Image filtering

In Fourier domain
In spatial domain

Linear filters
Non-linear filters
Image filtering in spatial domain

\[ g(x,y) = \text{IF}\{} H(u,v) \ F\{f(x,y)\} \ \text{=} \]

\[ IF \ \{} H(u,v) \} ** IF \ \{} F \ \{} f(x,y) \} \ \text{=} \]

\[ h(x,y) ** f(x,y) \]
Filter definition in spatial domain

\[ IF \{ H(u, v) \} = h(x, y) \]

\( \hat{h}(x, y) \) is selected so that

\[ F(\hat{h}(x, y)) = \hat{H}(x, y) \approx H(x, y) \]
Image and the filer mask convolution

$$G(i,j) =
\begin{align*}
l(i-1,j-1)h(-1,-1) + l(i-1,j)h(-1,0) + l(i-1,j+1)h(-1,1) + \\
l(i,j-1)h(0,-1) + l(i,j)h(0,0) + l(i,j+1)h(0,1) + \\
l(i+1,j-1)h(1,-1) + l(i+1,j)h(1,0) + l(i+1,j+1)h(1,1)
\end{align*}$$

This is true for symmetric masks only!
Computing the filtered image

source image $f$  
output image $g$

g(x,y) = h(x,y) * f(x,y)
Boundary effects

source image $f$

output image $g$
Boundary effects – 3x3 mask

Boundary columns and rows of (NxN) image are neglected and the filtered image is of size (N-2)x(N-2)
Image filtering – the algorithm

f, g : array[0..N-1, 0..N-1] of byte;
{ size2 – half size of the mask }
h : array[-size2..size2,-size2..size2] of integer;
...
for i:=1 to N-2 do for j:=1 to N-2 do
   begin
      g[i,j]:=0;
      for k:=-size2 to size2 do for l:=-size2 to size2 do
         g[i,j]:=g[i,j] + f[i+k,j+l] * h[i+k,j+l];
   end;
...

Range check g[i,j] !!!
Low pass filter

\[ h_1 = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad h_2 = \frac{1}{25} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \]

Can one use mask of even size?
Frequency characteristics of low pass filters

\[ H = \text{freqz}(h, m, n) \]

for 5x5 mask

for 3x3 mask
Low-pass filtering the image

Source image

3x3 mask

5x5 mask
Gaussian filter

\[ h = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \]

\[ h(x, y) = e^{-\pi(x^2 + y^2)} \frac{-\pi d_0^2}{d_0^2} \]

\[ H(u, v) = e^{-\pi d_0^2 (u^2 + v^2)} \frac{-\pi d_0^2 (u^2 + v^2)}{N} \]
Image filtering using the Gaussian filter

source image

filtered image
Image low-pass filters - examples

Image distorted by the Gaussian noise $N(0, 0.01)$

Low pass filter 3x3

for grayscale <0,1>

Gaussian filter 3x3

Butterworth filter $D_0=50$
Image low-pass filters - examples

Image distorted by the Gaussian noise $N(0, 0.01)$

low-pass filter 5x5

Gaussian filter 5x5

Butterworth filter $D_0=30$
Image low-pass filters - examples

Image distorted by the Gaussian noise $N(0, 0.002)$

Low pass filter 3x3

Gaussian filter 3x3

Butterworth filter $D_0=50$
High-pass filters (derivative filters)

\[ h_1 = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \]

\[ h_2 = \begin{bmatrix} 0.17 & 0.67 & 0.17 \\ 0.67 & -3.33 & 0.67 \\ 0.17 & 0.67 & 0.17 \end{bmatrix} \]
High-pass filtering the image

mask $h_1$

mask $h_2$
The „high boost” filter

\[ f(x, y) = f_L(x, y) + f_H(x, y) \]

\[ f_{HB}(x, y) = Af(x, y) - f_L(x, y) = \]
\[ = (A - 1) f(x, y) + f(x, y) - f_L(x, y) = \]
\[ = (A - 1) f(x, y) + f_H(x, y), \quad A \geq 1 \]

\[ A=? \]

\[ h_{HB} = \begin{bmatrix}
-1 & -1 & -1 \\
-1 & 9A -1 & -1 \\
-1 & -1 & -1 \\
\end{bmatrix} \]
High boost filter - example

Laplace filter

A=1.1

A=1.5
A modified Laplace filter

In order to keep the average value of the image add 1 do the centre element of the Laplace mask

\[ h_2 = \begin{bmatrix} 0.17 & 0.67 & 0.17 \\ 0.67 & -3.33 & 0.67 \\ 0.17 & 0.67 & 0.17 \end{bmatrix} \]

\[ h'_2 = \begin{bmatrix} 0.17 & 0.67 & 0.17 \\ 0.67 & -2.33 & 0.67 \\ 0.17 & 0.67 & 0.17 \end{bmatrix} \]
Other high-pass filters

$$h'_3 = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$  
$$h'_4 = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$
High-pass filters

%MATLAB
out_image = filter2(filter_mask, in_image);
Nonlinear filters

The filtered image is defined by a non-linear function of the source image.

Can we compute spectral characteristics for nonlinear filters?

**NO**

Because transfer characteristics of nonlinear filters depend on image content itself!
Median filter (order statistic filter)

The median $m$ of a set of values (e.g. image pixels in the filtering mask) is such that half the elements in the set are less than $m$ and other half are greater than $m$.

$x(n)=\{1, 5, -7, 101, -25, 3, 0, 11, 7\}$

Sorted sequence of elements:

$x_s(n)=\{-25, -7, 0, 1, 3, 5, 7, 11, 101\}$
Median filtering the image

\[ g(x,y) = \text{median}\{f(x,y); (x,y) \in h}\]
Demo – median filter

Source image distorted by „salt and pepper noise”

Enhanced image using the median filter (3x3)"

%MATLAB
out_image = medfilt2(in_image, [m n]);
Median filter:

1. Excellent in reducing impulsive noise (od size smaller than half size of the filtering mask)

2. Keeps sharpness of image edges (as opposed to linear smoothing filters)

3. Values of the output image are equal or smaller than the values of the input image (no rescaling)

4. Large computing cost involved
Median filter

[1 x 3]

1/3*[1 1 1]

median

average
MATLAB Demo – median filter