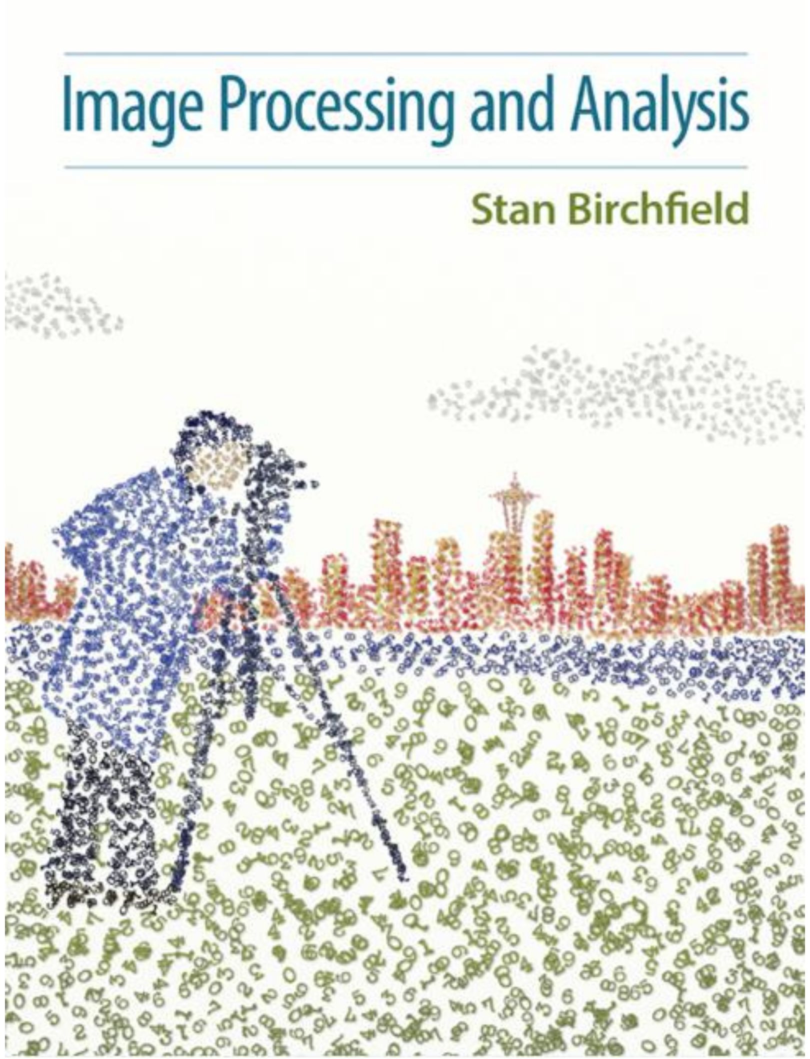
Prof. Kjersti Engan

University of Stavanger

ELE510 Image processing and computer vision

Nonlinear filtering (chap 5.5 Birchfield) 2023



Parts in course presentations is material from Cengage learning. It can be used for teaching, and it can be share it with students on access controlled web-sites (Canvas) for use in THIS course. **But it should not be copied or distributed further in any way** (by you or anybody).

Spatial domain filtering - Nonlinear filtering

Three points from the topic:

1. What do we seperate in linear and non-linear filtering?
2. Why can non-linear filtering be good in ex. Enhancement/denoising?
3. Give example on filters that both perserve edges and smooth other regions

Remember – linear shift-invariant filters

Filters so far: linear and shift-invariant filters. Convolution formula is valid. *w* and *h* are the width and height of the kernel.

$$I'(x,y) = I(x,y) \circledast G(x,y) = \sum_{i=0}^{w-1}\sum_{j=0}^{h-1} I(x+\tilde{w}-i, y+\tilde{h}-j)G(i,j)$$

Effect of mean /box filtering

3x3 5x5 7x7

Salt & Pepper noise

Gaussian noise

Side effect - blur

(5.5) Non-linear filtering

- filters that are not linear are called **non-linear filters**
- **Common example: Median filter -** replaces each pixel with the median of all the gray levels in a local neighborhood, generally defined by a square window.

Median filter

- Different type of noise needs different filters.
- We have seen that for **salt and pepper** noise the Gaussian, or flat mean (box filters) does not work well. These are both examples of linear low pass (LP) filters.
- For salt/pepper noise a median filter is a good alternative
- A *median filter* operates over a window by selecting the median intensity in the window.

Median filter – not linear

A *median filter* operates over a window by selecting the median intensity in the window.

Median filter is an example of non-linear filtering, and an example of rank-order filter

$$Median(f_1(x)+f_2(x)) \neq Median(f_1(x)) + Median(f_2(x))$$

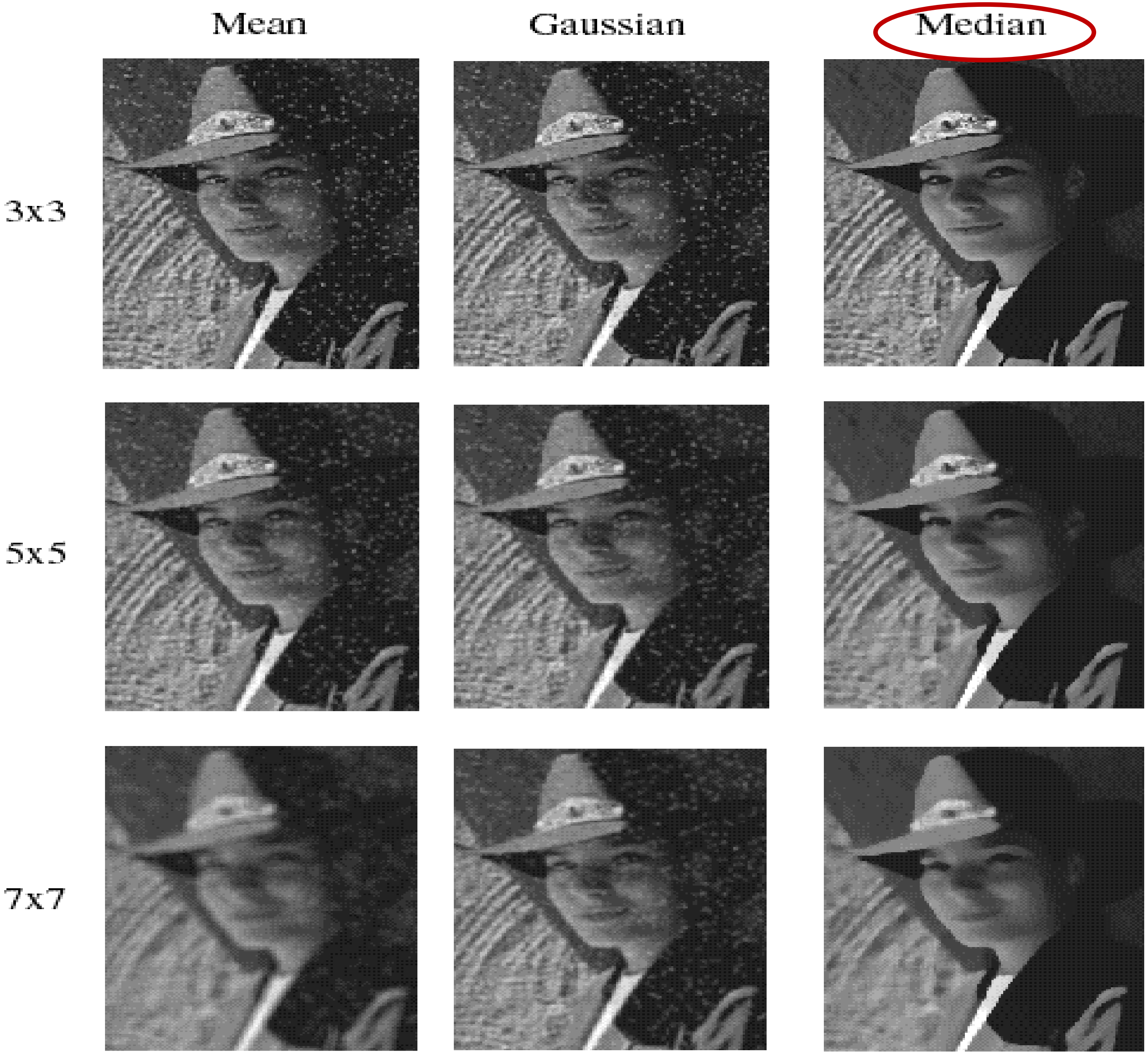
$$1 = Median(\begin{bmatrix} 1 & 1 & 1 \\ 1 & 3 & 1 \\ 1 & 1 & 1 \end{bmatrix}) = Median(\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 0 \end{bmatrix}) \neq$$

$$Median(\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}) + Median(\begin{bmatrix} 0 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 0 \end{bmatrix}) = 1 + 1 = 2$$

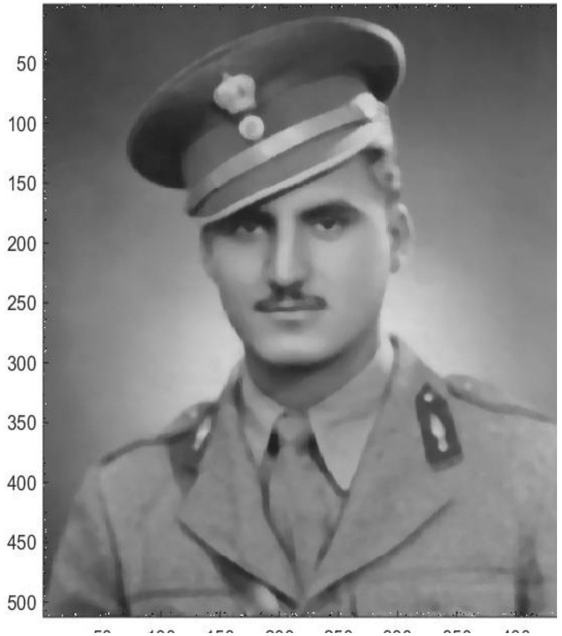
Median filter

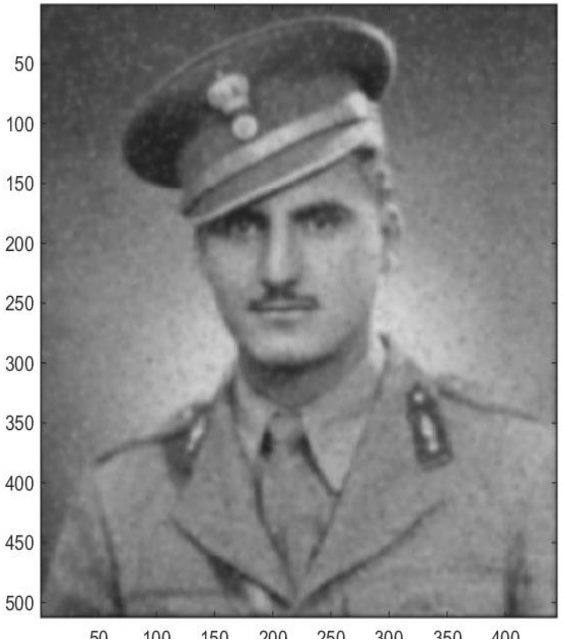
Median filter, salt end pepper noise. Mask 3x3, 5x5, 7x7

Salt & Pepper noise



Gaussian noise

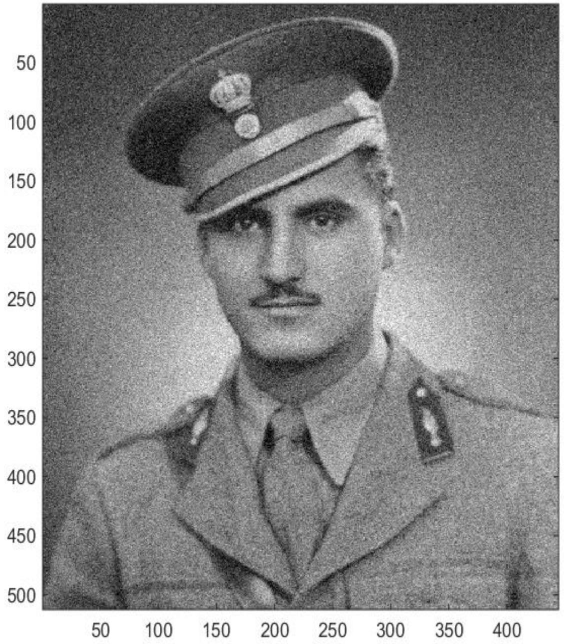


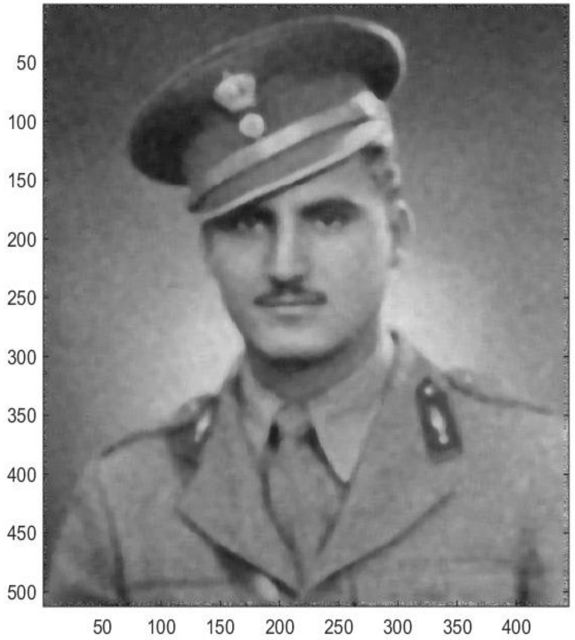


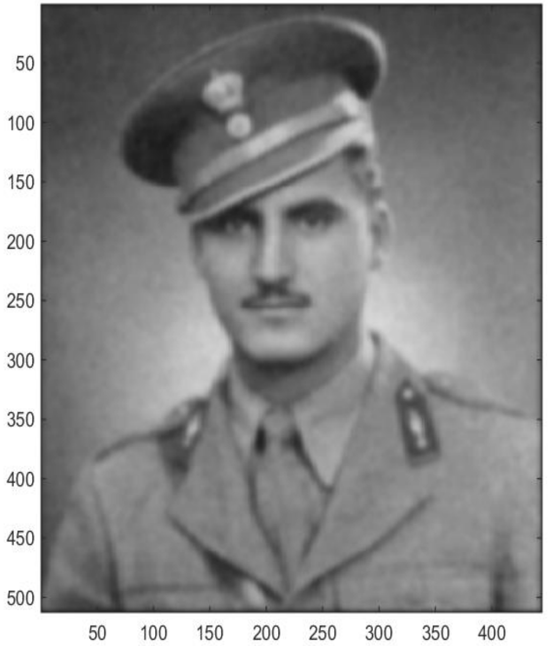
Salt and pepper noise,

median filter,

gaussian filter







Gaussian noise,

median filter,

gaussian filter

Mode filter

Example nonlinear filter - Non-local means

- **Non-local means (NLM)**: a particularly effective way to reduce the effects of noise in an image by computing a weighted average over *all* pixels in the image.
- Weight based on the similarity between the regions

ALGORITHM 5.12 Perform non-local means filtering on an image

```
NonLocalMeans(I, w)

Input:  grayscale image I, set of pixels W specifying window
Output: smoothed image from applying non-local means filtering

for (x, y) ∈ I' do                              ▷ For each pixel in the output image (same size as input image),
    val ← 0                                       initialize value to zero,
    norm ← 0                                      and normalization factor to zero.
    for (x', y') ∈ I do                          ▷ For each pixel in input image,
        d ← 0                                     initialize distance to zero.
        for (δx, δy) ∈ W do                      ▷ Compute the dissimilarity between the two windows.
            d ←+ I(x + δx, y + δy) − I(x' + δx, y' + δy)
        w ← exp(−d * d/(2 * σ * σ))              ▷ Set the weight to the similarity.
        val ←+ w * I(x', y')                     ▷ Update the sum of the weighted pixels,
        norm ←+ w                                 and the normalization factor.
    I'(x, y) ← val/norm                          ▷ Set the output pixel to the normalized value.
return I'
```

Non-local means

OpenCV, python

From documentation; https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_photo/py_non_local_means/py_non_local_means.html#non-local-means

OpenCV provides four variations of this technique.

1. **cv2.fastNlMeansDenoising()** - works with a single grayscale images
2. **cv2.fastNlMeansDenoisingColored()** - works with a color image.
3. **cv2.fastNlMeansDenoisingMulti()** - works with image sequence captured in short period of time (grayscale images)
4. **cv2.fastNlMeansDenoisingColoredMulti()** - same as above, but for color images.

- Common arguments are:h : parameter deciding filter strength . Higher h value removes noise better, but removes details of image also. (10 is ok)

- hForColorComponents : same as h, but for color images only. (normally same as h)

- templateWindowSize : should be odd. (recommended 7)
- searchWindowSize : should be odd. (recommended 21)

- Parctically dont look in entire image, just the searchWindow.

Example

Upper line:
Left: original
Right: with added gaussian noise

Lower line:
left:gaussian smoothing
Right: non-local means

Example nonlinear filter - Bilateral filtering

- Contains two kernels, a *spatial kernel* and a *range kernel*.
- The spatial kernel g_s weights neighboring samples according to their proximity to the central sample, while the range kernel g_r weights neighboring samples according to their similarity in *value* to the central sample.

$$f'(x) = f(x) \circledcirc \langle g_s(x), g_r(z) \rangle = \frac{1}{\eta(x)} \sum_i f(i) g_s(x-i) g_r(f(x) - f(i))$$

- Smoothing, but

preserves edges

- Repeated applications

gives a cartoon-like image

OpenCV, python:

dst=cv2.bilateralFilter(src, d, sigmaColor, sigmaSpace[, dst[, borderType]])

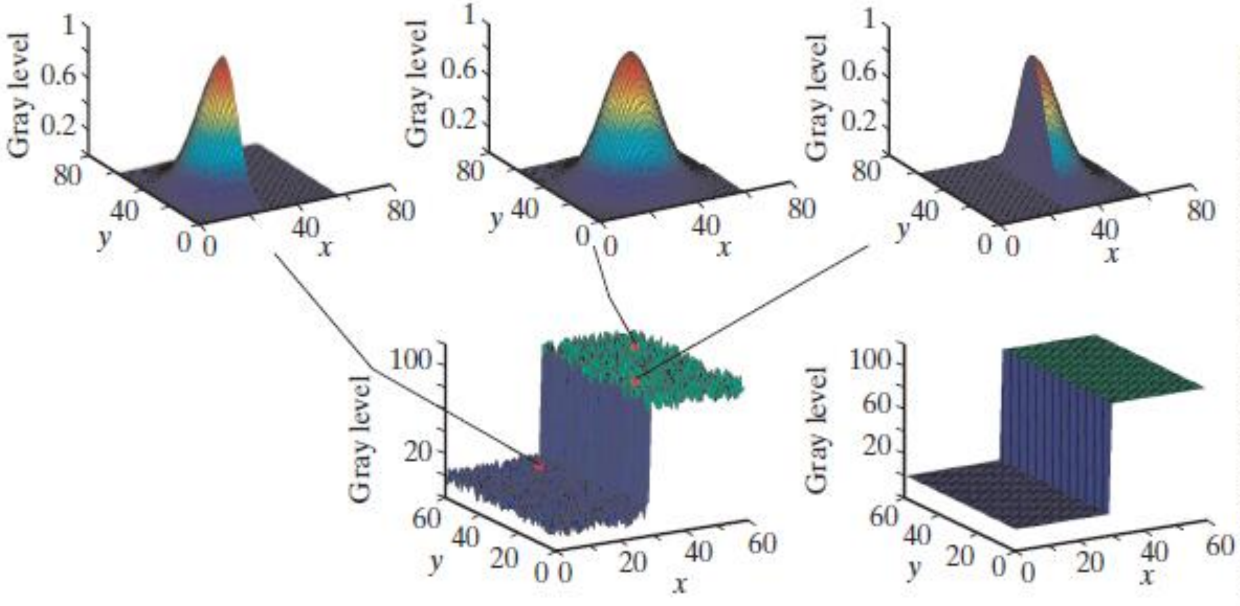


Figure 5.17 Bilateral filtering of a noisy step edge preserves the crisp edge as it smooths out the noise on either side of the edge. The top row shows the kernel at three locations: Far from the edge, the kernel approximates a Gaussian, whereas near the edge it approximates half a Gaussian.

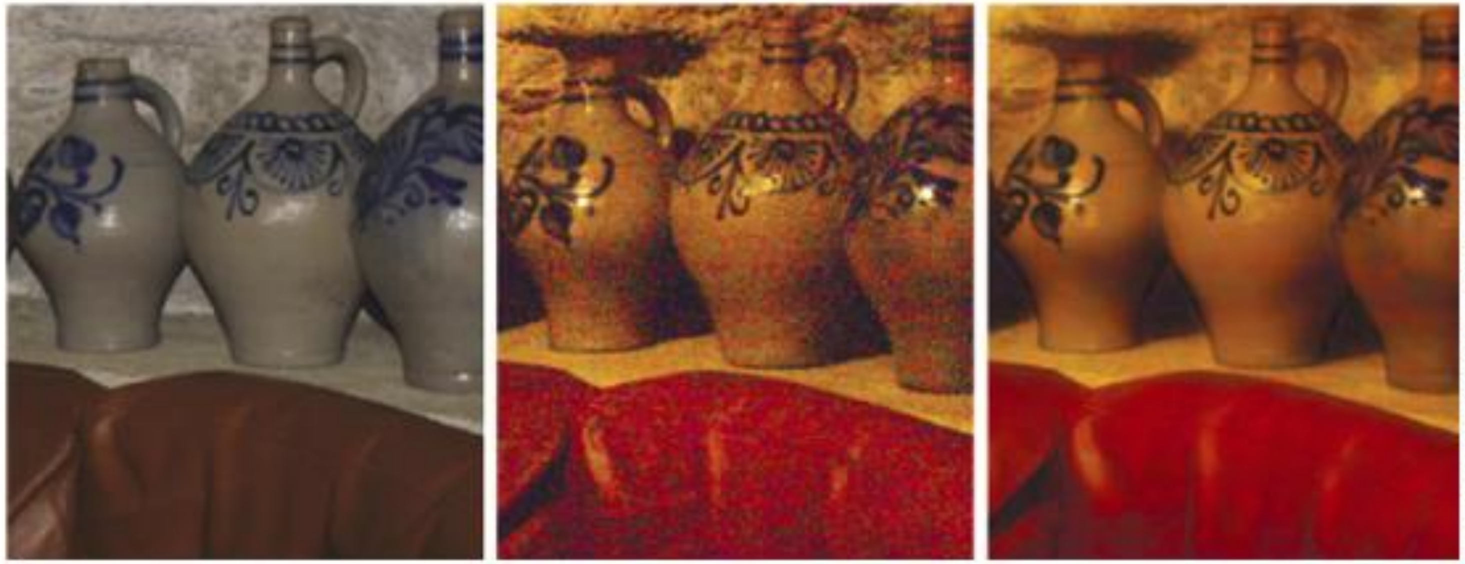


Figure 5.21 One of the more interesting applications of the bilateral filter is to combine a flash image (left) with a no-flash image (middle) to preserve both detail and the original viewing conditions of the scene (right).

Example

Upper line:
Left: original
Right: with added gaussian noise

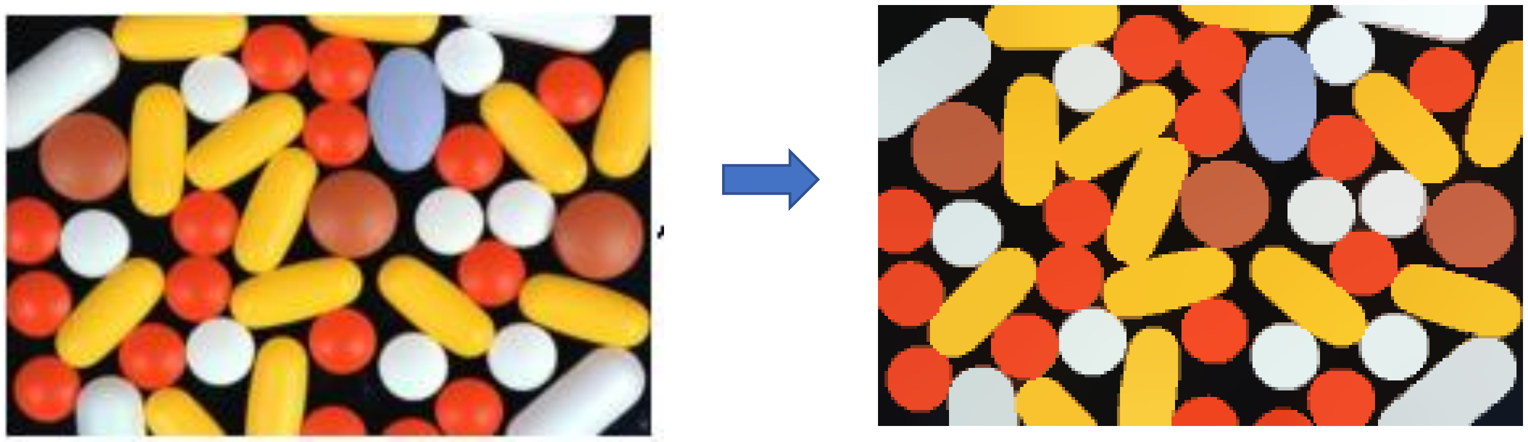
Lower line:
left:gaussian smoothing
Right: bilateral filtering

Other non-linear filters

- Anisotropic diffusion
  - The image values are smoothed by repeatdly performing local averages, but doing so in a way that weights neighbour pixels less if they lie on an intensity edge. Blurs the images within regions, but not across boundaries.
- Adaptive smoothing
  - Find weighted averages of neighbors of pixels, with weigths discourageing smoothing across boundaries. Bilateral filter is special case of adaptive smoothing.
- Gray-scale morphological operators

Other non-linear filters

- Mean-shift filter
  - Clustering both in color values and in distance. Group pixels near in color values AND distance, and recalcualte mean according to mean-shift. Iteratively replaces each pixel with the mean of the pixels in a range-r neighborhood and whose value is within a distance d until mean-shift =0.

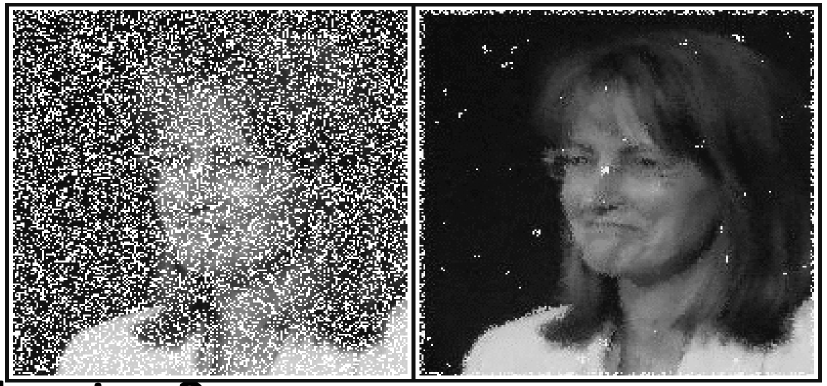


Example

Upper line:
Left: original
Right: with added gaussian noise

Lower line:
left:gaussian smoothing
Right: mean shift

Spatial domain filtering - Nonlinear filtering



Three points from the topic:

1. What do we seperate in linear and non-linear filtering?
   - ✓ Because linear filters have so many properties that make them tractable (convolution formula ..)

2. Why can non-linear filtering be good in ex. Enhancement/denoising?
   - ✓ Different noise needs different types of filters. Non-linear filters are less restrictive, and sometimes really good alternative

3. Give example on filters that both perserve edges and smooth other regions
   - ✓ Non-local means, bilateral filtering, mean-shift