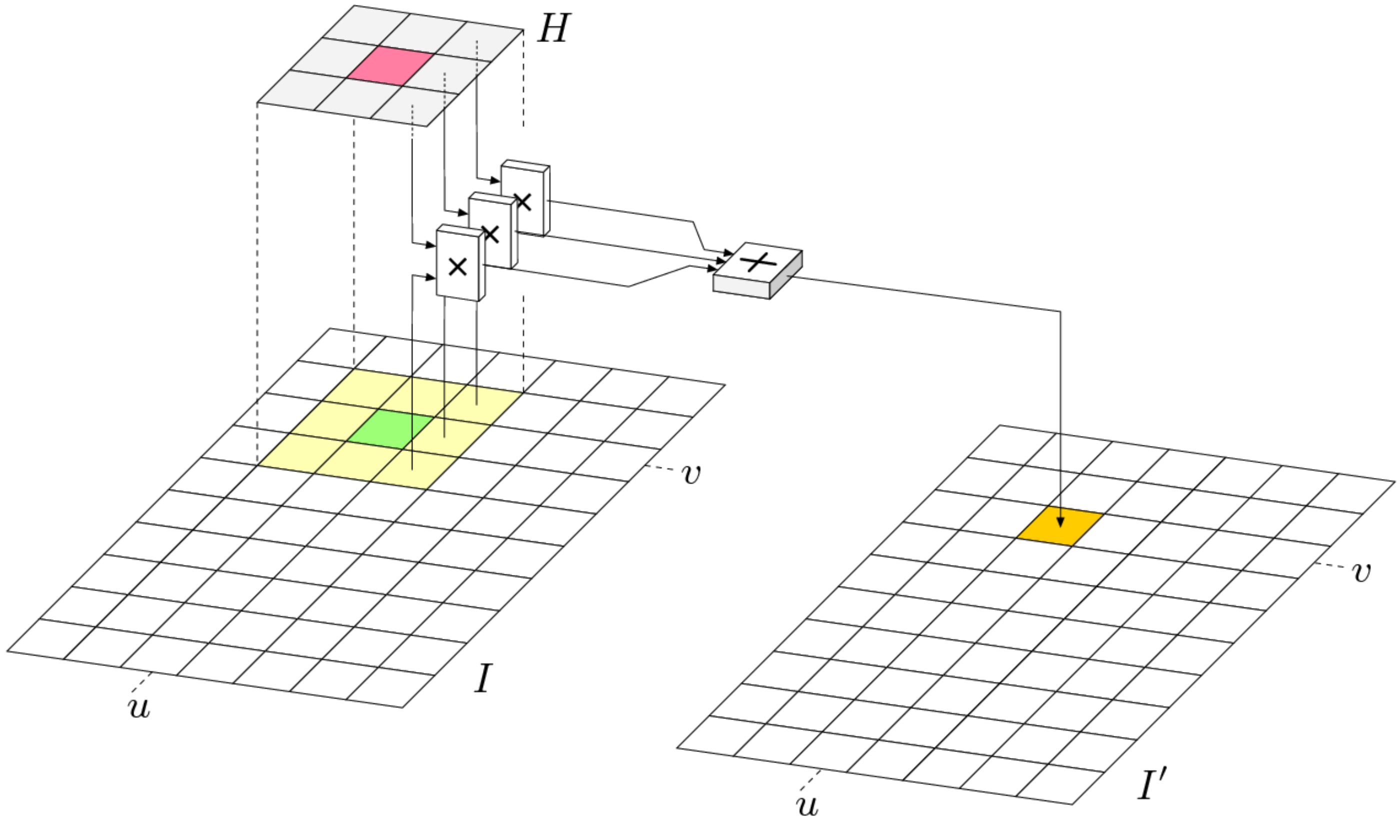
INF250

3 Filter

Filter or point operation

- The main difference between filter and point operation
  - Filter: Computations involve several pixels from the source
  - Point operations: Computations involve only one point (pixel) from the source
  - To blur (unsharpen) or sharpen an image you need to use a filter operation

Filtering



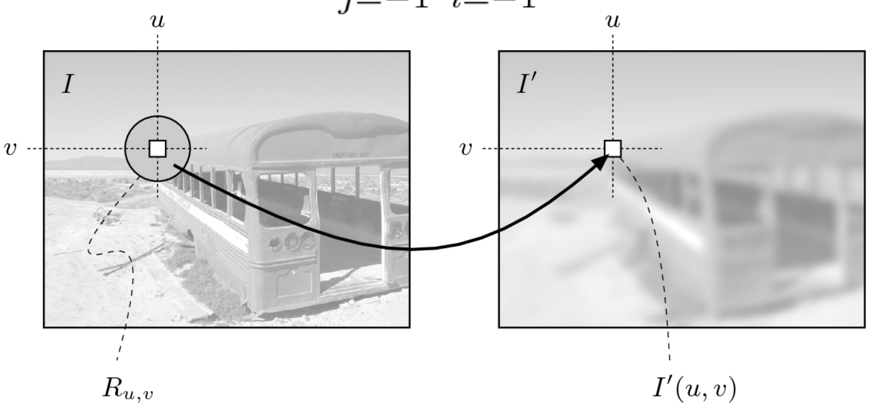
Smoothing

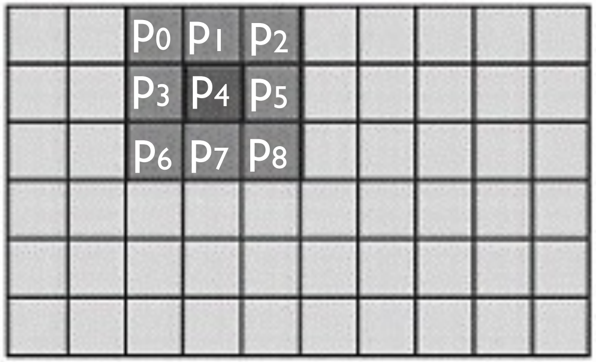
MEAN filter

$$I'(u,v) \leftarrow \frac{p_0 + p_1 + p_2 + p_3 + p_4 + p_5 + p_6 + p_7 + p_8}{9}$$

$$\begin{aligned} I'(u,v) \leftarrow \tfrac{1}{9} \cdot [\, & I(u-1,v-1) + I(u,v-1) + I(u+1,v-1) + \\ & I(u-1,v) + I(u,v) + I(u+1,v) + \\ & I(u-1,v+1) + I(u,v+1) + I(u+1,v+1) \,] \end{aligned}$$

$$I'(u,v) \leftarrow \frac{1}{9} \cdot \sum_{j=-1}^{1} \sum_{i=-1}^{1} I(u+i, v+j)$$





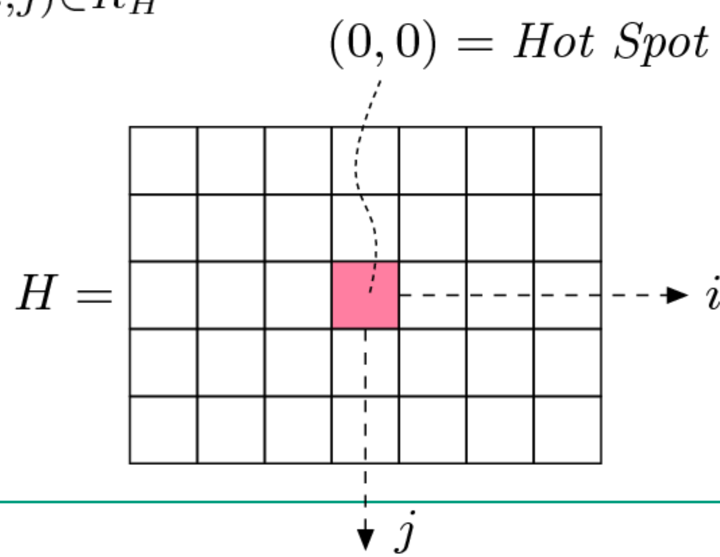
Smoothing filter

$$H(i,j) = \begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix} = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

filter operator

$$I'(u,v) \leftarrow \sum_{(i,j)\in R_H} I(u+i, v+j)\cdot H(i,j)$$

Convolution between filter and pixel value



Linear filters

Linear filters combine pixel values in a linear way, i.e., as a weighted summation. The results from a linear filter is completely specifed by the coefficients of the filter matrix.

The following steps are performed:

1. The filter matrix H is moved over the original image I such that its origin $H(0,0)$ coincides with the current image position (u,v)
2. All filter coefficients $H(i,j)$ are multiplied with the corresponding image element $I(u+I,v+j)$ and the results are added
3. The resulting sum is stored at the current position in the new image $I'(u,v)$

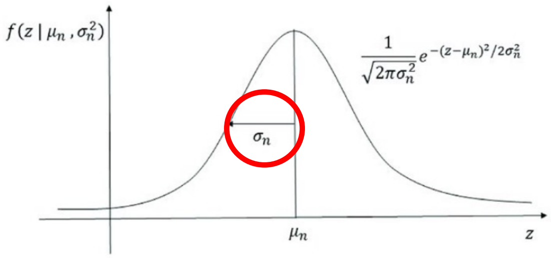
Linear convolution

$$I'(u,v) \leftarrow \sum_{i=-1}^{i=1} \sum_{j=-1}^{j=1} I(u+i, v+j) \cdot H(i,j)$$

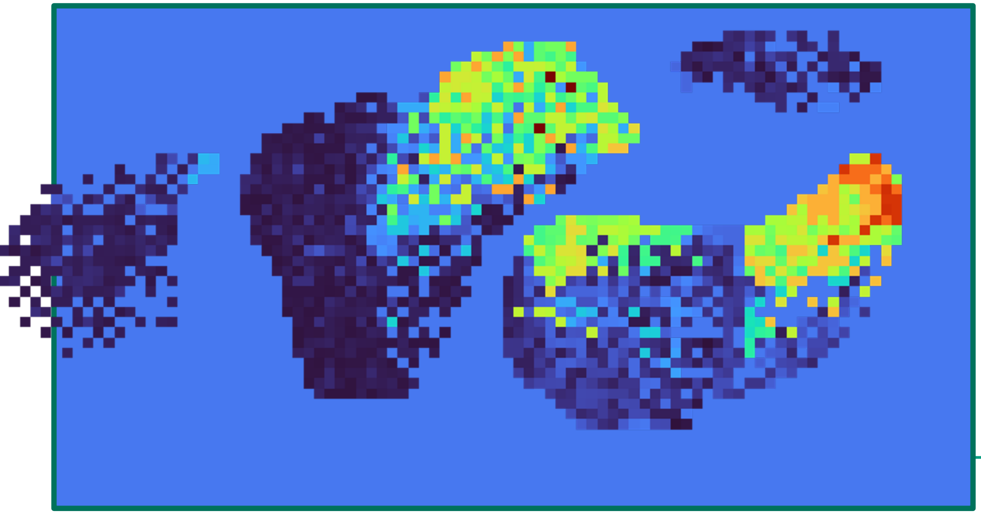
Clown example …

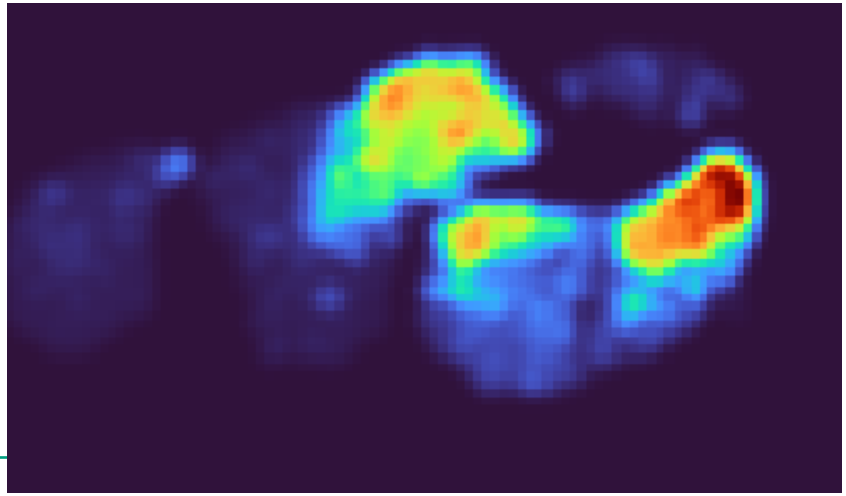
Convolution

Gaussian blur



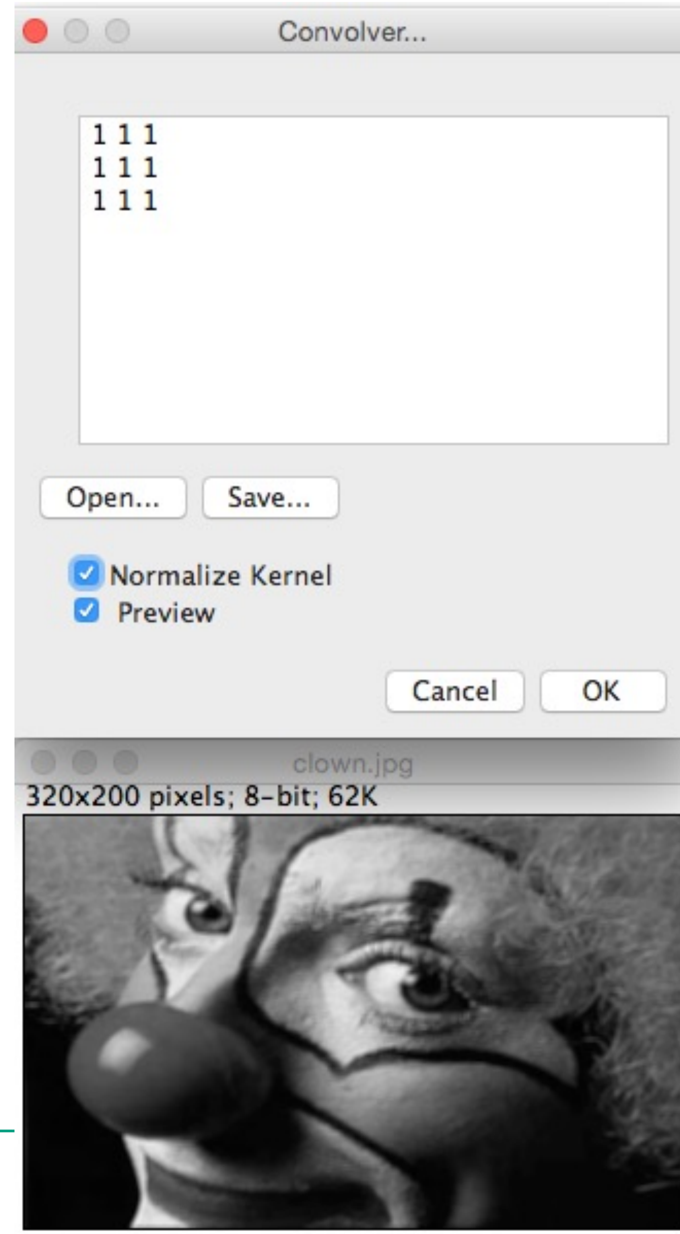
…





Smoothing in ImageJ

- Process/Filters/Convolve
- Edit filter
- Use Normalize Kernel
- Preview shows the result
- Save the filter



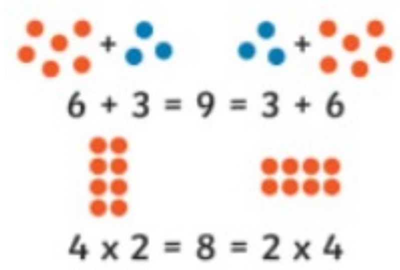
Properties for linear convolution

- Commutativity
- Linearity
- Associativity
- x/y separability

Commutativity

Linear convolution is commutative, i.e.,

$$I*H = H*I$$



Linearity

$$(s \cdot I) * H = s \cdot (\mathrm{I} * \mathrm{H})$$

$$(I_1 + I_2) * H = (I_1 * \mathrm{H}) + (I_2 * \mathrm{H})$$

$$(b+I) * H \neq \mathrm{b} + (I * H)$$

Assosiativity

The order of the successive filter operations is irrelevant

$$A * (\mathrm{B} * \mathrm{C}) = (A * B) * C$$

Separability

If $$H = H_1 * H_2 * \ldots \mathrm{H_n}$$

then

$$I * H = I * (H_1 * H_2 * \ldots \mathrm{H_n})$$

X/Y Separability

$$H_x = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$$

$$H_y = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

$$(I * H_x) * H_y = I * (H_x * H_y)$$

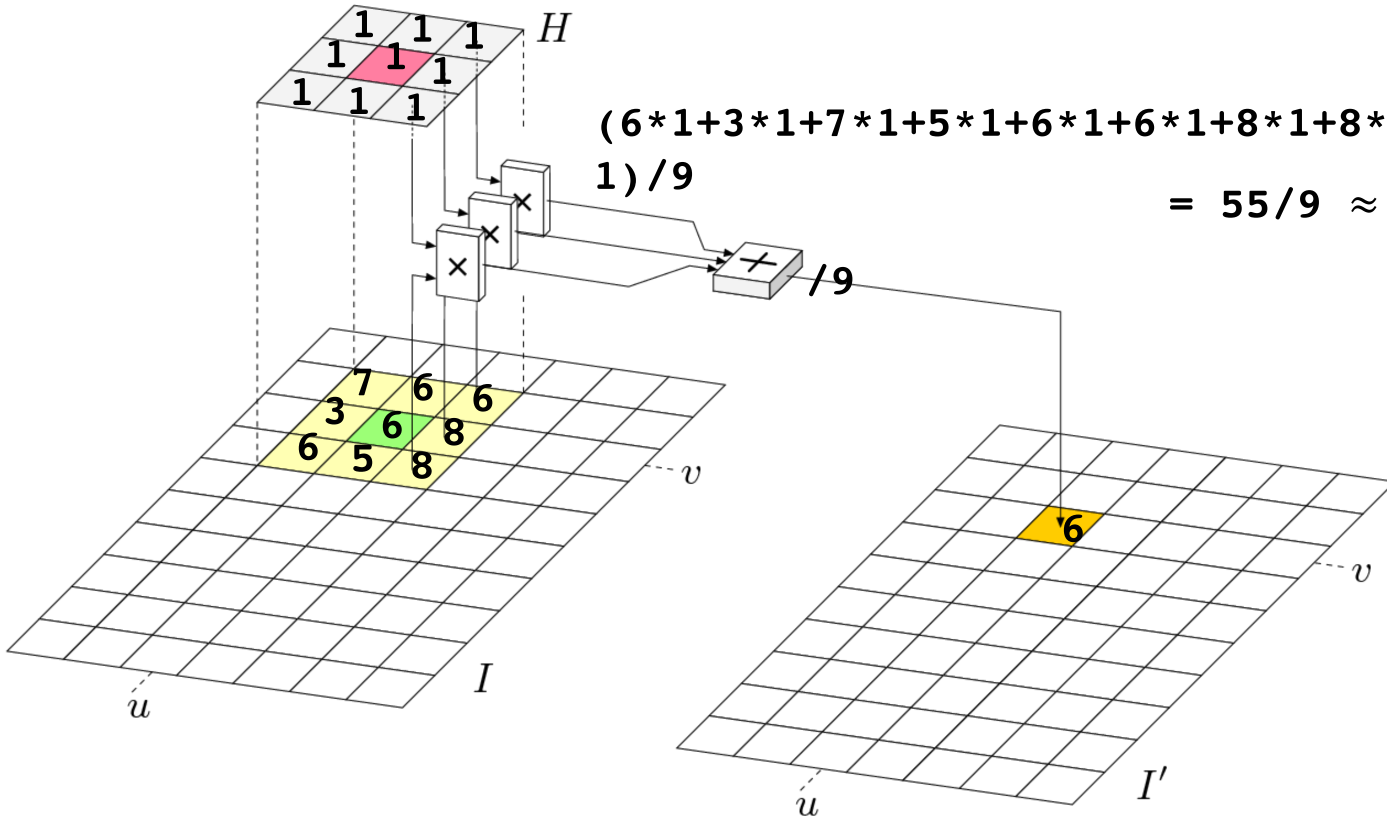
Smoothing mean filter

$$H_x = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$$

$$H_y = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

$$H_{xy} = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Smoothing mean filter



(6*1+3*1+7*1+5*1+6*1+6*1+8*1+8*1+6*1)/9

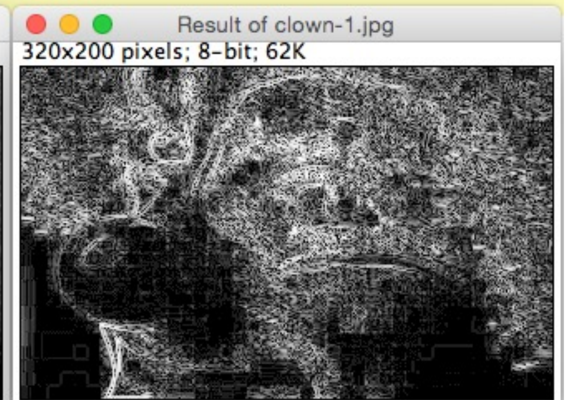
= 55/9 ≈ 6

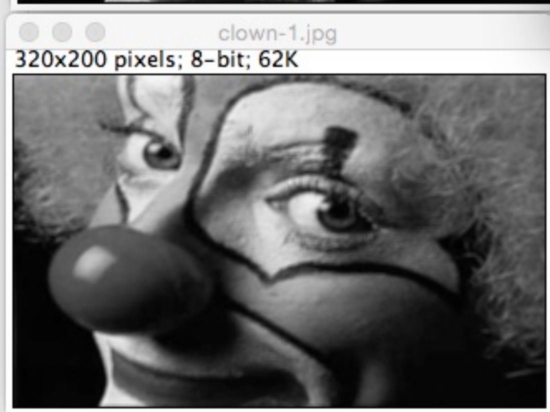
Analysing the residual image

- The residual image is the difference between the original image and the filtered image
- There can be a lot of information in a residual image

Smooting and residual

```
// Macro som utfører glatting og finner residualbildet
// Residual = Original - Glattet (difference)
run("Clown (14K)");
run("8-bit");selectWindow("clown.jpg");
run("Duplicate...", " ");
run("Convolve...", "text1=[1 1 1\n1 1 1\n1 1 1\n] normalize");
selectWindow("clown-1.jpg");
imageCalculator("Difference create", "clown-1.jpg","clown.jpg");
selectWindow("Result of clown-1.jpg");
run("Enhance Contrast...", "saturated=0.4 equalize");
```

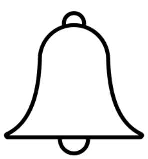




Other smoothing filters

- An alternative is to give the elements in H(i,j) a bell form
- Focus is given to the pixels close the center of the filter (hot spot)

$$H(i,j) = \begin{bmatrix} 0.075 & 0.125 & 0.075 \\ 0.125 & \mathbf{0.2} & 0.125 \\ 0.075 & 0.125 & 0.075 \end{bmatrix}$$



Integer coefficients

- In stead of using decimal numbers as coefficients in the filter matrix it is common to combine a new filter matrix with integer numbers and a constant

$$H(i,j) = s \cdot H'(i,j)$$

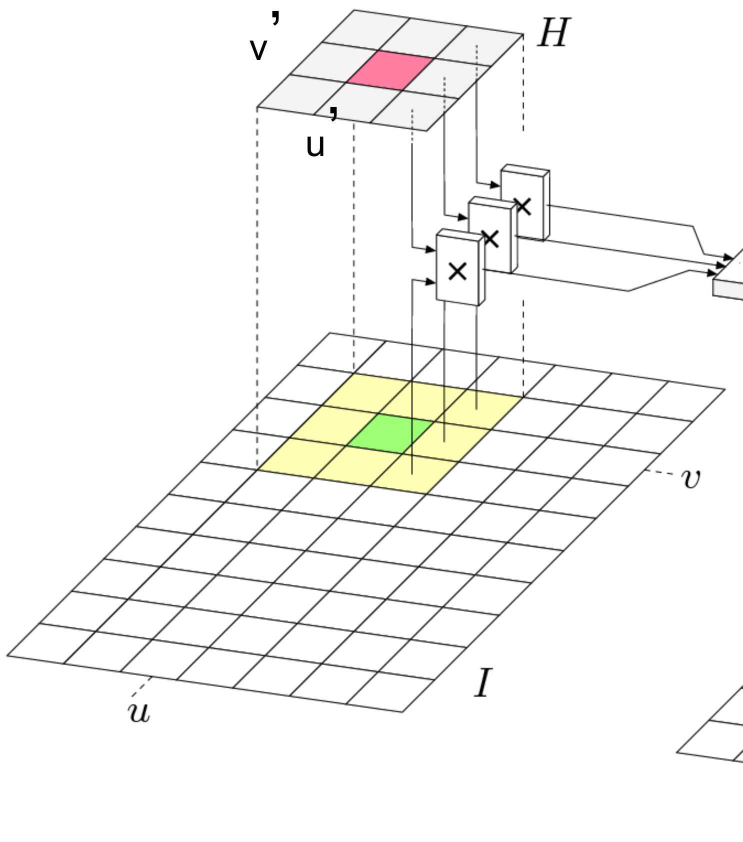
$$s = \frac{1}{\sum_{i,j} H'(i,j)}$$

$$H(i,j) = \begin{bmatrix} 0.075 & 0.125 & 0.075 \\ 0.125 & \underline{0.200} & 0.125 \\ 0.075 & 0.125 & 0.075 \end{bmatrix} = \frac{1}{40} \begin{bmatrix} 3 & 5 & 3 \\ 5 & \underline{8} & 5 \\ 3 & 5 & 3 \end{bmatrix}$$

Nonlinear filters

- Nonlinear filters also compute the result at some image position from the pixels inside a moving region of the original image.
- Nonlinear filters are combined by some nonlinear function
- Examples : maximum, minimum filters

Nonlinear filters

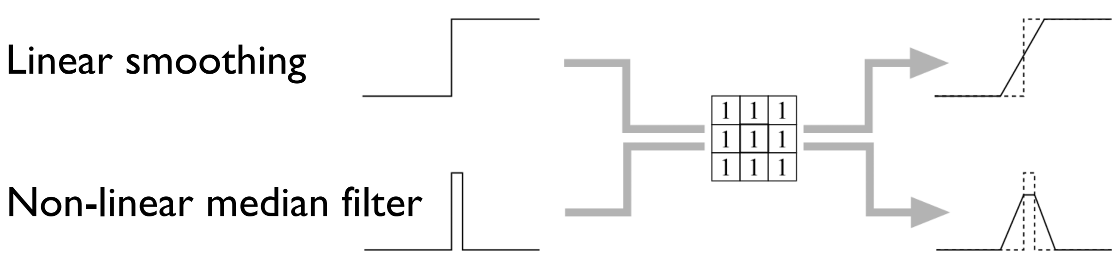


I'(u,v)=f_H(u',v')

Nonlinear filters and noise reduction

- Linear smoothing filters have the inconvenience of smoothing edges and lines, reducing the image quality
- Nonlinear filters can be used as a better solution

- Mean vs median filter applied to noisy image -> ImageJ



Noise reduction

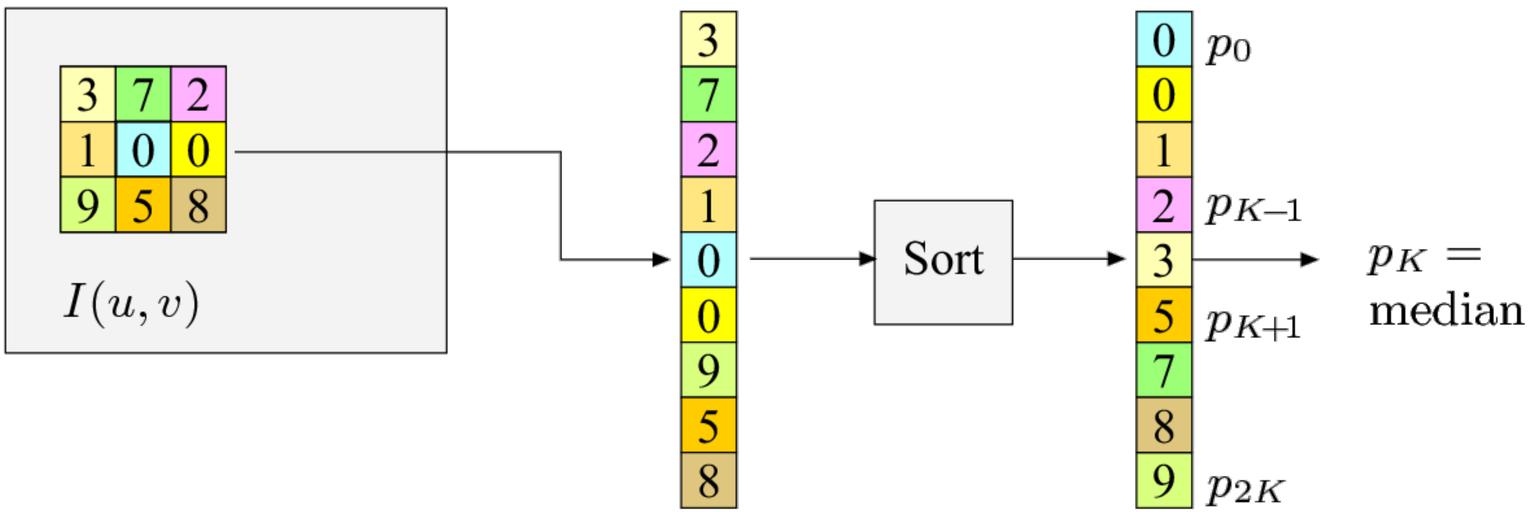
(a) Original image

(b) Minimum filter

(c) Maximum filter

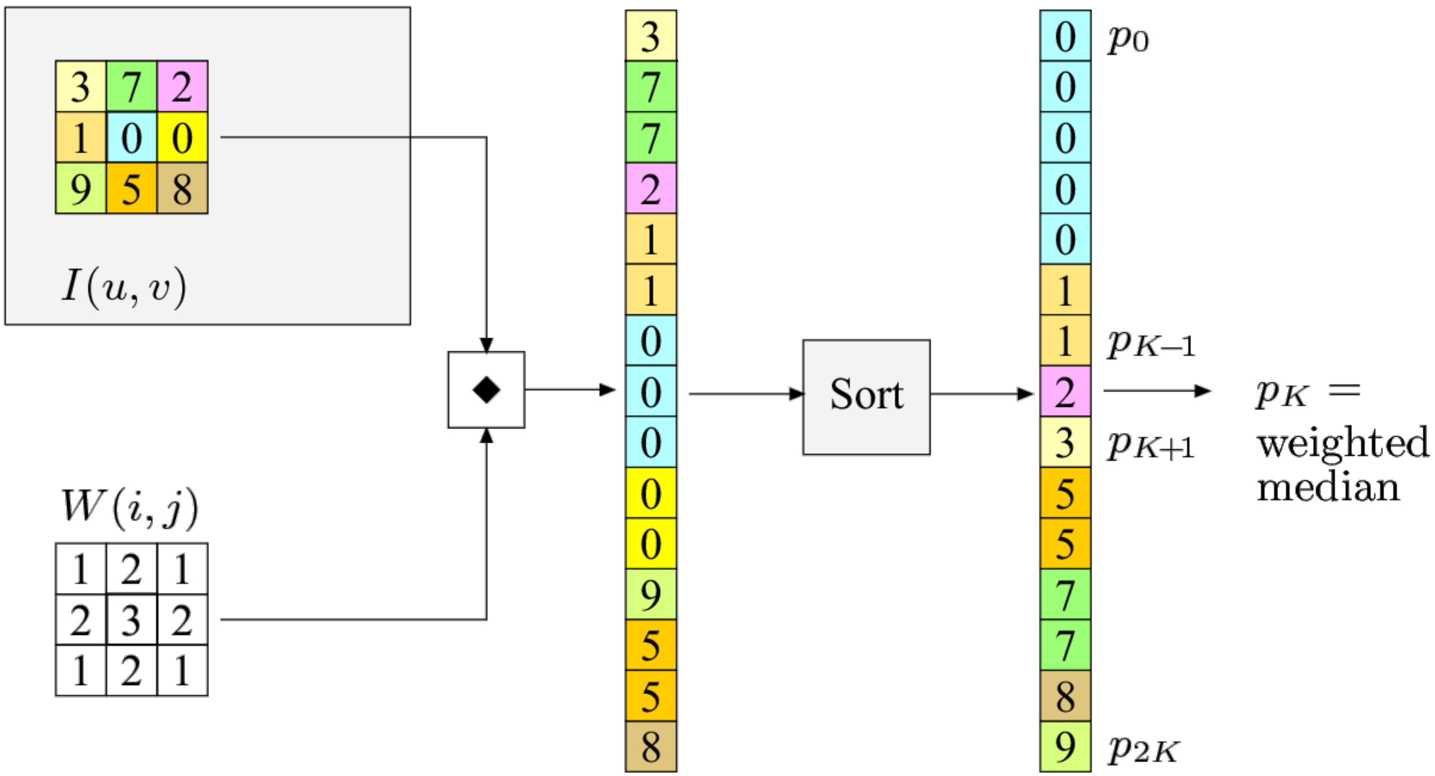
$$I'(u,v) \leftarrow \text{median}\left\{I(u+i, v+j) \mid (i,j) \in R\right\}$$

Median filter and noise reduction

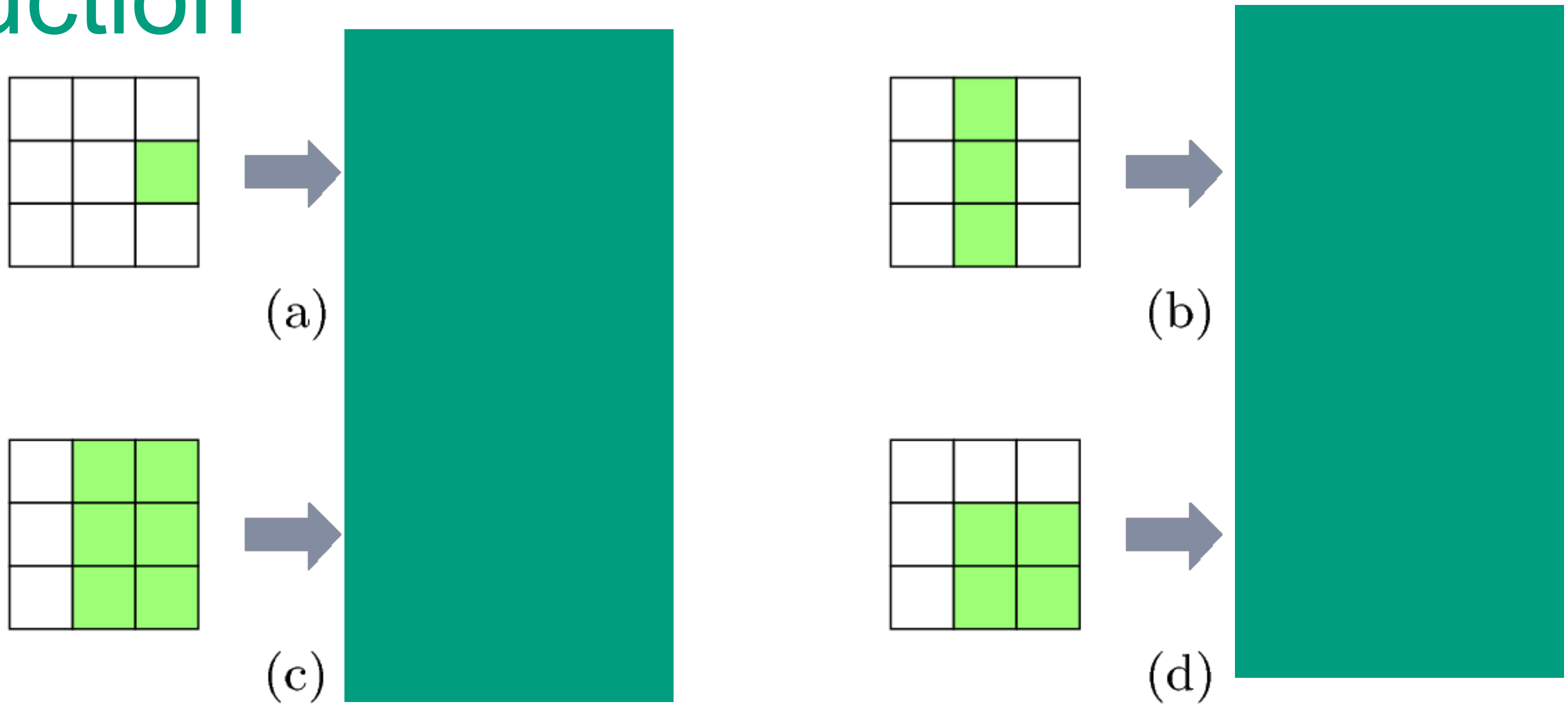


$$I'(u,v) \leftarrow \text{median}\,\{I(u+i, v+j) \mid (i,j) \in R\}$$

Weighted median filter and noise reduction



Median filter and noise reduction



Median filter and noise reduction

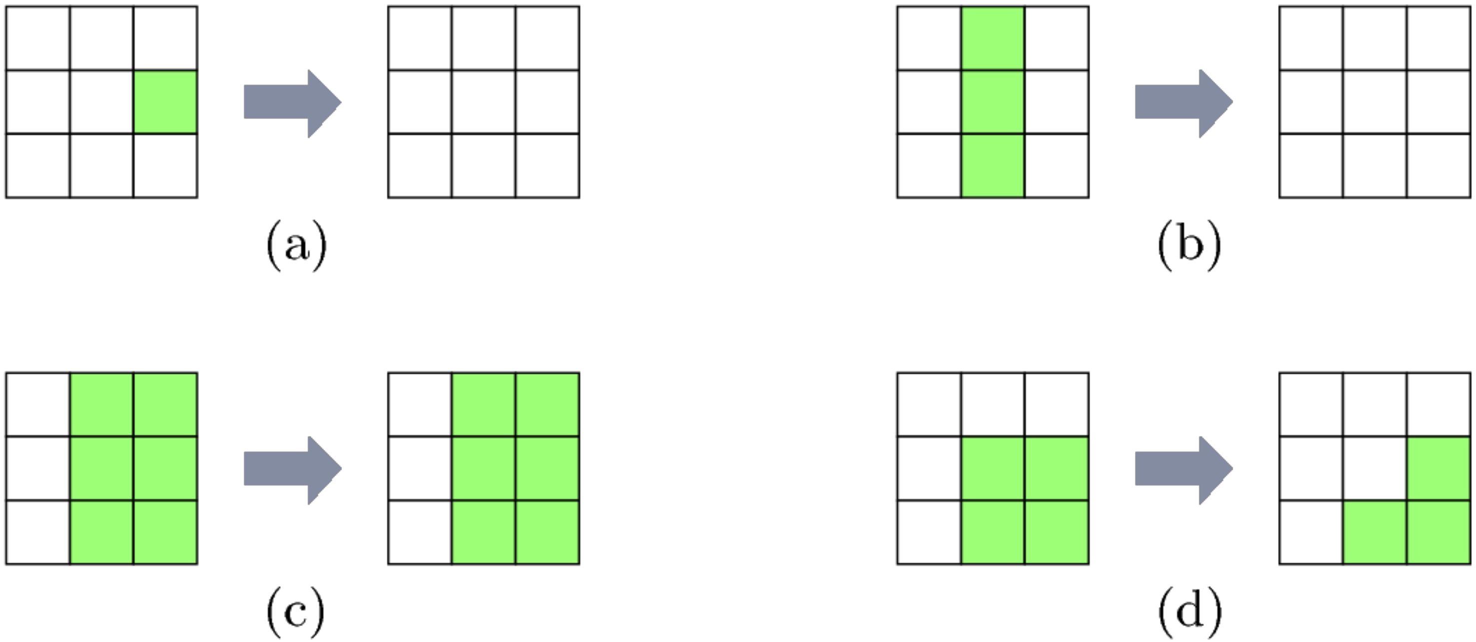


Figure 5.17 Effects of a 3×3 pixel median filter on two-dimensional image structures. Isolated dots are eliminated (a), as are thin lines (b). The step edge remains unchanged (c), while a corner is rounded off (d).

Borders

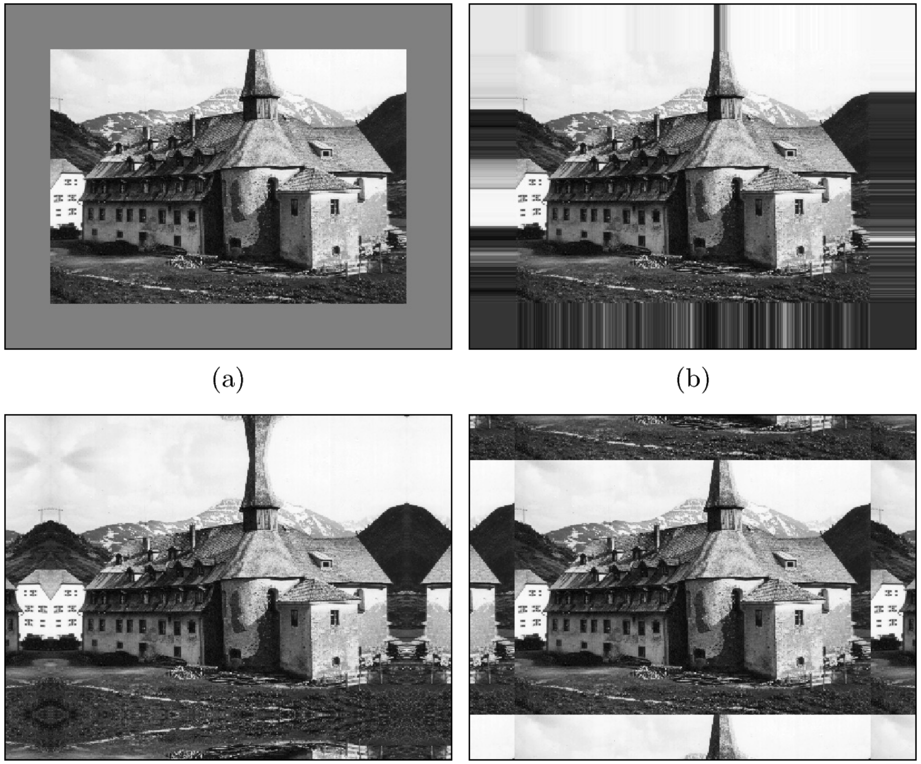
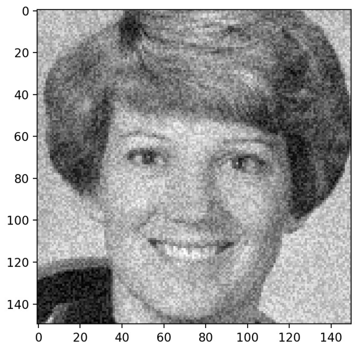
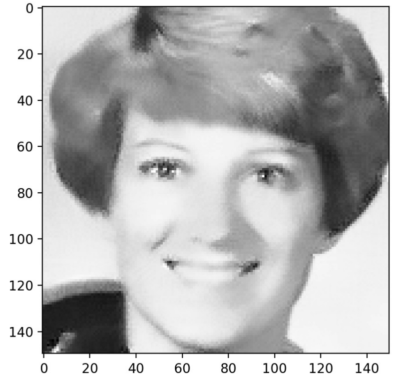


Figure 5.20 Methods for padding the image to facilitate filtering along the borders. The assumption is that the (nonexisting) pixels outside the original image are either set to some constant value (a), take on the value of the closest border pixel (b), are mirrored at the image boundaries (c), or repeat periodically along the coordinate axes (d).

Non local means

Better at preserving textures






```python
import matplotlib.pyplot as plt
import numpy as np
from skimage.filters import median
from skimage.filters import gaussian
from skimage.morphology import disk
from skimage import data, img_as_float
from skimage.color import rgb2gray

from skimage.restoration import denoise_nl_means


astro1 = img_as_float(data.astronaut())
astro = rgb2gray(astro1)

astro = astro[30:180, 150:300]
plt.imshow(astro, 'gray')

noisy = astro + 0.3*np.random.random(astro.shape)
plt.imshow(noisy,'gray')
noisy = np.clip(noisy, 0, 1)

medima = median(noisy,disk(2))
plt.imshow(medima,'gray')

gaussima = gaussian(noisy, sigma=2)
plt.imshow(gaussima, 'gray')

denoise = denoise_nl_means(noisy, 7, 9, 0.08)
plt.imshow(denoise, 'gray')

difference = astro-denoise
plt.imshow(difference,'gray')
```

https://www.youtube.com/watch?v=9tUns4HYtcw

https://www.youtube.com/watch?v=k8hS6uTz-Uc