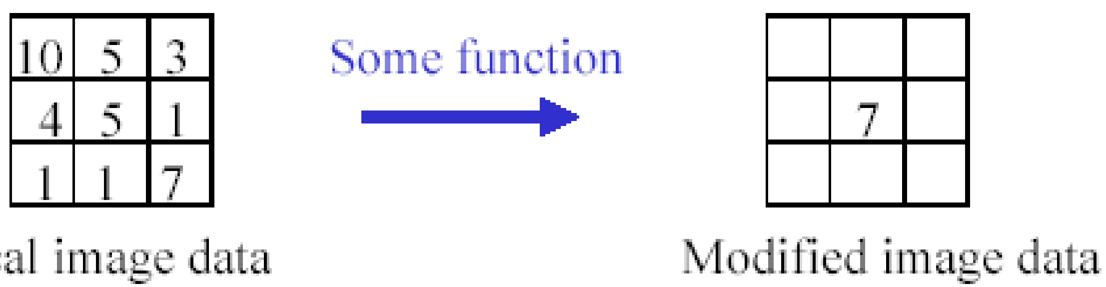
Linear Filtering

- About modifying pixels based on neighborhood. Local methods simplest.
- Linear means linear combination of neighbors. Linear methods simplest.
- Useful to:
  - Integrate information over constant regions.
  - Scale.
  - Detect changes.
- Many nice slides taken from Bill Freeman.

What is image filtering?

- Modify the pixels in an image based on some function of a local neighborhood of the pixels.



(Freeman)

Linear functions

- Simplest: linear filtering.
  - Replace each pixel by a linear combination of its neighbors.
- The prescription for the linear combination is called the “convolution kernel”.

10	5	3
4	5	1
1	1	7

Local image data

0	0	0
0	0.5	0
0	1	0.5

kernel

	7	

Modified image data

(Freeman)

Correlation

Examples on white board – 1D

Examples -2D

For example, let’s take a vector like:

(1 2 3 2 3 2 1), and filter it with a filter like (1/3 1/3 1/3)

Ignoring the ends for the moment, we get a result like:

2 2 1/3 2 2/3 2 1/3 2. We can also graph the results and see that the original vector is smoothed out.

Boundaries

- Zeros
- Repeat values
- Cycle
- Produce shorter result
- *Examples*

Correlation

$$F \circ I(x) = \sum_{i=-N}^{N} F(i)I(x+i)$$

For this notation, we index *F* from –N to N.

$$F \circ I(x,y) = \sum_{j=-N}^{N} \sum_{i=-N}^{N} F(i,j)I(x+i,y+j)$$

Convolution

- Like Correlation with Filter Reversed
- Associative

1D

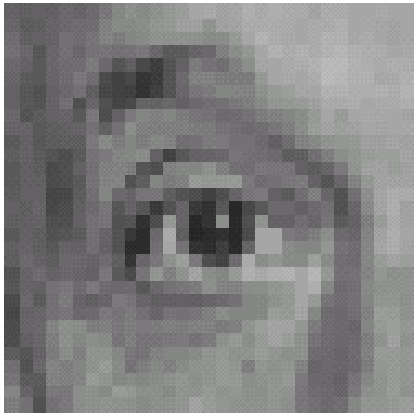
$$F * I(x) = \sum_{i=-N}^{N} F(i) I(x-i)$$

2D

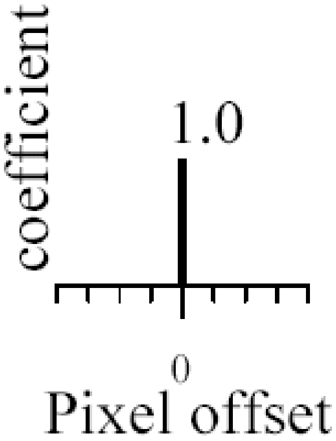
$$F * I(x, y) = \sum_{j=-N}^{N} \sum_{i=-N}^{N} F(i, j) I(x-i, y-j)$$

Some Examples

Linear filtering

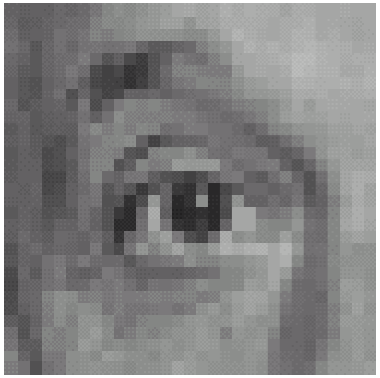


original

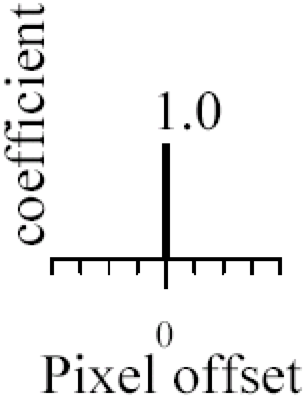


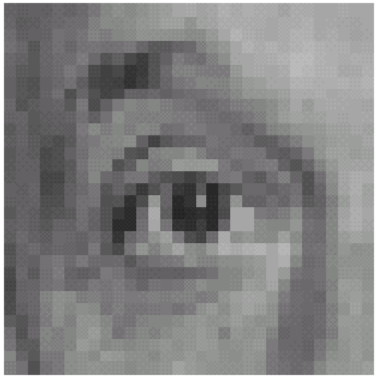
?

Linear filtering



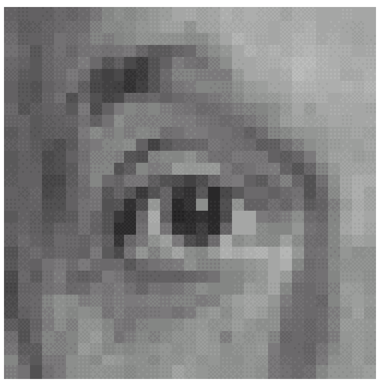
original



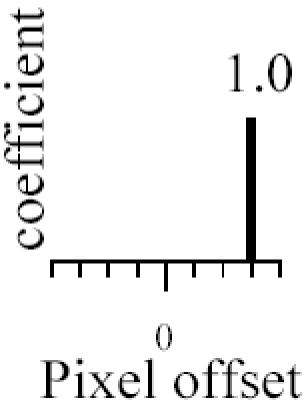


Filtered
(no change)

Linear filtering

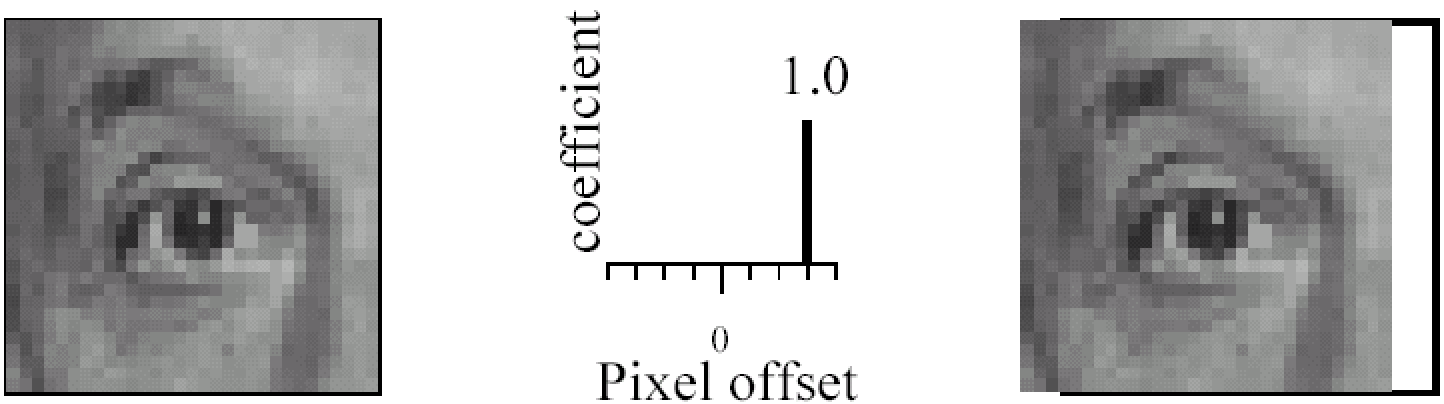


original



?

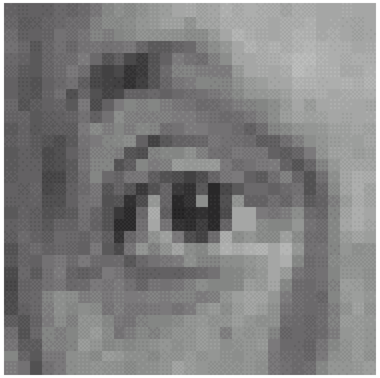
shift



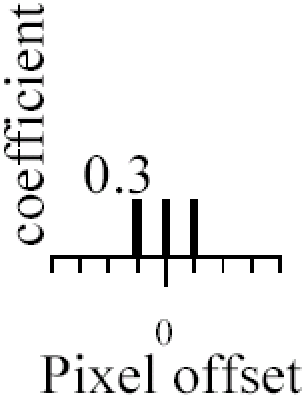
original

shifted

Linear filtering

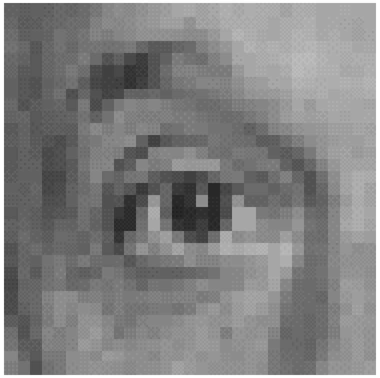


original

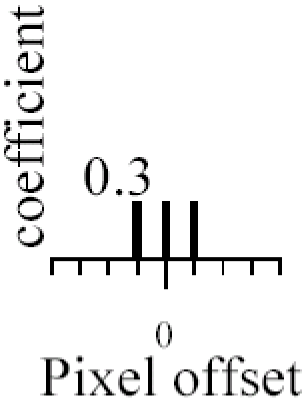


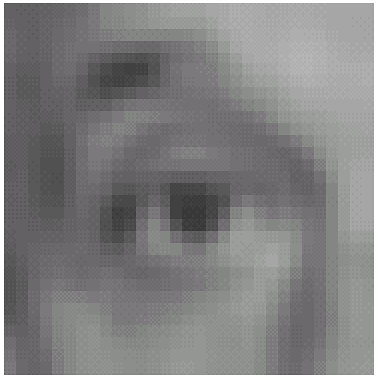
?

Blurring



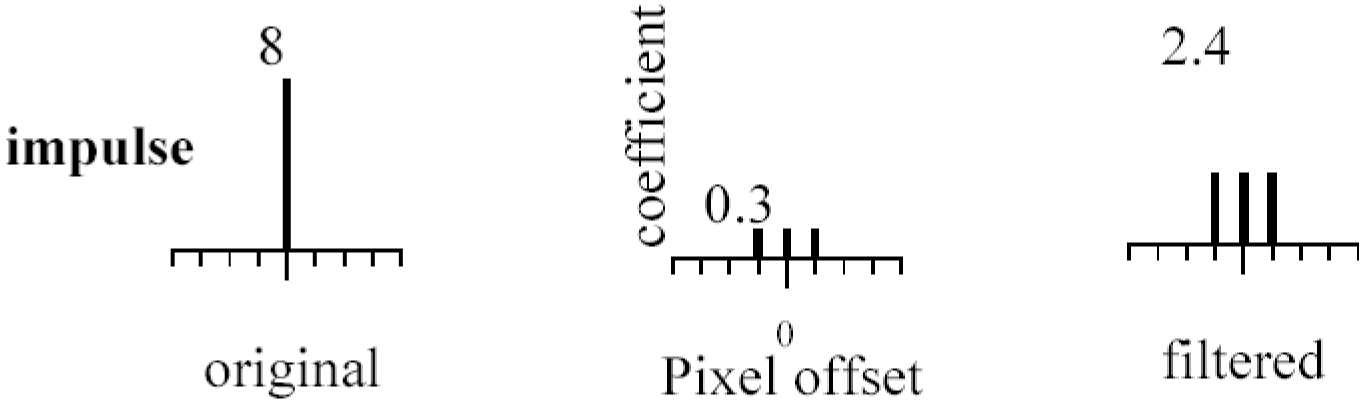
original



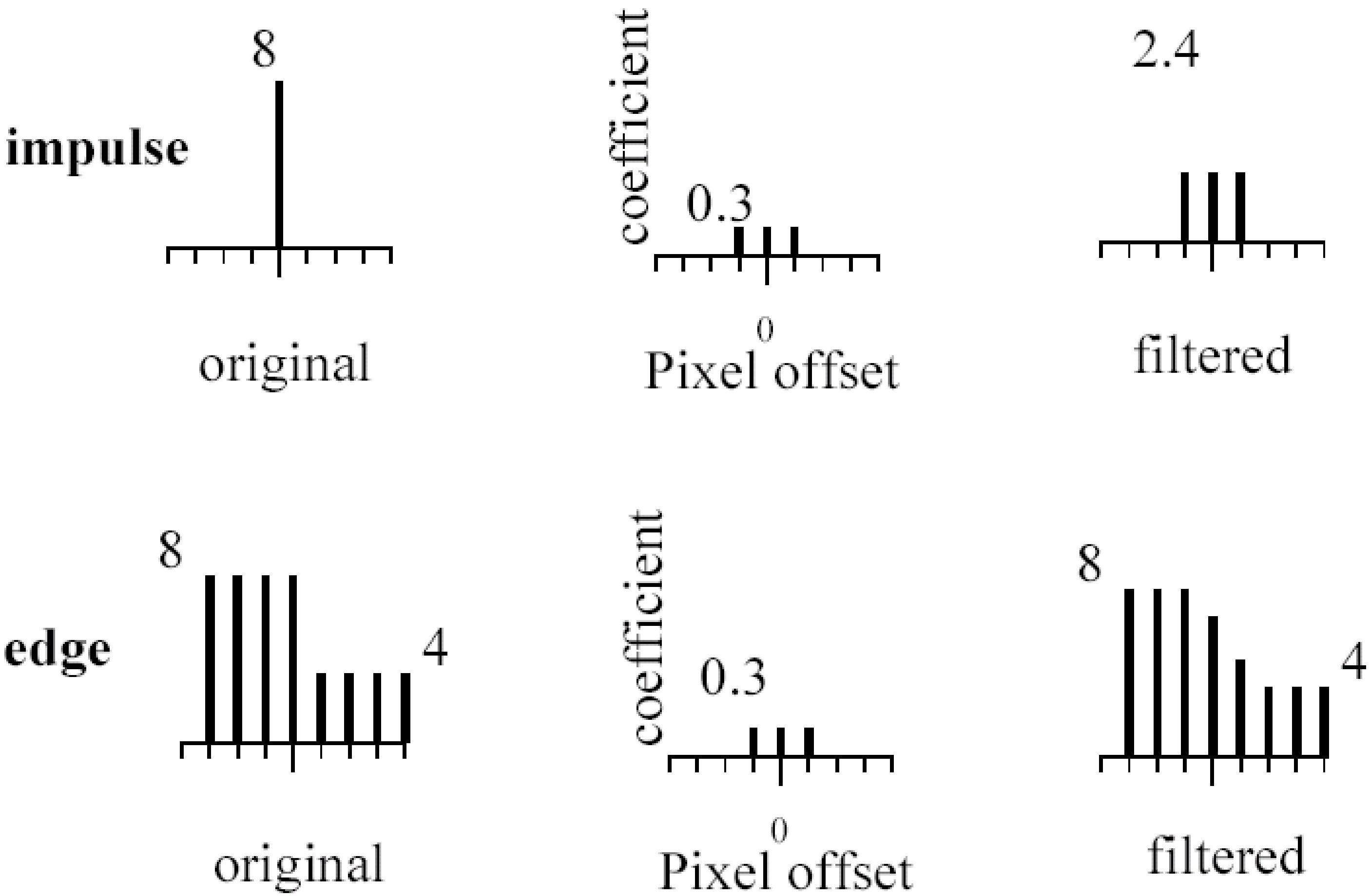


Blurred (filter applied in both dimensions).

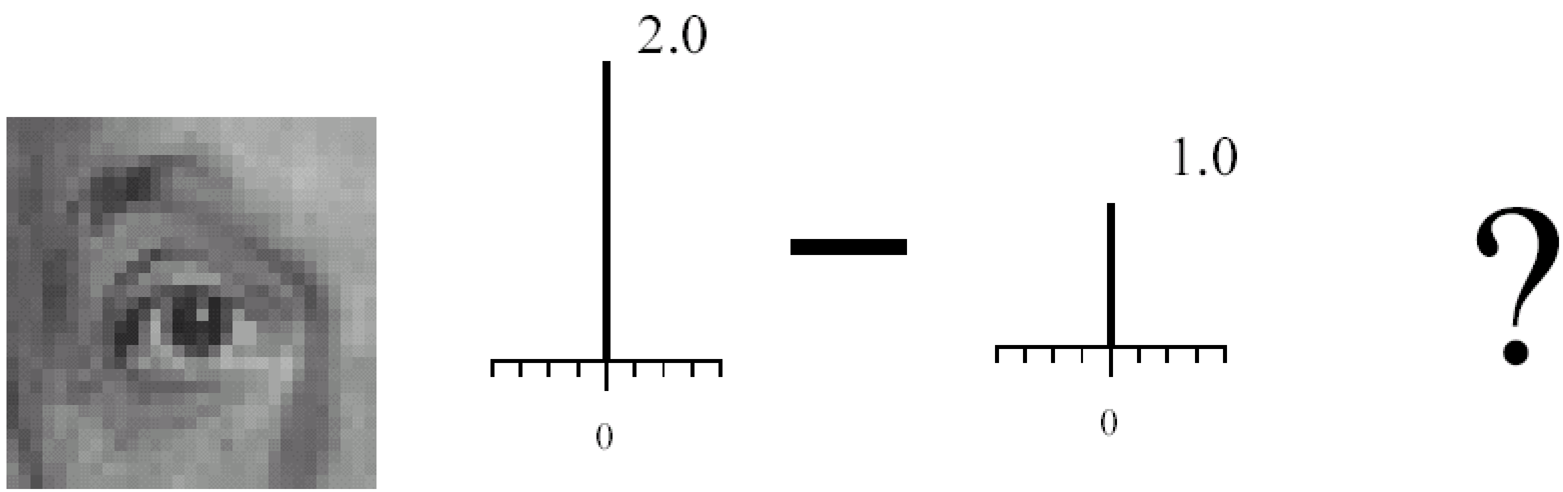
Blur examples



Blur examples

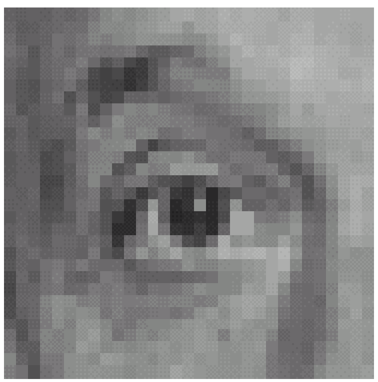


Linear filtering

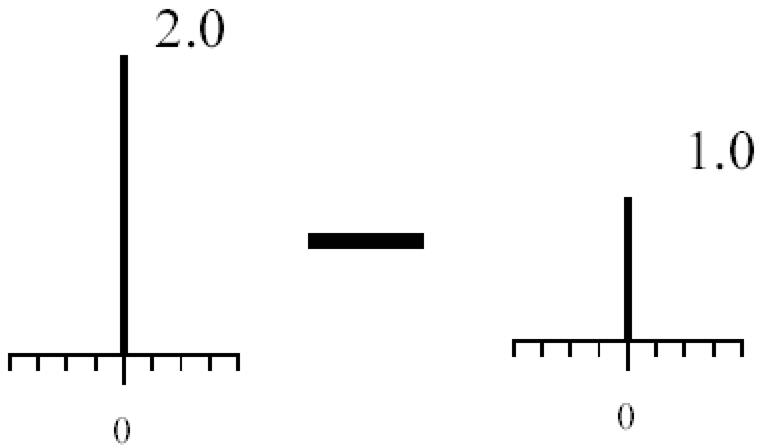


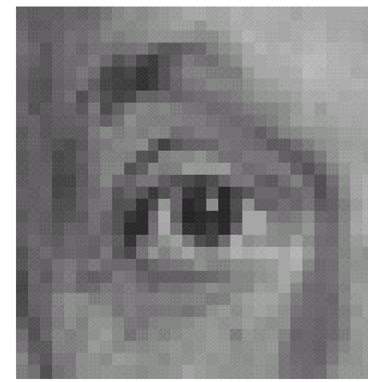
original

Linear filtering (no change)



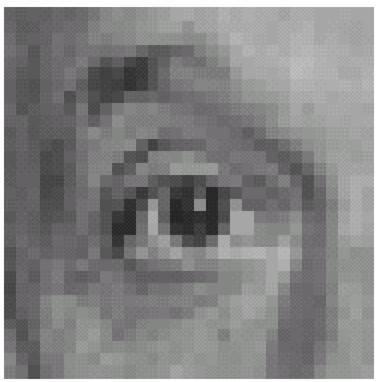
original



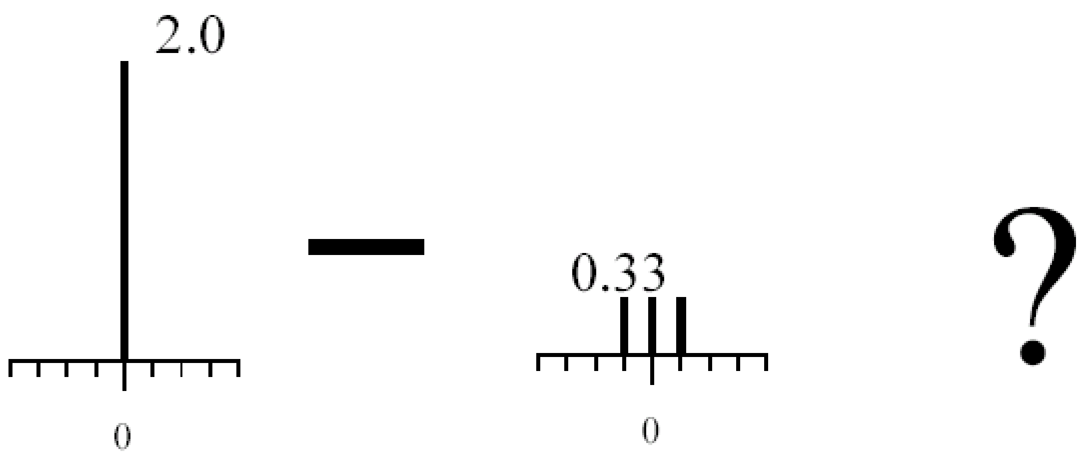


Filtered
(no change)

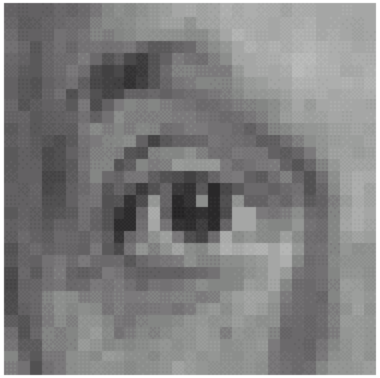
Linear filtering



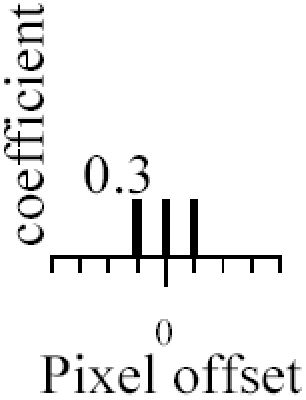
original

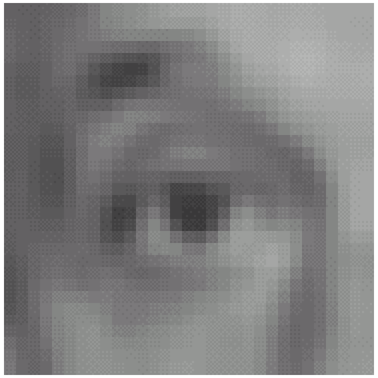


(remember blurring)



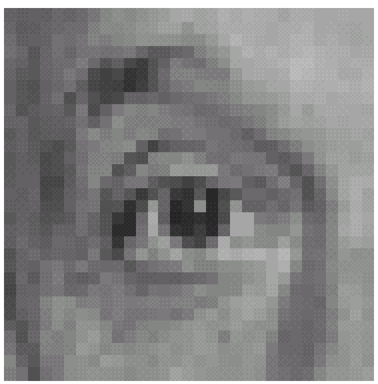
original



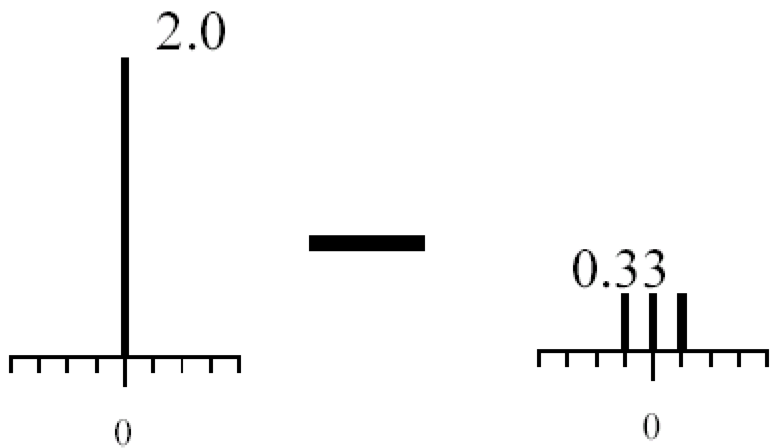


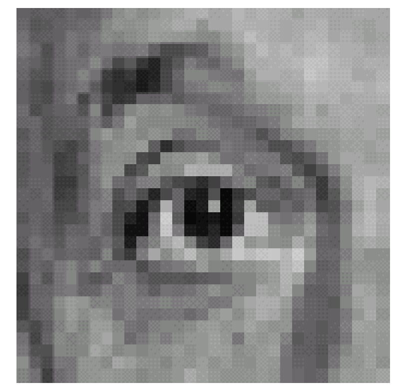
Blurred (filter applied in both dimensions).

Sharpening



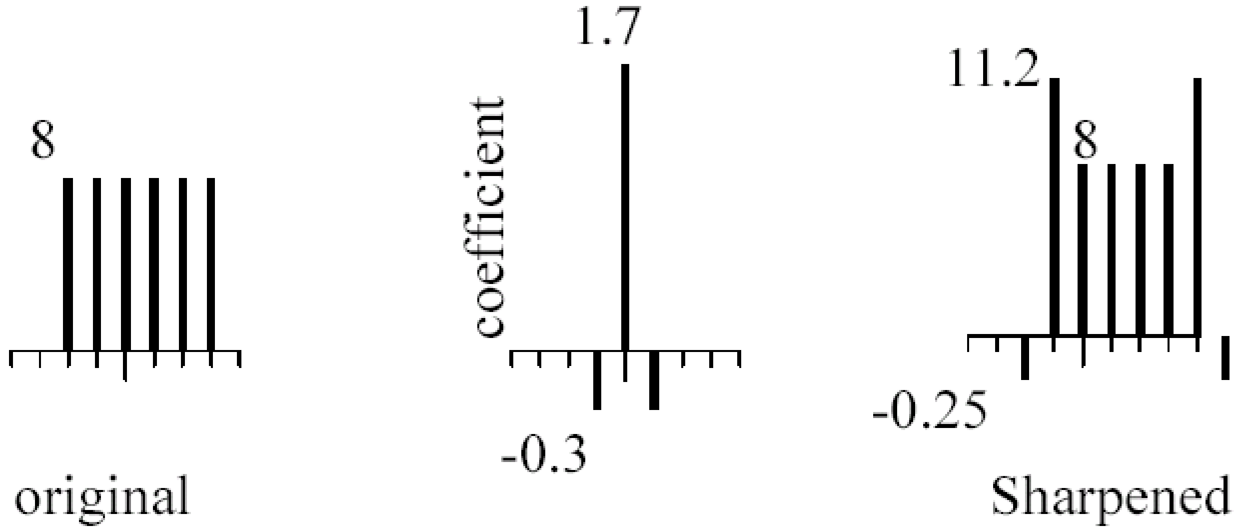
original





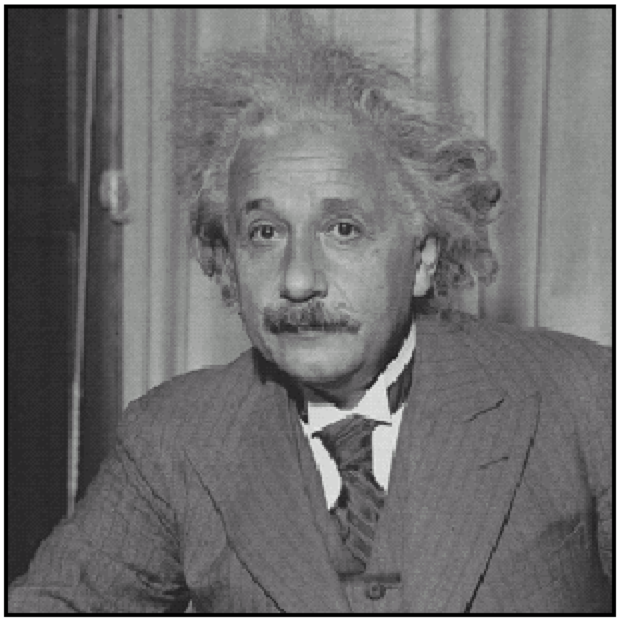
Sharpened
original

Sharpening example

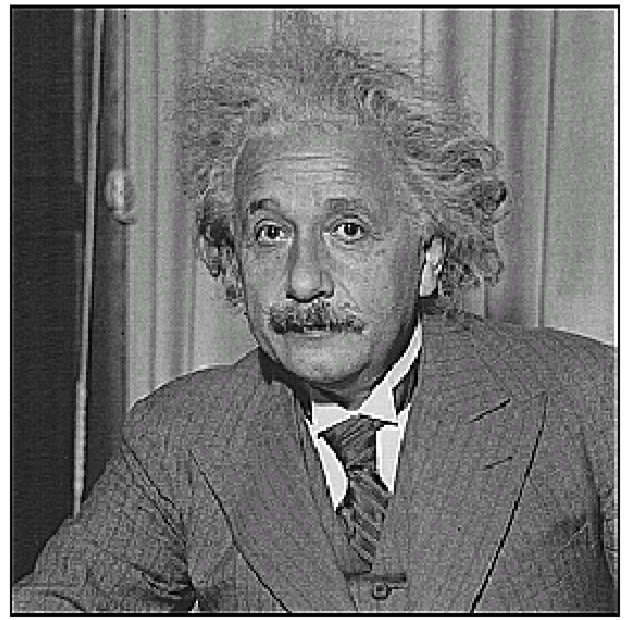


Sharpened
(differences are accentuated; constant areas are left untouched).

Sharpening



before



after

Filtering to reduce noise

- Noise is what we're not interested in.
  - We'll discuss simple, low-level noise today: Light fluctuations; Sensor noise; Quantization effects; Finite precision
  - Not complex: shadows; extraneous objects.
- A pixel's neighborhood contains information about its intensity.
- Averaging noise reduces its effect.

Additive noise

- I = S + N. Noise doesn't depend on signal.
- We'll consider:

$$I_i = s_i + n_i \text{ with } E(n_i) = 0$$

$$s_i \text{ deterministic.}$$

$$n_i, n_j \text{ independent for } n_i \neq n_j$$

$$n_i, n_j \text{ identically distributed}$$

Average Filter

- Mask with positive entries, that sum 1.
- Replaces each pixel with an average of its neighborhood.
- If all weights are equal, it is called a BOX filter.

F

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

(Camps)

Averaging Filter and noise reduction

- Example: try executing:

*k=2; figure(1); hist(sum((1/k)*rand(k,1000)))*

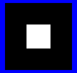
for different values of *k.*

- The average of noise is smaller than one example.
  - This is intuitive
  - Can be proven in many cases (some technical conditions: noise must be independent, many samples….)
  - Actually true for many real examples: Gaussian noise, flipping a coin many times

Filtering reduces noise if signal stable

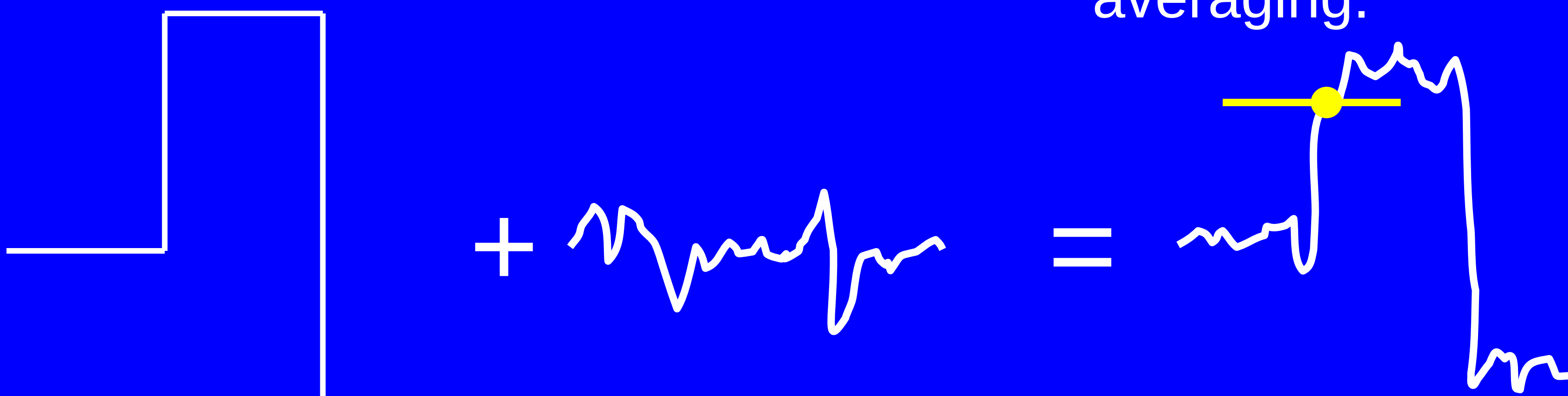
- Suppose I(i) = I+n(i), I(i+1) = I+n(i+1) I(i+2) = I+n(i+2).
- Average of I(i), I(i+1), I(i+2) = I + average of n(i), n(i+1), n(i+2).
- When there is no noise, averaging smooths the signal.
- So in real life, averaging does both.

Example: Smoothing by Averaging



Smoothing as Inference About the Signal

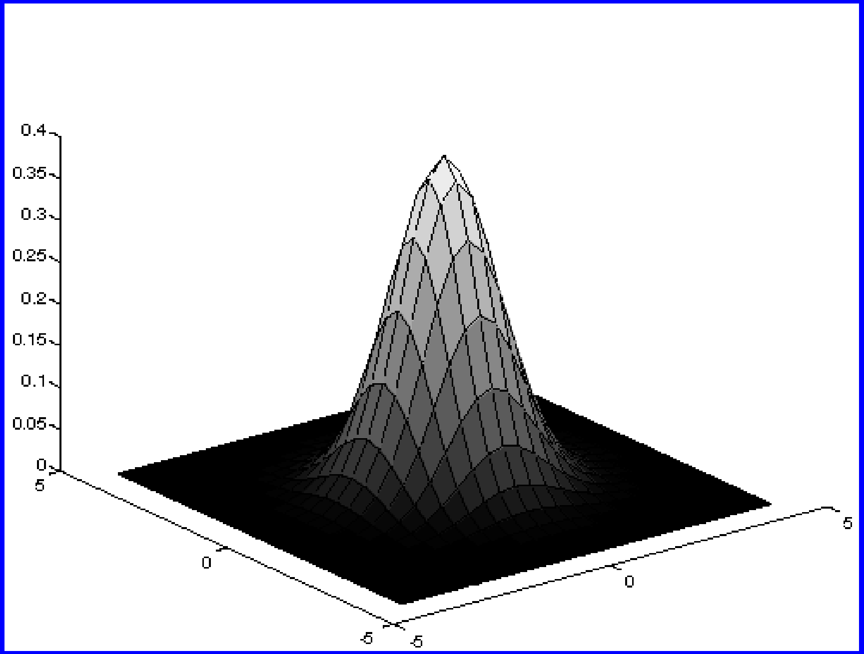
Neighborhood for averaging.



Nearby points tell more about the signal than distant ones.

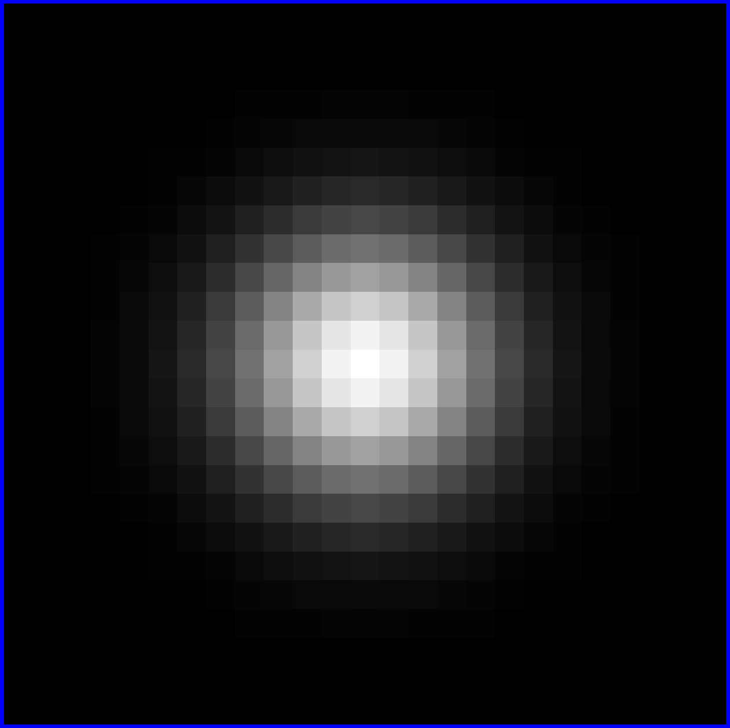
Gaussian Averaging

- Rotationally symmetric.
- Weights nearby pixels more than distant ones.
  - This makes sense as probabalistic inference.



- A Gaussian gives a good model of a fuzzy blob

An Isotropic Gaussian

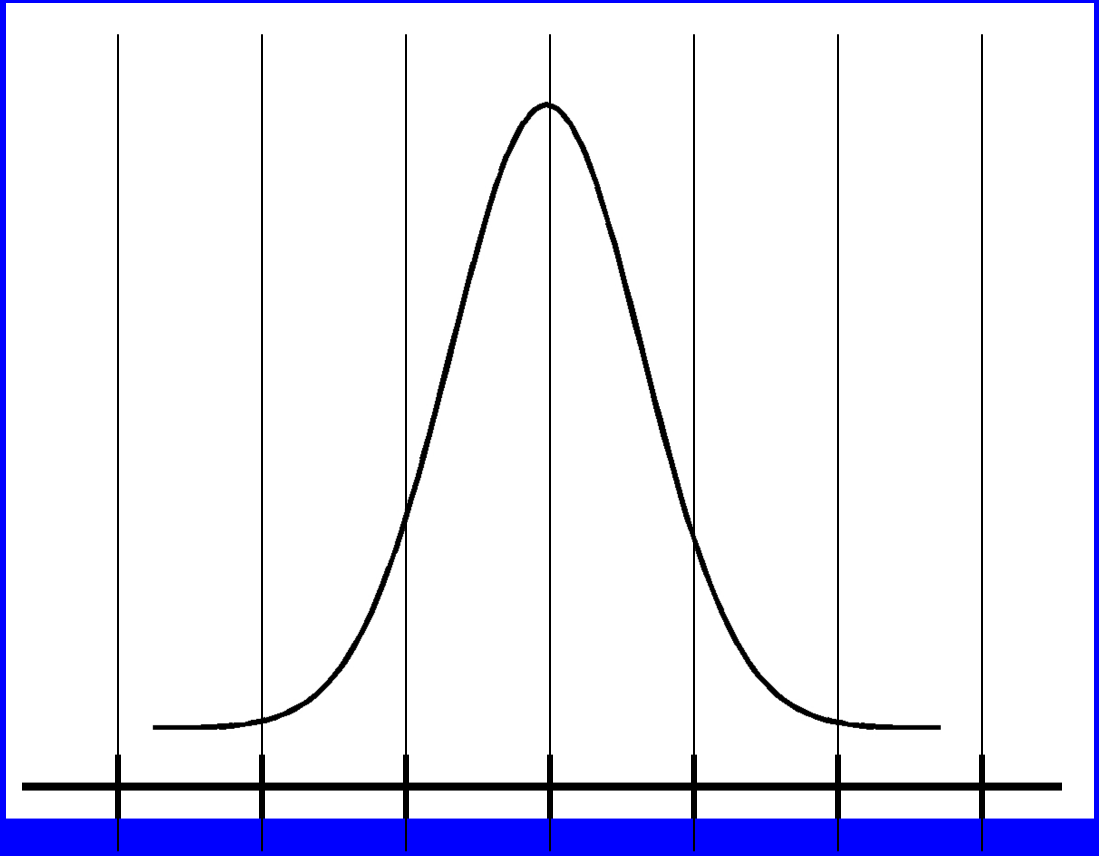


- The picture shows a smoothing kernel proportional to

$$G_0(x, y) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\left(x^2 + y^2\right) \Big/ 2\sigma^2 \right)$$

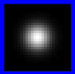
(which is a reasonable model of a circularly symmetric fuzzy blob)

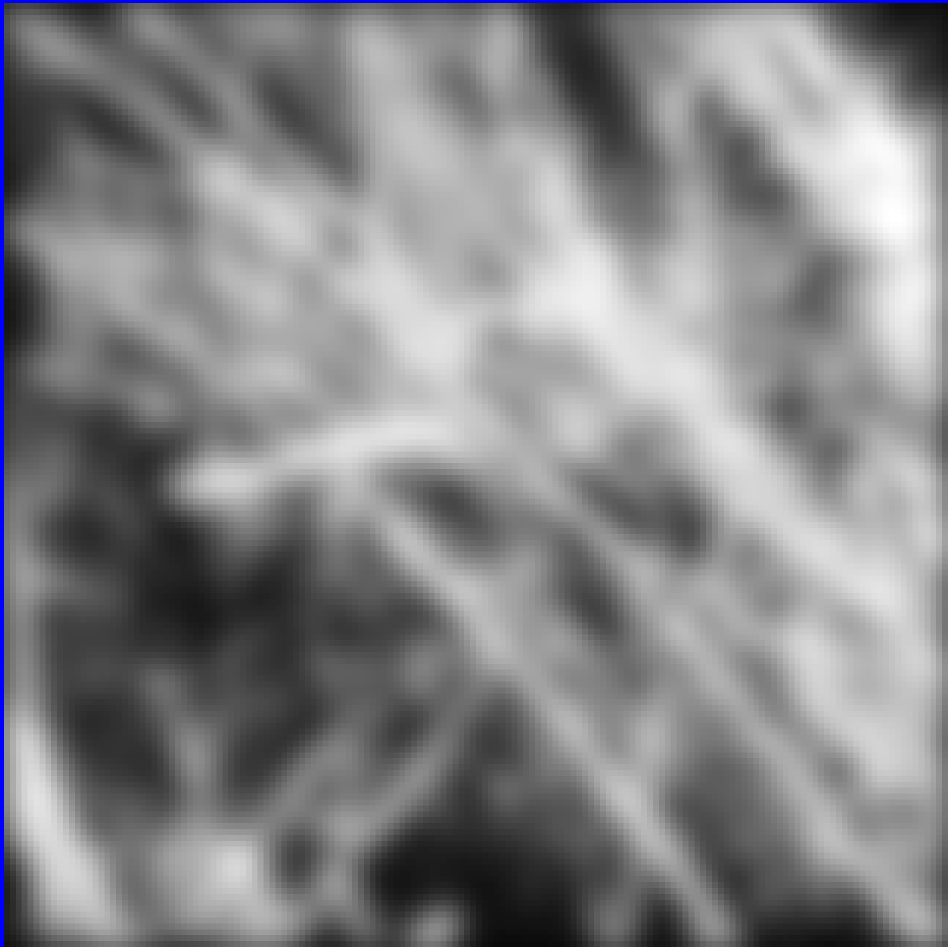
Building a Filter from a Continuous Function

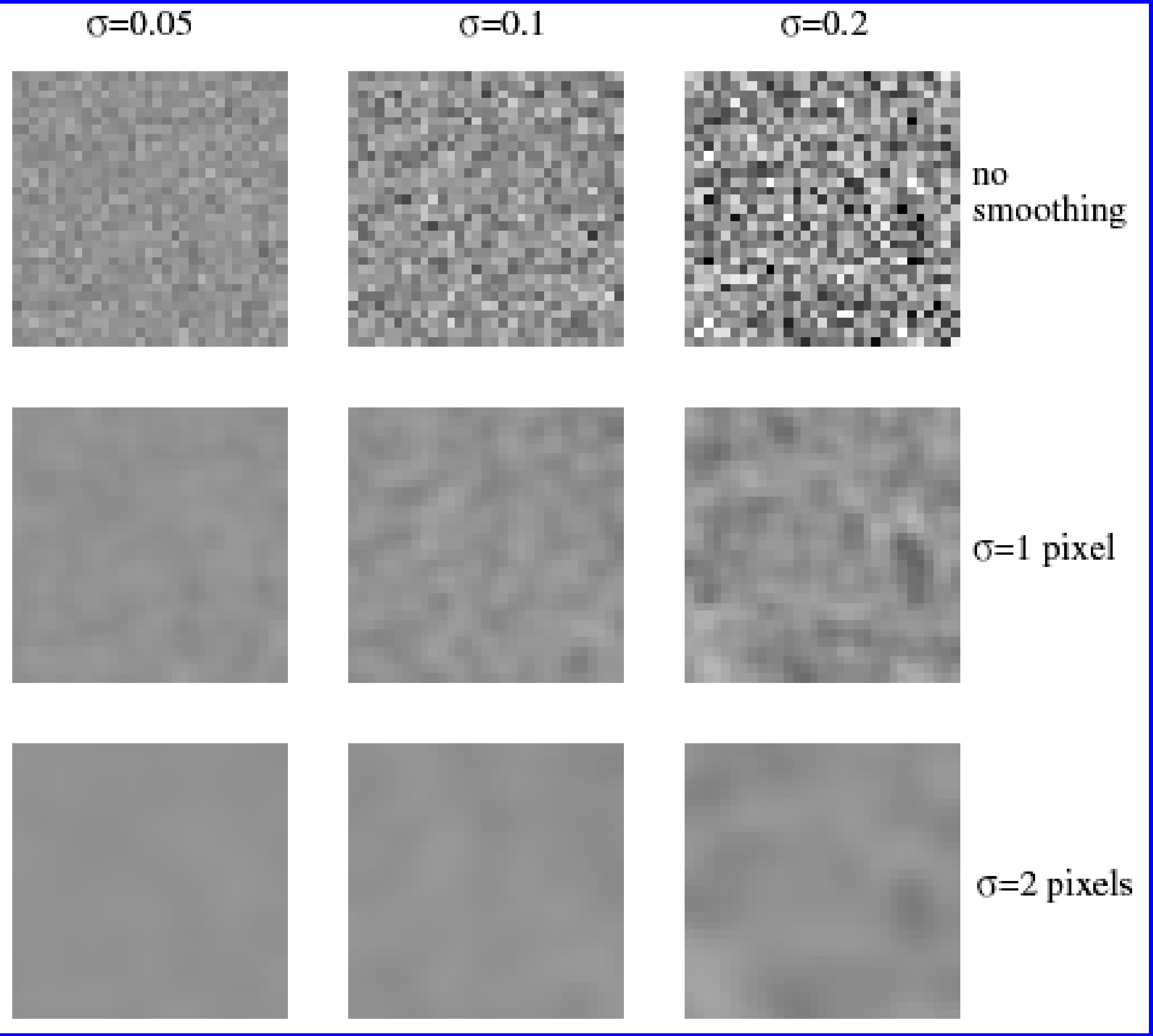


- Take a function
- Sample at integer positions.
- Sample values significantly more than zero.
- Normalize values
  - With averaging, you want to be sure that elements of filter sum to one.

Smoothing with a Gaussian







The effects of smoothing
Each row shows smoothing with gaussians of different width; each column shows different realizations of an image of gaussian noise.

(Forsyth and Ponce)

Efficient Implementation

- Both, the BOX filter and the Gaussian filter are separable:
  - First convolve each row with a 1D filter
  - Then convolve each column with a 1D filter.

Box Filter

$$\begin{bmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{bmatrix} = \begin{bmatrix} 0 & \frac{1}{3} & 0 \\ 0 & \frac{1}{3} & 0 \\ 0 & \frac{1}{3} & 0 \end{bmatrix} \circ \begin{bmatrix} 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ 0 & 0 & 0 \end{bmatrix}$$

Gaussian Filter

$$G_0(x,y) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\left(x^2 + y^2\right) \Big/ 2\sigma^2 \right) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-x^2 \Big/ 2\sigma^2 \right) \exp\left(-y^2 \Big/ 2\sigma^2 \right)$$

Smoothing as Inference About the Signal: Non-linear Filters.

What's the best neighborhood for inference?