Image preprocessing

Frequency analysis and filtering

Noise

- Degradation of image by random errors.
- Usually described by its probabilistic characteristic

Noise

White noise

- Has constant power spectrum.
- Models the worst approximation of degradation.
- Simplifies calculations.

Noise

- Additive noise

$$f(x,y) = g(x,y) + \nu(x,y)$$

- g – input image,
- ν – noise

Noise – original image

Noise – Gaussian white noise added

Noise

- Multiplicative noise

$$f(x,y) = g(x,y)\nu(x,y)$$

- g – input image,
- ν – noise
- models dependence of noise on the signal magnitude itself.

Noise

Impulse noise

- Image corrupted by individual noisy pixels whose brightness differs significantly from that of the neighbourhood.
- Salt-and-pepper noise – saturated impulse noise

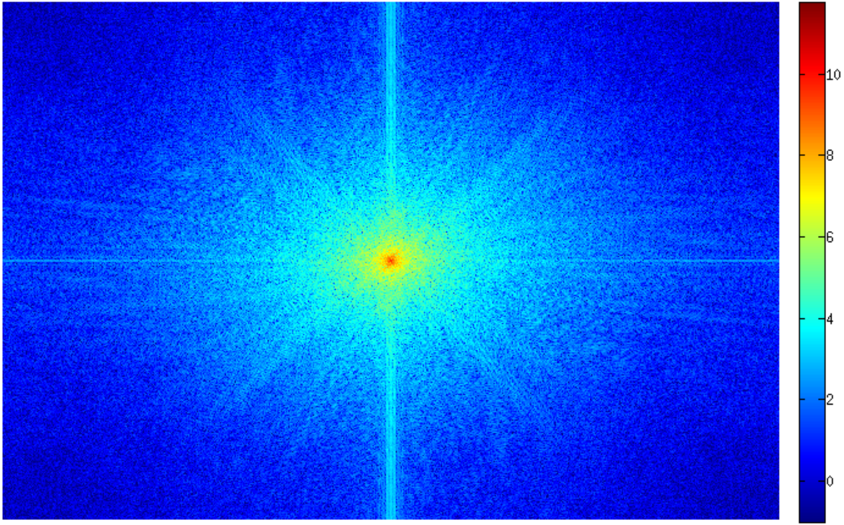
Noise – Salt-and-pepper

Filtering

- Use small neighbourhood of a pixel in an input image to produce a new brightness value on the output image
- Smoothing – suppress noise or other small fluctuations in the image
- Gradient operators – indicate locations where the image function undergoes rapid changes

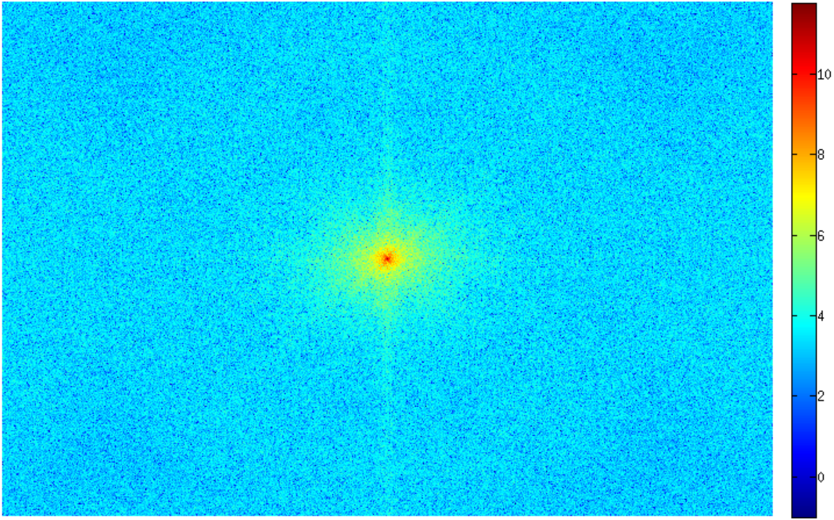
Filtering in frequency domain–original image

Filtering in frequency domain–original image spectrum

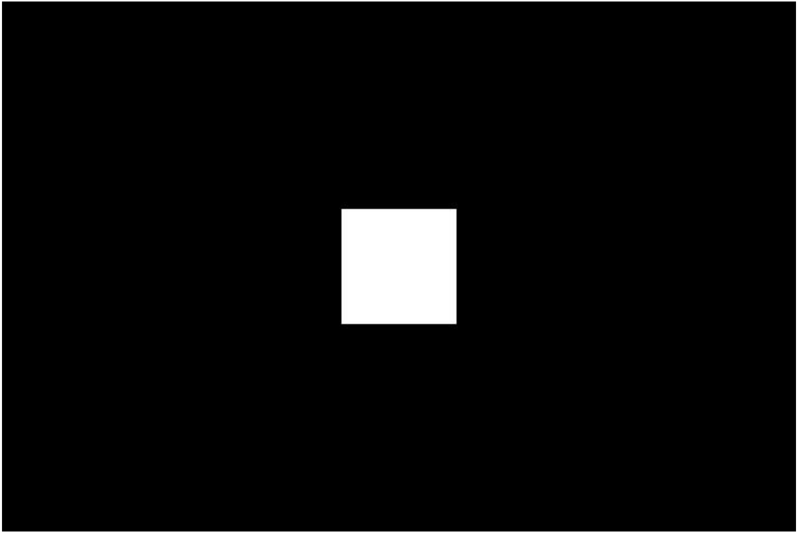


Filtering in frequency domain–image with noise

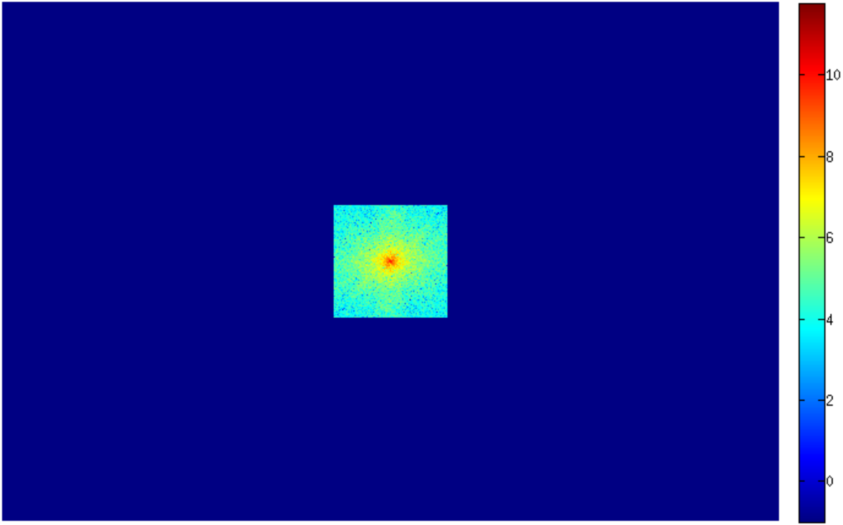
Filtering in frequency domain–image with noise spectrum



Filtering in frequency domain–mask

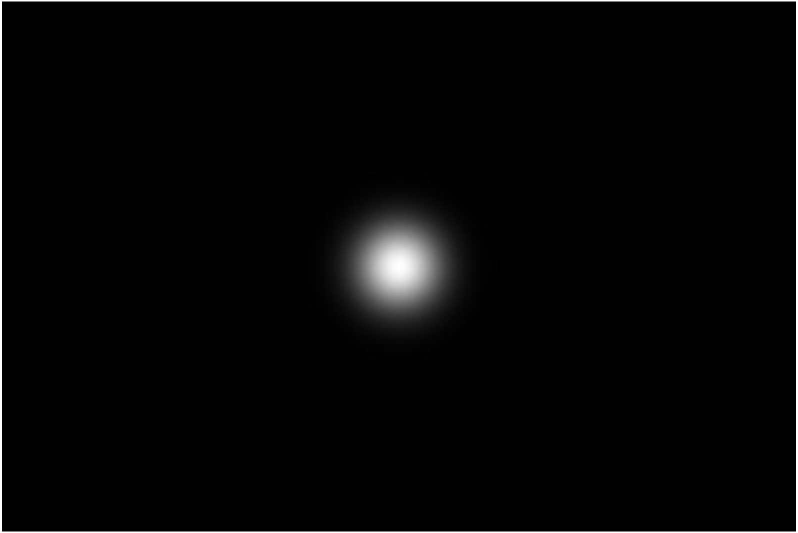


Filtering in frequency domain–mask applied

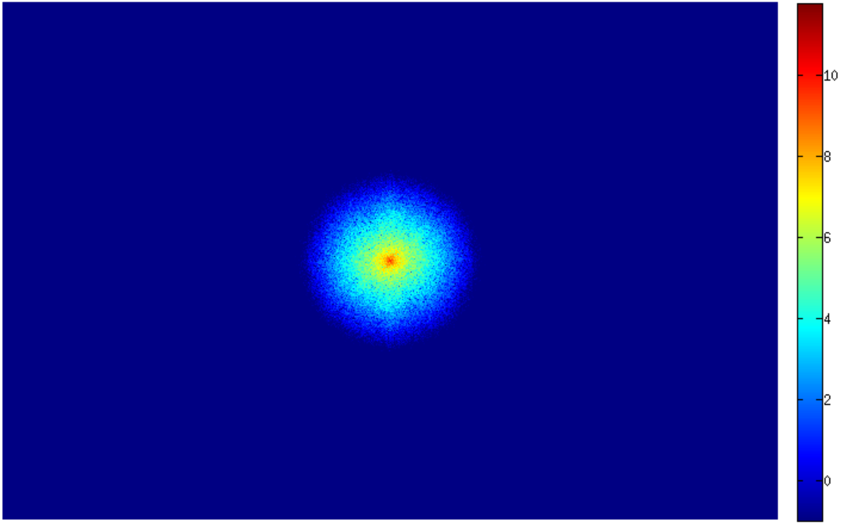


Filtering in frequency domain–filtered image

Filtering in frequency domain–Gaussian mask



Filtering in frequency domain–applied on spectrum



Filtering in frequency domain–filtered image

Filtering in frequency domain

Low-pass filter (LPF)

- Suppresses high frequencies – examples above

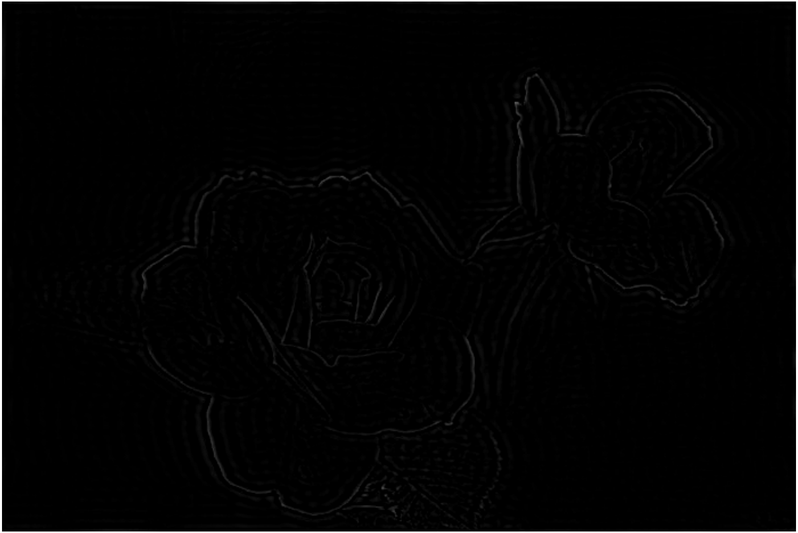
High-pass filter (HPF)

- Suppresses low frequencies

$$HPF = 1 - LPF$$

HPF example – Original image

HPF example – Ideal filter



HPF example – Gaussian filter

From frequency to space domain

- Convolution theorem

$$\mathcal{F}\{f * h\} = F \cdot H$$

- Discrete convolution

$$f(i,j) = \sum\sum_{(m,n)\in\mathcal{O}} h(i-m, n-j)g(m,n).$$

  - $\mathcal{O}$ – local neighbourhood of pixel (i,j).
  - h is called convolution mask

Image smoothing

Averaging

- Convolution mask for a 3×3 neighbourhood

$$h = \frac{1}{9}\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

Averaging – 9×9 filter

Gaussian filtering

- Elements in convolution mask are defined by

$$h(m,n) = \frac{1}{\sum\sum_{(i,j)\in\mathcal{O}} e^{\frac{-(i^2+j^2)}{2\sigma^2}}} e^{\frac{-(m^2+n^2)}{2\sigma^2}}$$

- Convolution mask for a 5×5 neighbourhood with mean = 0 and $\sigma = 3$

$$h = \begin{pmatrix} 0.0318 & 0.0375 & 0.0397 & 0.0375 & 0.0318 \\ 0.0375 & 0.0443 & 0.0469 & 0.0443 & 0.0375 \\ 0.0397 & 0.0469 & 0.0495 & 0.0469 & 0.0397 \\ 0.0375 & 0.0443 & 0.0469 & 0.0443 & 0.0375 \\ 0.0318 & 0.0375 & 0.0397 & 0.0375 & 0.0318 \end{pmatrix}$$

Gaussian filter – 9×9 filter, $\sigma = 3$

Averaging and Gauss filtering

- Disadvantage : Blurs edges.

Linear filters

- Averaging and Gaussian filtering are examples of linear filters – use convolution or correlation as filtering operation
- Separable filters – convolution mask can be written as product of two 1D vectors
  - computational optimization

Non-linear filters

Median filter

- Replace current point in the image by median of brightnesses in its neighbourhood.
- Reduces blurring of edges

Image with noise

Median filtered image, 9×9 neighbourhood

Median filter

- Particularly effective against salt-and-pepper noise.

Image with noise

Median filtered image, 9×9 neighbourhood

Median filtering

Disadvantage when filtering by rectangular neighbourhood

- Damages thin lines
- Damages sharp corners.

Can be avoided using different shape of neighbourhood.

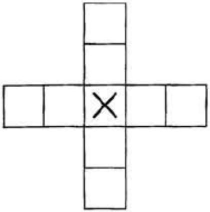


Figure 5.14: Horizontal/vertical line preserving neighborhood for median filtering.

Non-linear filters

Minimum filter

- Similar to median filter but chooses minimal value in the neighbourhood

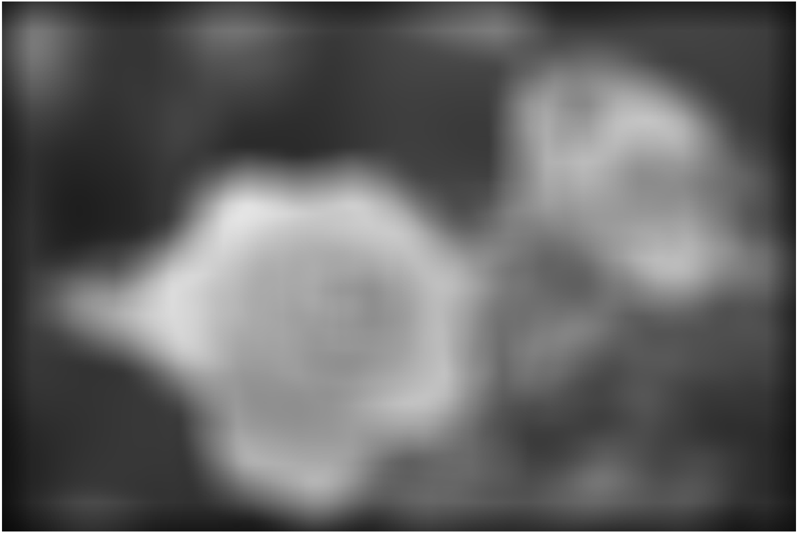
Maximum filter

- Similar to median filter but chooses maximal value in the neighbourhood

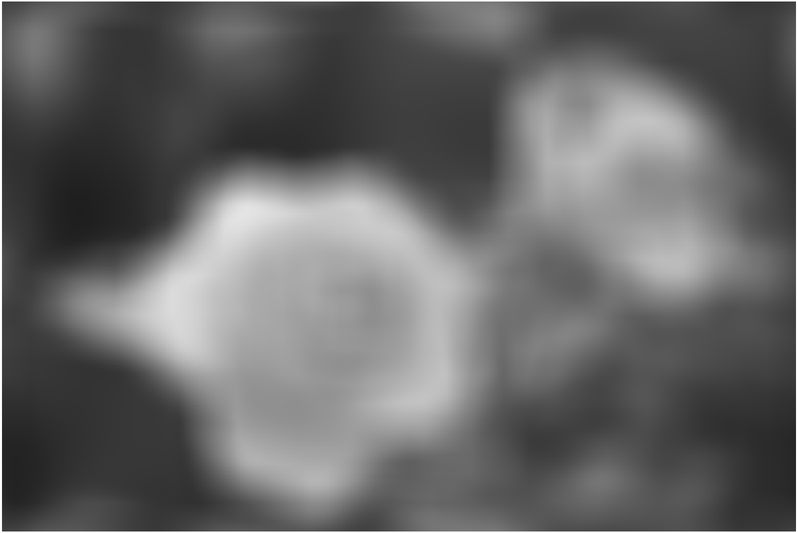
Boundary problem

- Padding with zeros
- Periodic copy
- Reflection
- Boundary repetition

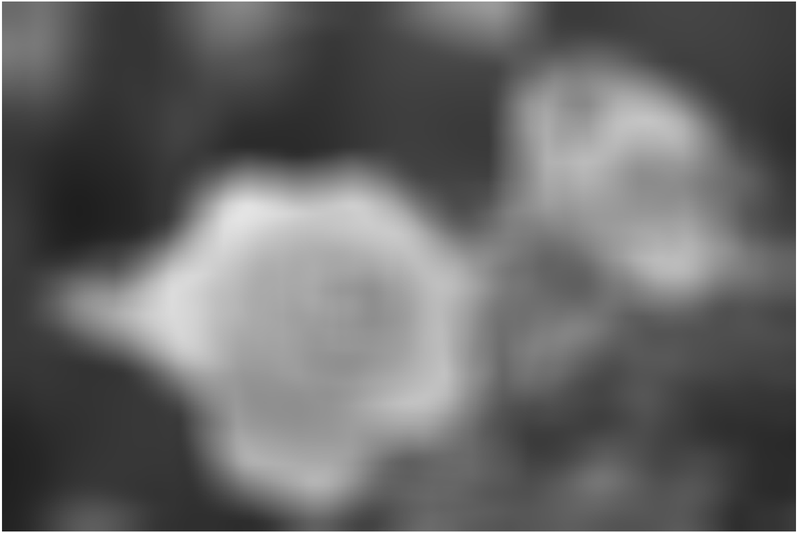
Zero padded boundary – 51×51 averaging kernel



Periodic – 51×51 averaging kernel



Reflection – 51×51 averaging kernel



Repeated boundary – 51×51 averaging kernel