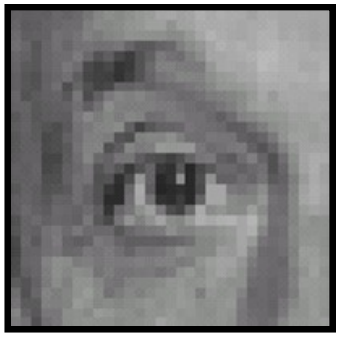
Practice with linear filters

Original

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

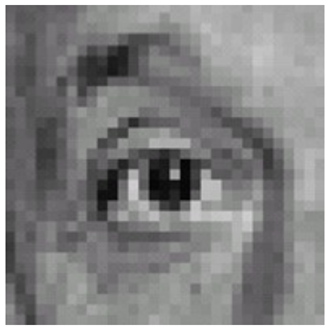
Source: D. Lowe

Practice with linear filters


Original

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

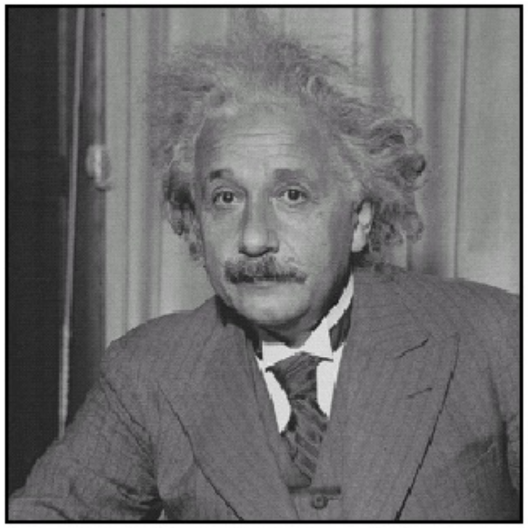
Sharpening filter: Accentuates differences with local average

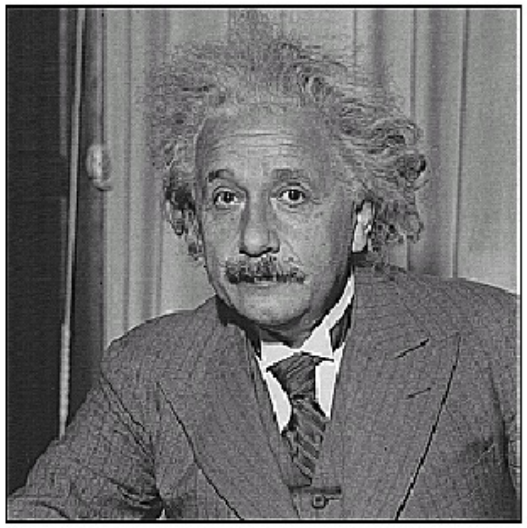

Sharpened

(Note that filter sums to 1)

Source: D. Lowe

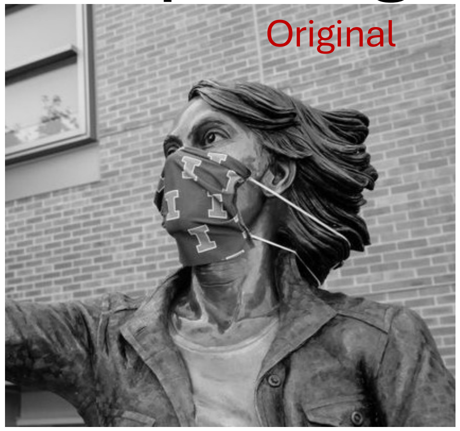
Sharpening


before

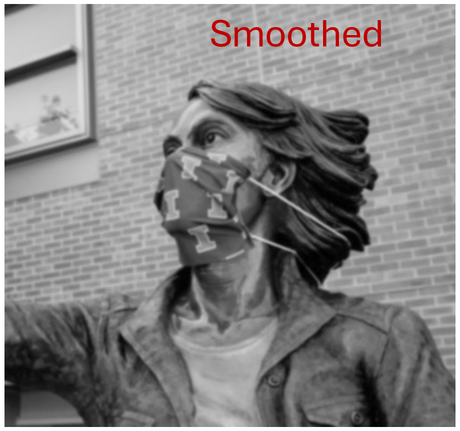

after

Source: D. Lowe

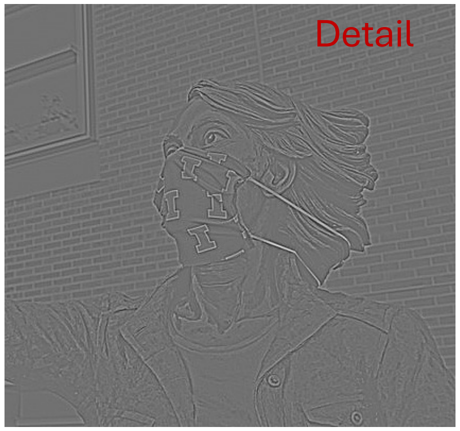
Sharpening

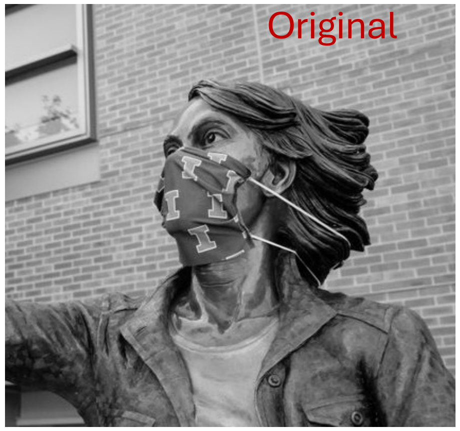


−

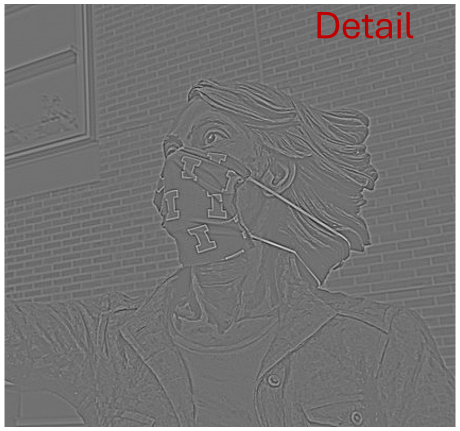


=

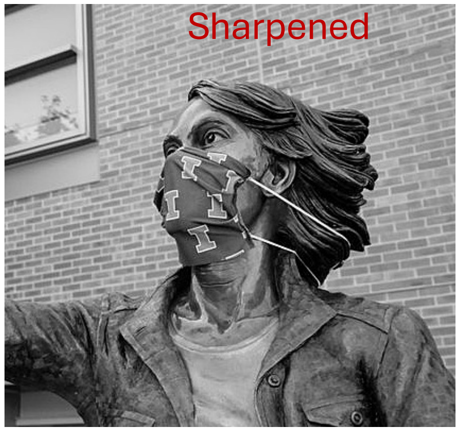




+



=



Source:
S. Gupta

Gradients with finite differences

For an image $\mathcal{I}$, the gradient is

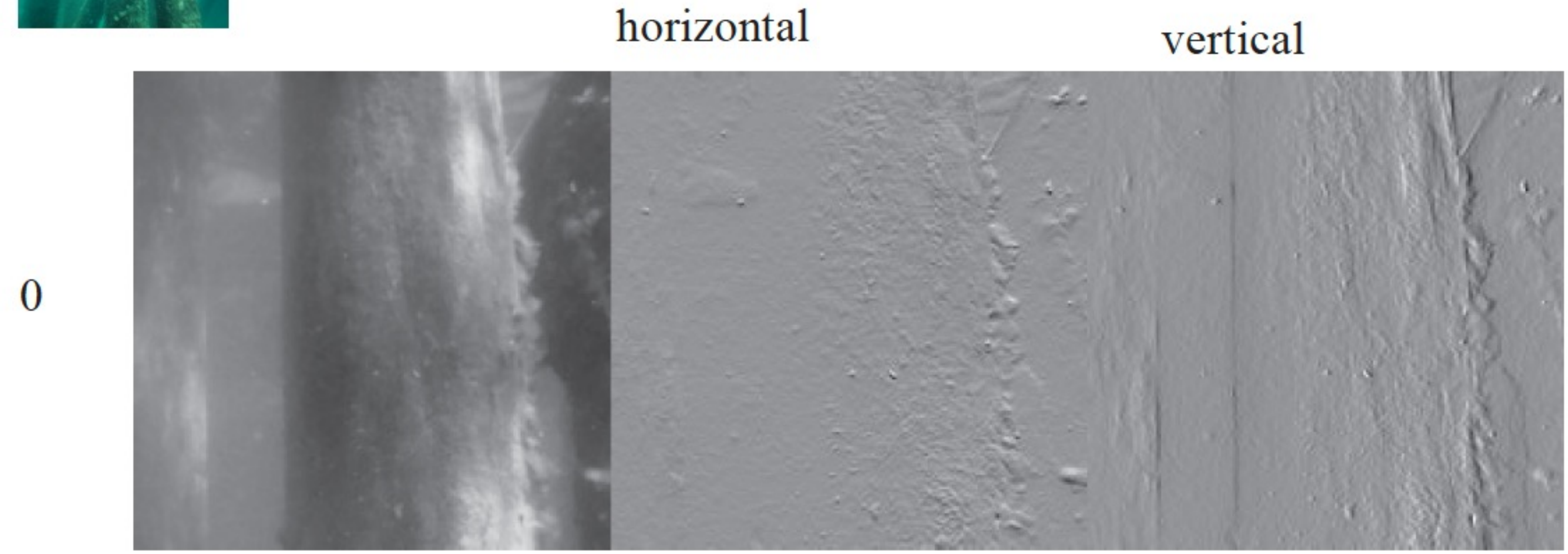
$$\nabla \mathcal{I} = (\frac{\partial \mathcal{I}}{\partial x}, \frac{\partial \mathcal{I}}{\partial y})^T,$$

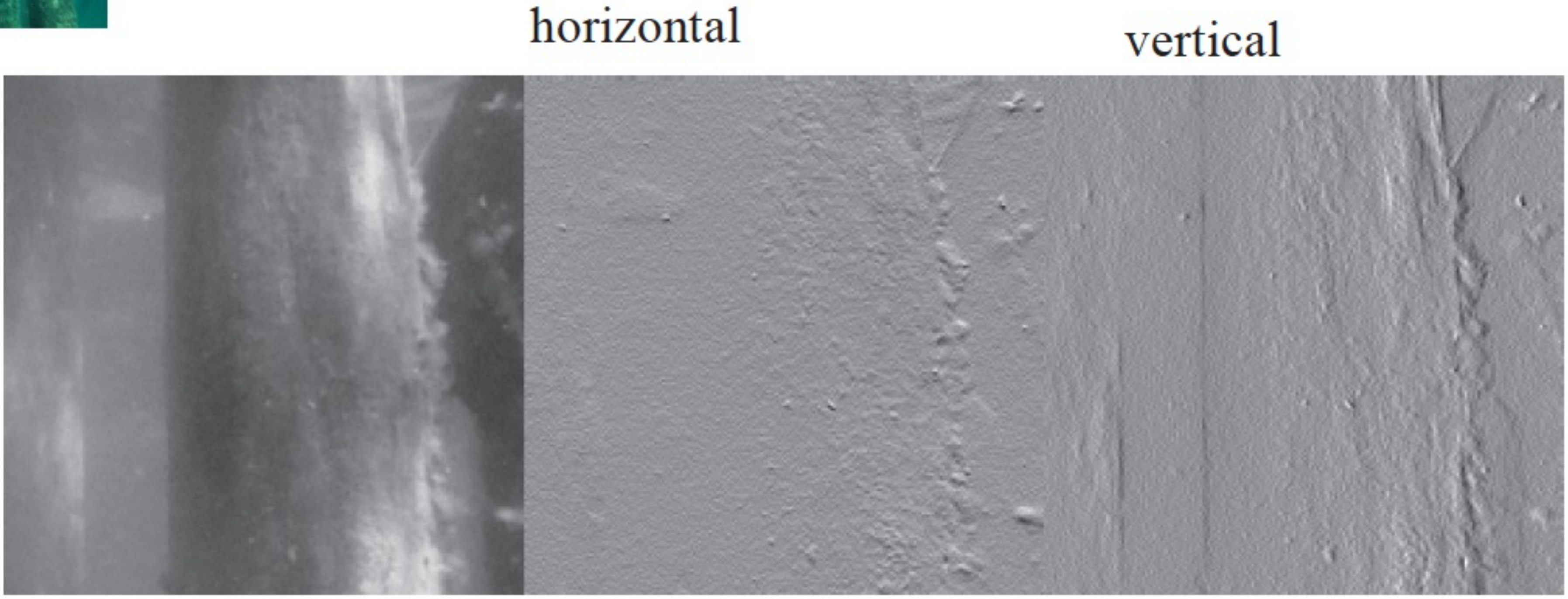
which we could estimate by observing that

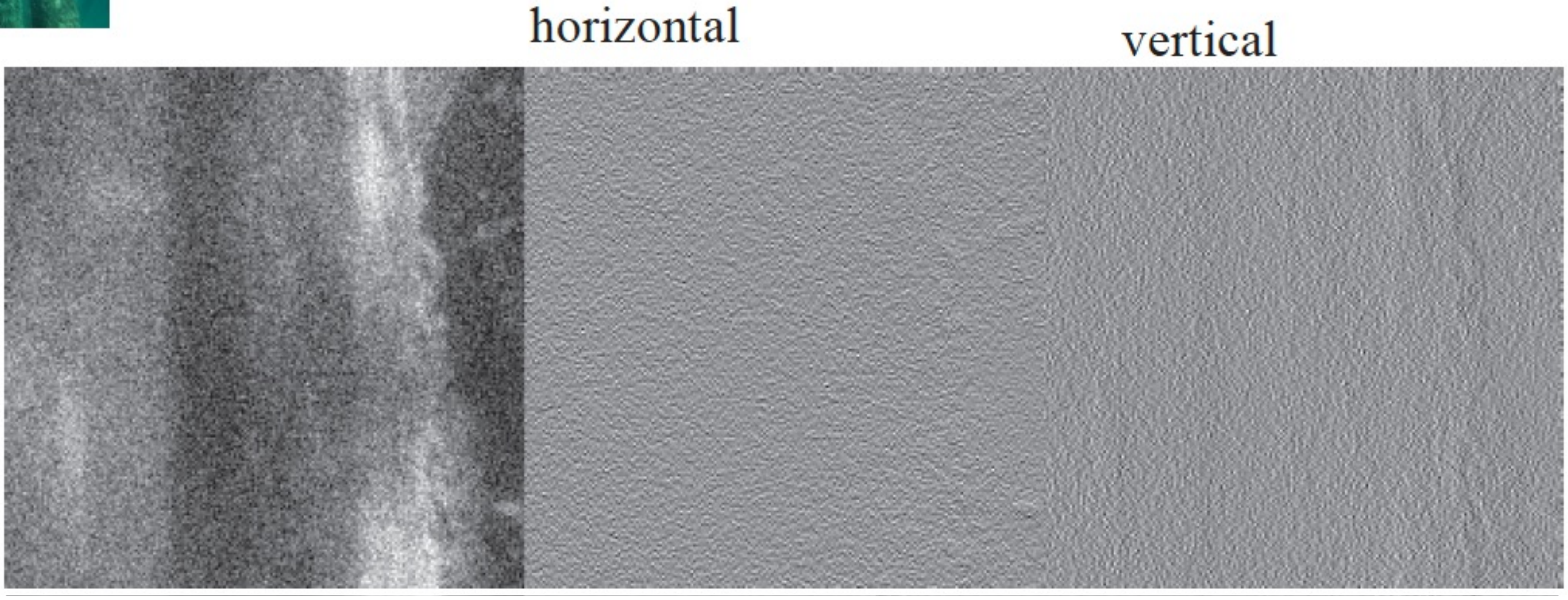
$$\frac{\partial \mathcal{I}}{\partial x} = \lim_{\delta x \to 0} \frac{\mathcal{I}(x + \delta x, y) - \mathcal{I}(x, y)}{\delta x} \approx \mathcal{I}_{i+1,j} - \mathcal{I}_{i,j}.$$

This means a convolution with

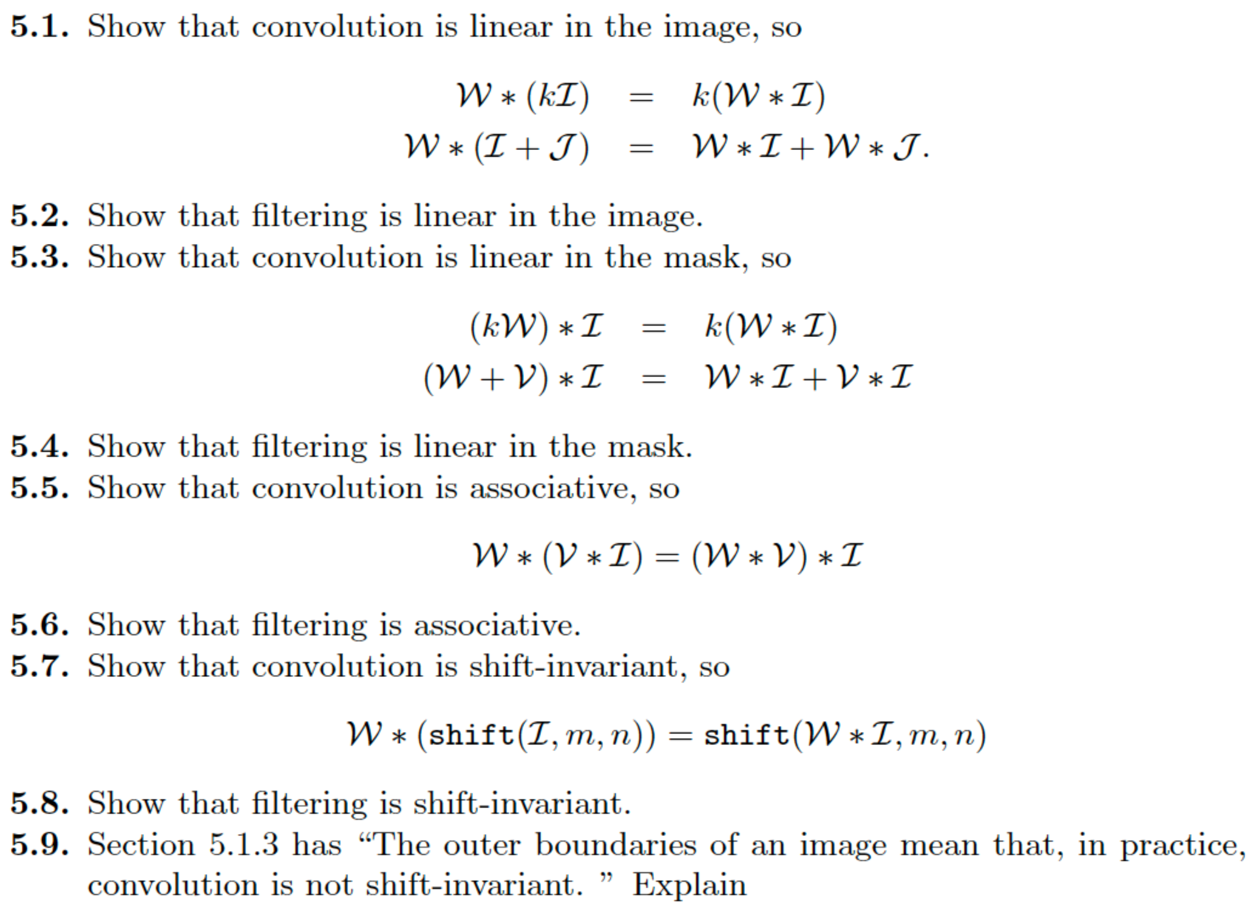
$$\begin{matrix} -1 & 1 \end{matrix}$$


horizontal
vertical
0


horizontal
vertical
0.01


horizontal
vertical
0.1

Think about this...



5.1. Show that convolution is linear in the image, so

$$\begin{aligned} \mathcal{W} * (k\mathcal{I}) &= k(\mathcal{W} * \mathcal{I}) \\ \mathcal{W} * (\mathcal{I} + \mathcal{J}) &= \mathcal{W} * \mathcal{I} + \mathcal{W} * \mathcal{J}. \end{aligned}$$

5.2. Show that filtering is linear in the image.

5.3. Show that convolution is linear in the mask, so

$$\begin{aligned} (k\mathcal{W}) * \mathcal{I} &= k(\mathcal{W} * \mathcal{I}) \\ (\mathcal{W} + \mathcal{V}) * \mathcal{I} &= \mathcal{W} * \mathcal{I} + \mathcal{V} * \mathcal{I} \end{aligned}$$

5.4. Show that filtering is linear in the mask.

5.5. Show that convolution is associative, so

$$\mathcal{W} * (\mathcal{V} * \mathcal{I}) = (\mathcal{W} * \mathcal{V}) * \mathcal{I}$$

5.6. Show that filtering is associative.

5.7. Show that convolution is shift-invariant, so

$$\mathcal{W} * (\texttt{shift}(\mathcal{I}, m, n)) = \texttt{shift}(\mathcal{W} * \mathcal{I}, m, n)$$

5.8. Show that filtering is shift-invariant.

5.9. Section 5.1.3 has "The outer boundaries of an image mean that, in practice, convolution is not shift-invariant. " Explain