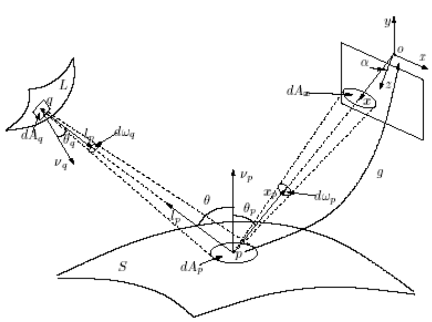
Images

- photometric aspects of image formation
- gray level images
- point-wise operations
- linear filtering

Image model	Mathematical tools
Analog intensity function	Analysis
Temporal/spatial sampled function	Linear algebra
Quantization of the gray levels	Numerical methods
Point sets	Set theory, morphology
Random fields	Stochastic methods
List of image features, regions	Geometry, AI, logic

Basic Photometry



Basic ingredients

Radiance – amount of energy emitted along certain direction

Iradiance – amount of energy received along certain direction

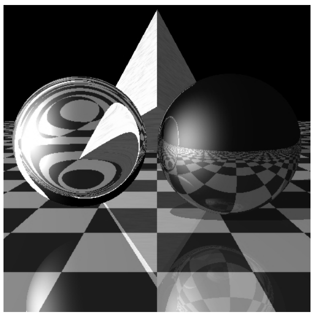
BRDF – bidirectional reflectance distribution

Lambertian surfaces – the appearance depends only on radiance, not on the viewing direction

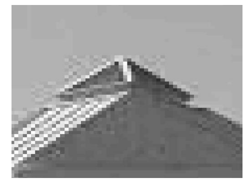
Image intensity for a Lambertian surface

$$I(\mathbf{x}) = \gamma \mathcal{R}(p)$$

Challenges



Images



- Images contain noise – sources sensor quality, light fluctuations, quantization effects

Image Noise Models

- Additive noise: Most commonly used

$$\hat{I}(x,y) = I(x,y) + n(x,y)$$

- Multiplicative noise:

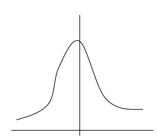
$$\hat{I}(x,y) = I(x,y).n(x,y)$$

- Impulsive noise (salt and pepper):

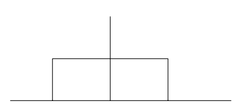
$$I(i,j) = \begin{cases} \hat{I}(i,j) & \text{if } x < l \\ i_{\min} + y(i_{\max} - i_{\min}) & x \geq l \end{cases}$$

Additive Noise Models

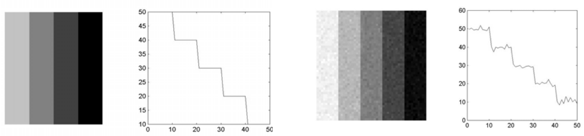
- Noise Amount: SNR = σ_s / σ_n
- Gaussian - Usually, zero-mean, uncorrelated



- Uniform



Example

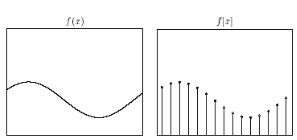


How can we reduce noise?

- Image acquisition noise due to light fluctuations and sensor noise can be reduced by acquiring a sequence of images and averaging them.
- Solution – filtering the image

Image Processing

1D signal and its sampled version



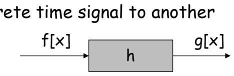
$$f[x] = f(xT), \quad x \in \mathbb{Z},$$

f = { f(1), f(2), f(3), …, f(n)}

f = {0, 1, 2, 3, 4, 5, 5, 6, 10 }

Discrete time system

- maps 1 discrete time signal to another



$$g(x) = T\{f(x)\}$$

- Special class of systems – linear , time-invariant systems

Superposition principle

$$T\{af_1(x) + bf_2(x)\} = aT\{f_1(x)\} + bT\{f_2(x)\}$$

Shift (time) invariant – shift in input causes shift in output

$$g[x] = T\{f[x]\} \rightarrow g[x - x_0] = T\{f[x - x_0]\}$$

Examples

Convolution sum:

$$f[x] = \sum_{k=-\infty}^{\infty} f[k]\delta[x-k]$$

$\delta[x]$ unit impluse – if x = 0 it's 1 and zero everywhere else

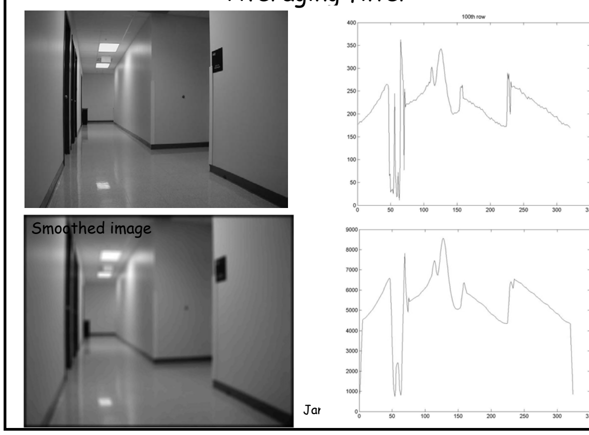
Every discrete time signal can be written as a sum of scaled and shifted impulses

The output the linear system is related to the input and the transfer function via convolution

Convolution sum: $g[x] = \sum_{k=-\infty}^{\infty} f[k]h[x-k]$

Notation: $g[x] = f[k] * h[x]$

Averaging filter



Averaging filter

$$g[x] = \sum_{k=-\infty}^{\infty} f[k]h[x-k]$$

$$f[x] = [...0, 0, 2, -2, 2, 0, 0, ...]$$

Averaging box filter $h[x] = \frac{1}{3}[1, 1, 1]$

$$g[x] = \sum_{k=-1}^{1} f[k]h[x-k]$$

Averaging filter
Center pixel weighted more

$$h[x] = [0.25, 0.5, 0.25]$$

2D convolution - next

Convolution in 2D

$$g[x,y] = \sum_{k=-\frac{w}{2}}^{\frac{w}{2}} \sum_{l=-\frac{w}{2}}^{\frac{w}{2}} f[k,l]h[x-k,y-l]$$

f

10	11	10	0	0	1
9	10	11	1	0	1
10	9	10	0	2	1
11	10	9	10	9	11
9	10	11	9	99	11
10	9	9	11	10	10

h

1/9

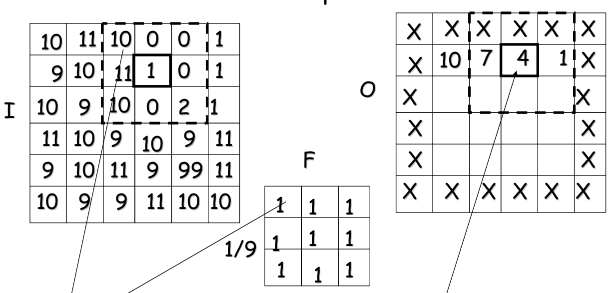
1	1	1
1	1	1
1	1	1

g

X	X	X	X	X	X
X	10				X
X					X
X					X
X					X
X	X	X	X	X	X

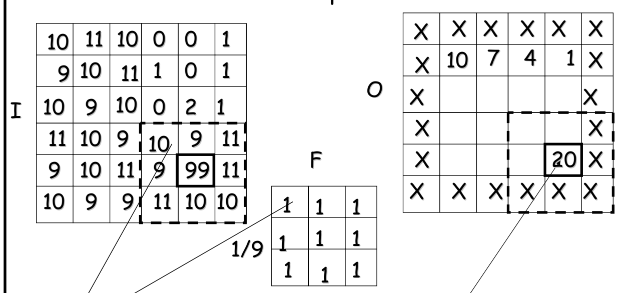
1/9.(10x1 + 11x1 + 10x1 + 9x1 + 10x1 + 11x1 + 10x1 + 9x1 + 10x1) =
1/9.(90) = 10

Example:



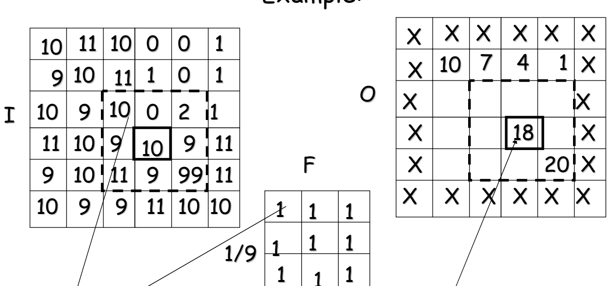
1/9.(10x1 + 0x1 + 0x1 + 11x1 + 1x1 + 0x1 + 10x1 + 0x1 + 2x1) =
1/9.(34) = 3.7778

Example:



1/9.(10x1 + 9x1 + 11x1 + 9x1 + 99x1 + 11x1 + 11x1 + 10x1 + 10x1) =
1/9.(180) = 20

Example:



1/9.(10x1 + 0x1 + 2x1 + 9x1 + 10x1 + 9x1 + 11x1 + 9x1 + 99x1) =
1/9.(159) = 17.6667

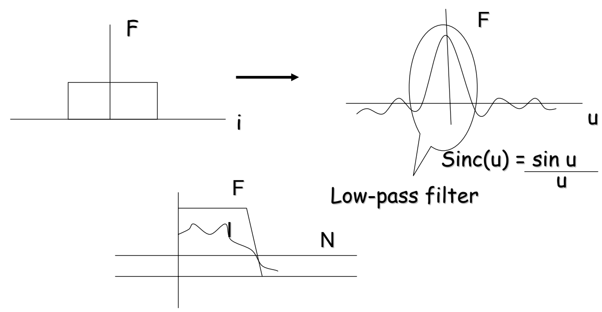
How big should the mask be?

- The bigger the mask,
  - more neighbors contribute.
  - smaller noise variance of the output.
  - bigger noise spread.
  - more blurring.
  - more expensive to compute.

Limitations of averaging

- Signal frequencies shared with noise are lost, resulting in blurring.
- Impulsive noise is diffused but not removed.
- The secondary lobes of the sinc let noise into the filtered image.
- It spreads the noise, resulting in blurring.
- Impulsive noise is diffused but not removed.

Frequency Domain Interpretation:



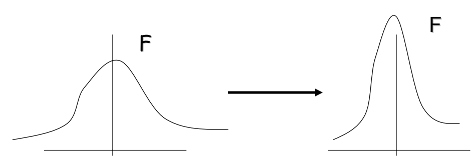
Gaussian Filter

- A particular case of averaging
  - The coefficients are samples of a 1D Gaussian.
  - Gives more weight at the central pixel and less weights to the neighbors.
  - The further away the neighbors, the smaller the weight.

$$g(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-x^2}{2\sigma^2}},$$

Sample from the continuous Gaussian

Frequency Domain Interpretation



Gaussian filter is the only one that has the same shape in the space and frequency domains.

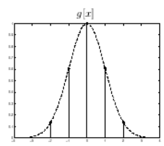
There are no secondary lobes – i.e. a truly low-pass filter

How big should the mask be?

- The std. dev of the Gaussian σ determines the amount of smoothing.
- The samples should adequately represent a Gaussian
- For a 98.76% of the area, we need

  $m = 5\sigma$

  $5.(1/\sigma) \leq 2\pi \Rightarrow \sigma \geq 0.796, m \geq 5$



5-tap filter

g[x] = [0.136, 0.6065, 1.00, 0.606, 0.136]

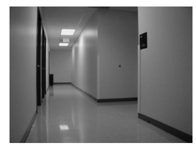
Image Smoothing

- Convolution with a 2D Gaussian filter

$$\tilde{I}(x,y) = I(x,y) * g(x,y) = I(x,y) * g(x) * g(y)$$

- Gaussian filter is separable, convolution can be accomplished as two 1-D convolutions

$$\tilde{I}[x,y] = I[x,y] * g[x,y] = \sum_{k=-\frac{w}{2}}^{\frac{w}{2}} \sum_{l=-\frac{w}{2}}^{\frac{w}{2}} I[k,l]g[x-k]g[y-l]$$

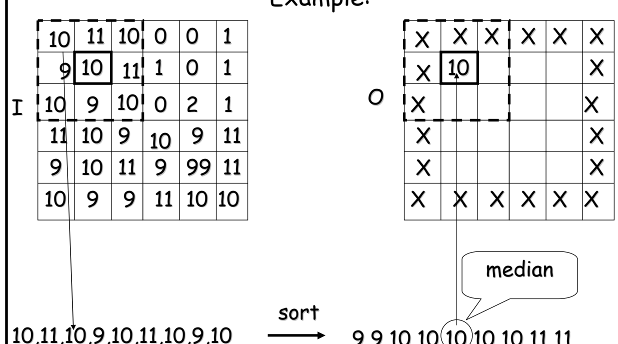


Non-linear Filtering

- Replace each pixel with the MEDIAN value of all the pixels in the neighborhood.

- Non-linear
- Does not spread the noise
- Can remove spike noise
- Expensive to run

Example:



Example:

I

10	11	10	0	0	1
9	10	11	1	0	1
10	9	10	0	2	1
11	10	9	10	9	11
9	10	11	9	99	11
10	9	9	11	10	10

O

X	X	X	X	X	X
X	10	10	1	1	X
X					X
X					X
X					X
X	X	X	X	X	X

median

11,10,0,10,11,1,9,10,0 —sort→ 0,0,1,9,10,10,10,11,11

CS682, Jana Kosecka

Example:

I

10	11	10	0	0	1
9	10	11	1	0	1
10	9	10	0	2	1
11	10	9	10	9	11
9	10	11	9	99	11
10	9	9	11	10	10

O

X	X	X	X	X	X
X	10				X
X					X
X			9		X
X				10	X
X	X	X	X	X	X

median

10,9,11,9,99,11,11,10,10 —sort→ 9,9,10,10,10,11,11,11,99

CS682, Jana Kosecka