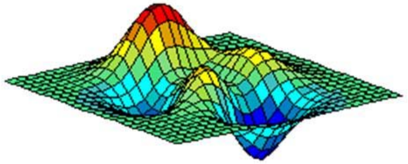
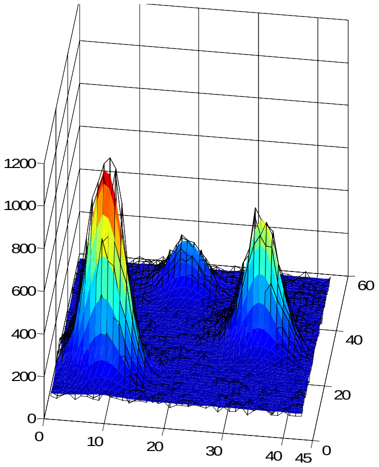
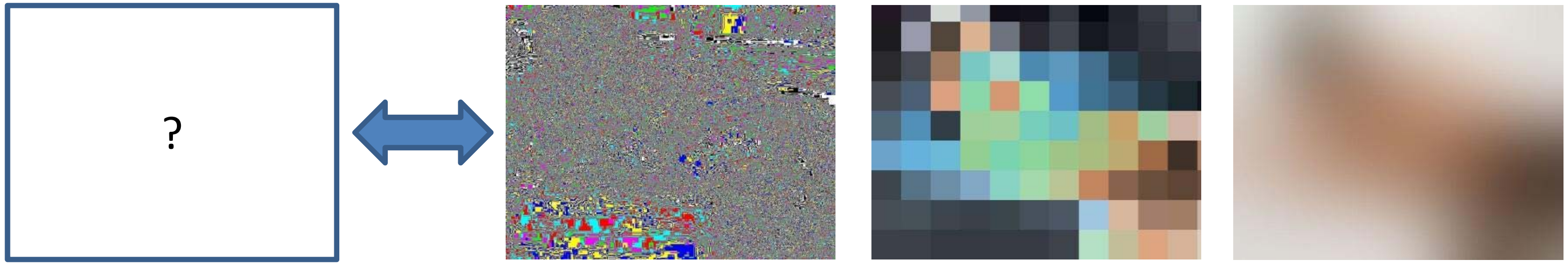




Digital image processing

Peter Nagy

Department of Biophysics and Cell Biology, University of Debrecen, Debrecen, Hungary



Digital image processing

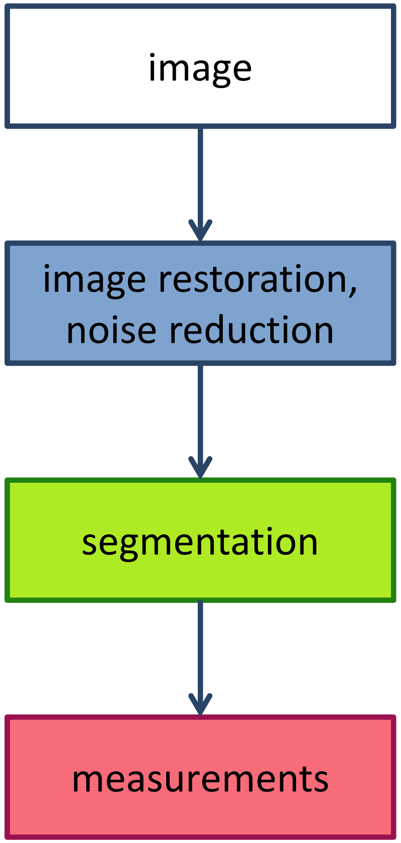
- Storing and compression (lossy and lossless (non-lossy)) of images, 3D image stacks, orthogonal view
- Digital and optical resolution
- Histogram, LUT

- Filters (in the spatial and frequency domains)
- Sources of image degradation
- Image restoration, noise reduction, smoothing
- Deconvolution
- Storing color images, color LUT, spectral unmixing
- Mathematical morphology

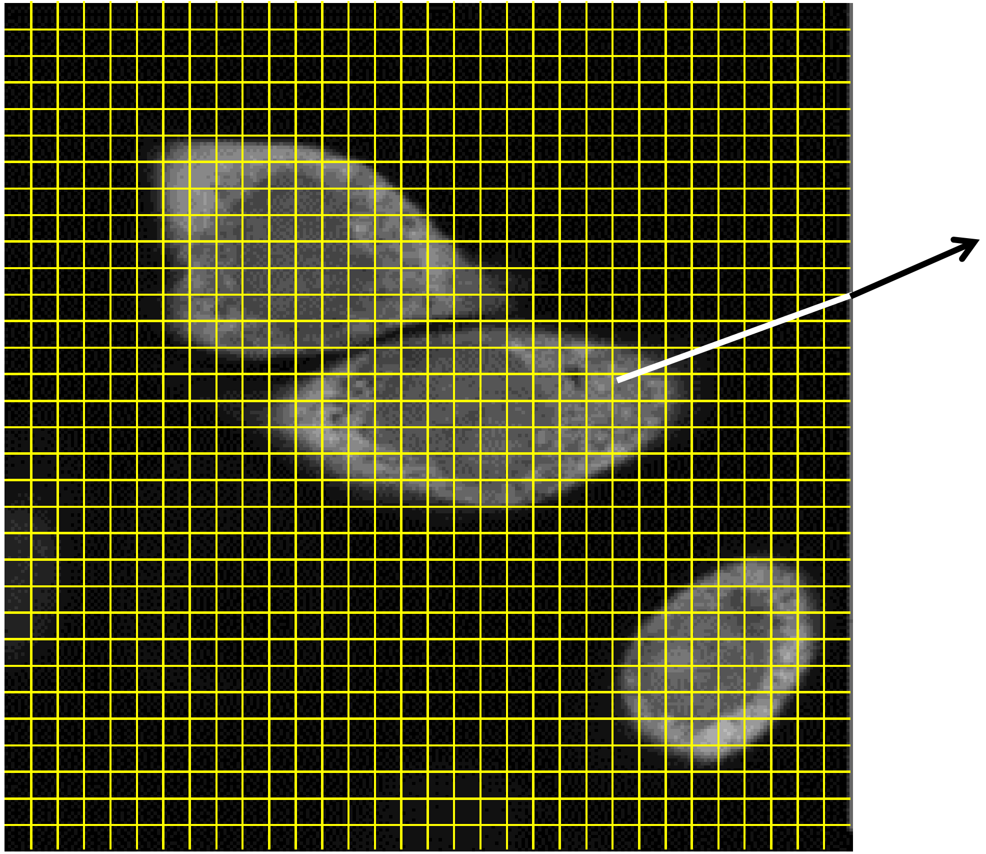
- Image segmentation

- Measurements

- Scientific image analysis programs



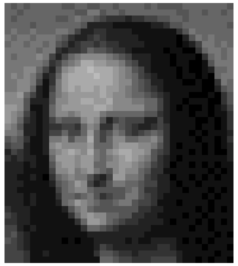
Representation of digital images



- Pixel („picture element”): the smallest unit in digital images
- Its value is a number whose value is proportional to the number of detected photons (in gray-scale images)
- According to how many values a pixel can have the image can be:

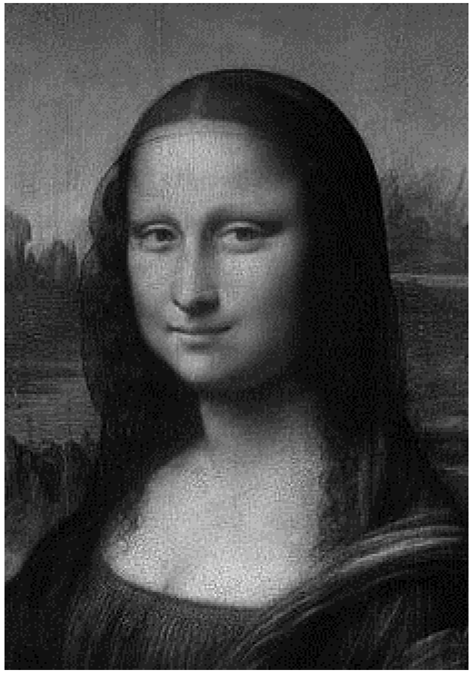
intensity scale	possible values	storage requirement
binary	0,1	1 bit/pixel
8-bit	0,1,…,255	1 byte/pixel
12-bit	0…4095	2 byte/pixel
16-bit	0…65535	2 byte/pixel
floating point	practically limitless	32 bit/pixel, 4 byte/pixel
double precision		64 bit/pixel, 8 byte/pixel

Why is digital image analysis necessary?



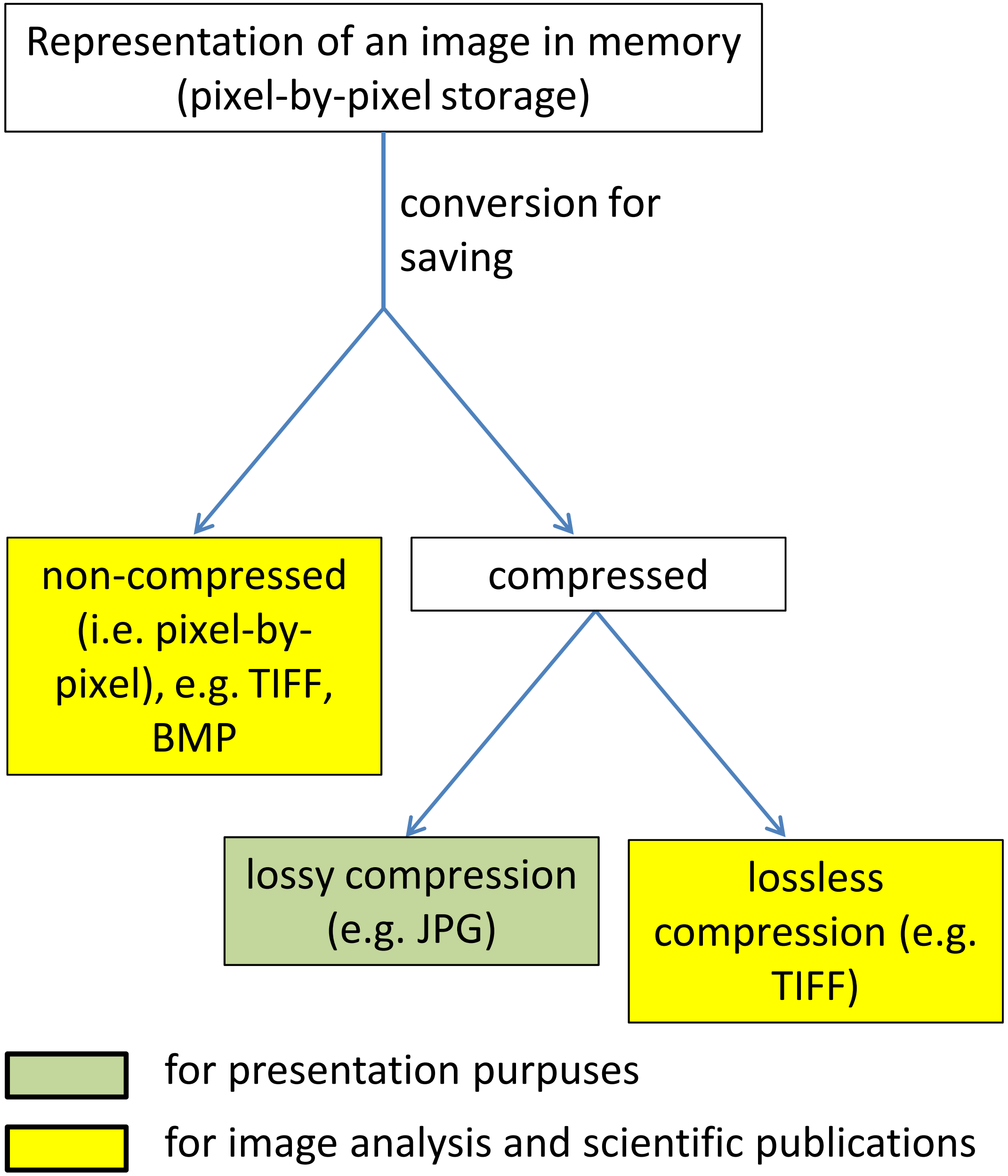
What is this?

96	95	94	96	99	101	101	99	95	87	92	100	94	90	96	100	99	97	97	99	103	103	97	92	92	90	91	96	100	100	101	104	100	99	96	94
99	98	97	99	101	103	102	101	103	105	103	94	93	109	110	92	99	96	94	95	99	103	105	105	100	100	93	87	92	103	102	94	102	100	98	95
99	98	97	98	100	101	100	99	95	92	101	110	104	96	87	76	56	53	49	47	49	55	62	66	80	92	101	98	94	96	100	102	105	103	99	96
102	101	101	101	101	102	101	101	109	102	108	108	81	50	38	38	30	27	24	19	18	20	25	30	42	54	76	96	103	99	100	107	106	104	101	97
110	110	110	110	109	108	109	109	117	111	94	62	36	40	47	40	34	34	32	28	23	21	20	21	21	19	26	51	86	110	109	98	107	105	102	100
112	113	114	113	112	111	112	113	110	83	56	39	38	53	65	61	37	37	34	31	27	21	17	14	13	17	12	11	40	87	110	106	109	107	105	104
112	113	115	114	112	112	113	115	95	59	43	55	63	62	60	57	51	47	40	34	30	25	19	15	11	19	19	8	8	36	81	115	113	111	109	108
115	117	118	118	116	115	117	119	48	44	47	56	71	91	92	74	66	56	44	35	30	26	21	16	17	5	8	17	7	2	44	102	116	114	112	110
118	112	116	122	119	128	120	73	38	65	91	114	135	140	134	133	120	105	88	65	39	27	25	20	15	16	8	10	15	7	15	47	113	109	112	116
119	119	122	126	125	122	97	50	65	98	126	143	159	163	157	155	142	128	113	90	60	42	32	21	15	16	10	11	16	6	5	23	85	117	114	117
122	125	126	127	129	108	67	36	91	130	159	167	175	179	174	170	162	149	136	115	83	58	40	23	18	17	10	9	14	8	1	6	40	110	114	115
124	125	126	129	127	86	42	45	106	148	173	172	174	179	177	173	169	157	146	127	98	72	50	31	23	19	11	7	9	10	8	9	8	86	115	115
123	122	131	133	119	63	23	60	116	154	174	168	167	173	173	171	168	155	143	127	102	81	61	41	28	21	15	11	8	11	13	11	2	56	112	114
125	123	139	131	101	43	14	70	121	151	165	160	161	165	165	165	161	146	134	121	101	85	69	51	37	23	19	18	11	9	10	5	5	26	92	102
132	131	144	111	69	32	20	79	122	143	154	155	159	161	158	159	157	142	131	122	108	99	87	71	56	27	17	19	11	8	9	4	7	8	62	85
141	138	143	89	41	27	32	90	125	141	151	156	164	163	157	158	160	146	137	132	123	118	109	95	74	32	13	15	9	9	14	10	9	6	44	79
149	150	128	85	24	16	36	99	130	158	146	155	154	161	167	162	157	140	134	138	133	123	109	96	69	41	16	10	11	7	7	12	5	11	31	68
150	147	123	74	26	19	42	98	131	133	120	133	160	156	152	132	97	100	102	106	106	99	88	82	59	37	18	14	13	8	7	13	8	11	26	58
145	133	105	51	30	22	36	74	98	95	84	89	138	144	130	86	45	73	83	76	73	60	44	42	55	36	20	18	16	9	8	13	11	10	20	49
138	120	90	35	35	24	28	51	58	77	66	59	101	141	116	57	54	90	88	60	50	42	35	46	68	43	22	17	17	11	8	12	11	9	17	46
123	117	93	37	35	22	34	72	72	94	71	73	95	154	109	57	90	120	105	65	58	68	83	107	89	54	22	14	15	12	9	10	12	9	17	46
93	107	92	41	24	13	41	114	122	123	92	111	118	163	103	72	103	132	129	104	101	113	123	135	96	57	20	11	15	13	9	9	14	9	16	44
60	84	73	40	17	10	39	130	146	145	129	140	137	161	106	91	111	138	155	151	145	142	129	113	85	50	17	10	15	13	10	10	14	8	12	38
43	66	53	39	23	19	33	122	139	158	165	151	142	157	119	108	132	150	170	171	161	152	130	100	71	41	15	12	15	13	10	11	13	5	8	33
51	45	47	28	29	22	23	101	148	156	158	139	152	160	119	103	123	145	158	165	156	140	118	77	57	35	16	13	15	13	13	15	14	11	18	34
40	35	38	24	27	21	19	88	135	147	153	139	148	148	113	103	124	143	145	148	141	121	101	73	50	30	14	13	16	14	12	13	12	8	12	25
34	29	31	20	25	19	14	67	126	146	147	127	141	151	117	92	93	139	148	140	130	108	84	61	42	26	13	13	16	14	11	11	11	7	9	17
36	31	28	19	24	18	11	47	106	124	130	126	122	93	52	38	71	141	153	128	114	97	75	53	39	25	13	12	14	13	11	11	12	9	10	16
39	33	25	17	21	18	13	33	81	97	123	153	137	68	34	54	83	128	125	108	102	84	67	56	40	27	15	11	12	13	12	12	12	11	13	17
37	33	23	17	20	19	18	25	73	100	111	130	135	107	86	81	81	90	86	104	114	85	64	58	38	28	16	11	12	14	14	13	11	12	14	16
37	36	26	22	22	21	21	18	59	104	101	94	107	109	91	54	70	78	87	114	116	86	65	50	32	25	16	12	13	16	15	12	9	11	14	16
40	42	32	28	25	22	21	11	36	94	121	141	141	112	100	90	78	103	116	116	94	76	67	43	26	22	16	13	15	17	15	11	7	11	15	17
51	38	35	29	13	12	20	16	15	43	90	128	131	107	87	83	84	88	94	92	78	58	45	40	28	20	15	15	15	12	11	13	9	11	14	17
61	50	42	33	20	17	18	16	11	20	49	95	137	153	143	128	109	100	88	77	63	48	38	35	24	18	13	14	14	11	10	11	9	11	14	17
58	55	40	28	25	22	18	21	18	12	17	52	111	155	155	133	103	85	65	53	46	38	33	30	21	17	14	14	13	11	10	11	9	11	14	16
63	62	41	23	25	22	17	24	21	19	14	21	56	94	99	83	67	55	43	39	38	33	27	24	20	18	16	15	14	13	13	13	10	12	14	15
79	77	51	26	23	19	14	21	11	18	16	9	17	37	45	40	42	37	33	34	32	26	21	20	23	22	20	16	14	14	14	14	11	12	13	14
79	76	55	29	19	17	17	19	10	14	15	12	12	20	30	36	34	30	28	27	25	22	24	28	31	31	26	18	13	12	13	12	12	12	13	13
71	67	57	36	18	18	23	19	18	14	15	19	17	19	37	60	49	42	35	31	30	30	37	44	46	45	38	25	15	13	13	11	12	12	13	13
75	71	67	48	21	19	25	15	19	12	14	21	16	15	46	86	77	66	54	47	44	43	47	52	59	58	48	31	19	15	14	12	13	13	12	12
69	70	61	52	21	22	19	20	17	18	18	17	22	16	32	98	104	86	69	63	59	55	57	62	71	69	57	38	22	15	13	13	10	14	15	12



- In order to replace the (pattern) recognition capability of the human visual system.
- Analyze images in a quantitative and reproducible way.

Storage of images



Representation of an image in memory (pixel-by-pixel storage)

conversion for saving

non-compressed (i.e. pixel-by-pixel), e.g. TIFF, BMP

compressed

lossy compression (e.g. JPG)

lossless compression (e.g. TIFF)

🟩 for presentation purpuses

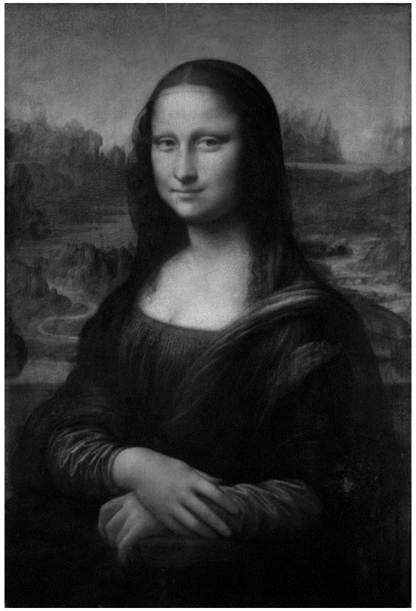
🟨 for image analysis and scientific publications

Parts of image files:

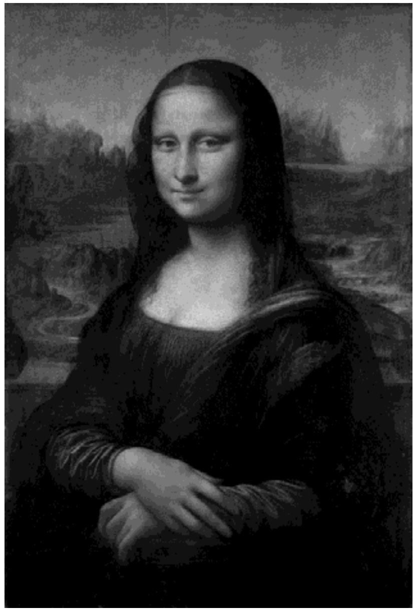
- descriptor: tells how to interpret image data (how many bits/pixel, X-Y size, etc.)
- image data

Lossy compression: the image reconstructed from the compressed data is not identical to the original one.

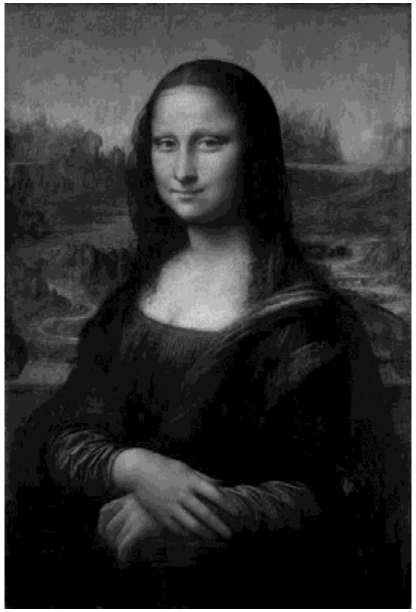
Lossy image compression



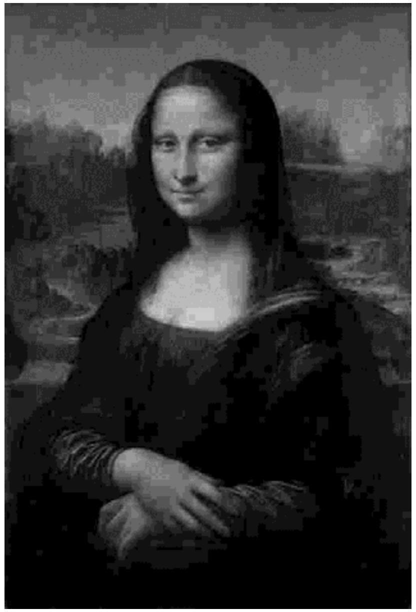
TIFF, 388 KB



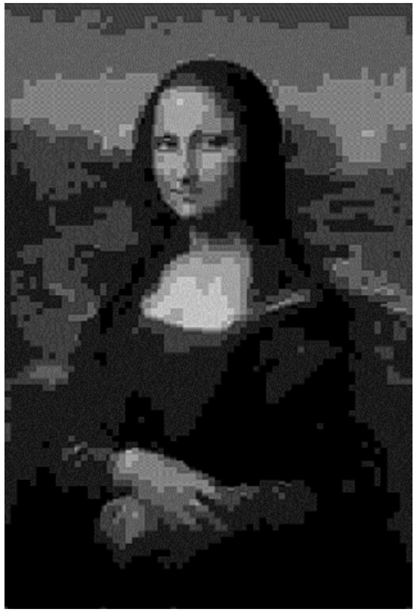
JPG, 59KB



JPG, 26KB



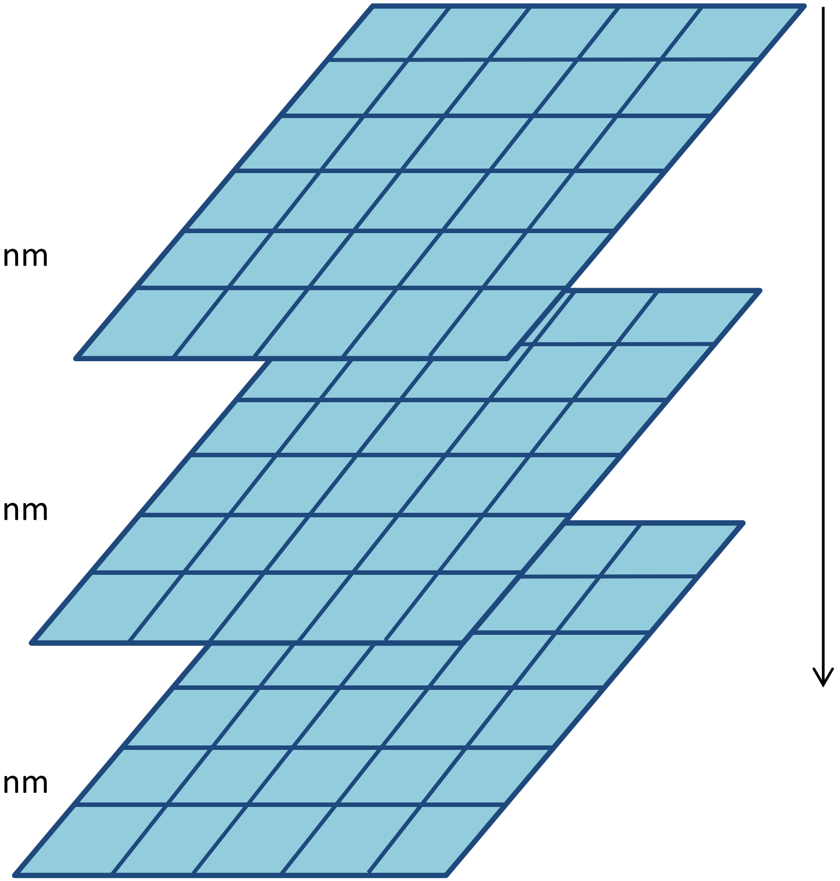
JPG, 20 KB



JPG, 18 KB

Images with more than two dimensions: 3D, 4D, 5D, …

3D:



z=0 μm t=0 sec λ_{em}=500 nm

z=0.3 μm t=5 sec λ_{em}=510 nm

z=0.6 μm t=10 sec λ_{em}=520 nm

The 3rd dimension can be:

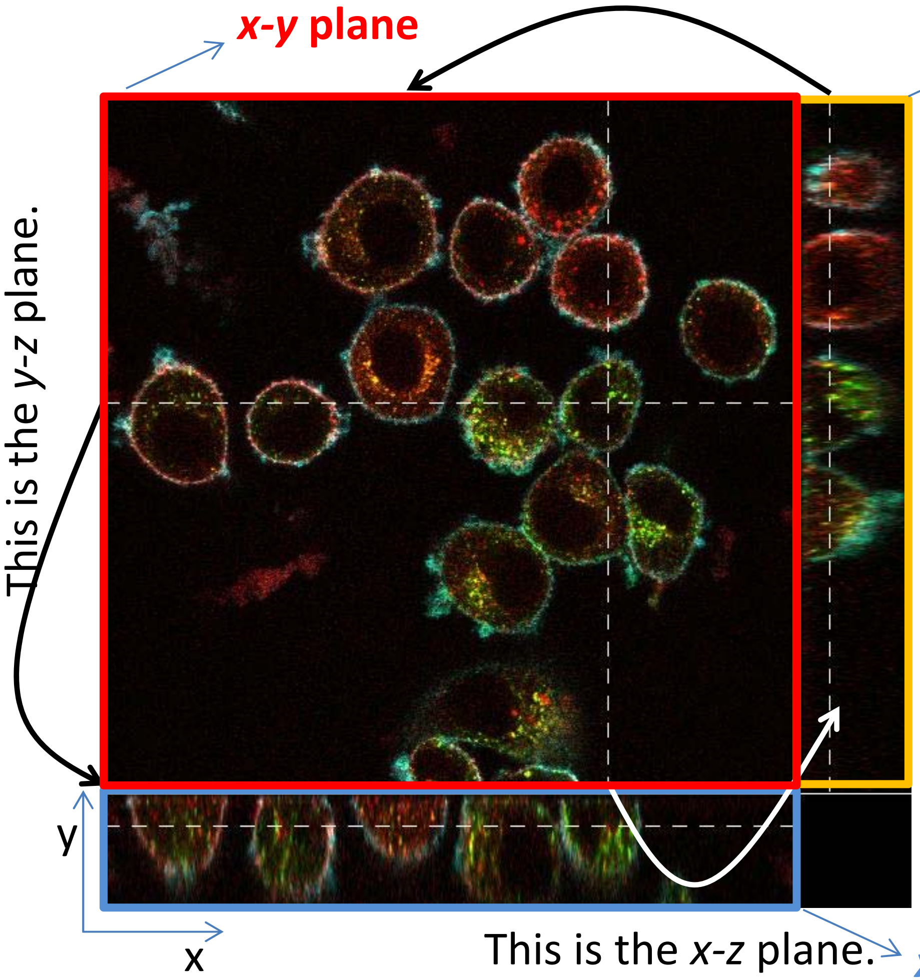
- depth (e.g. confocal microscopic 3D image stack)
- time
- wavelength
- …

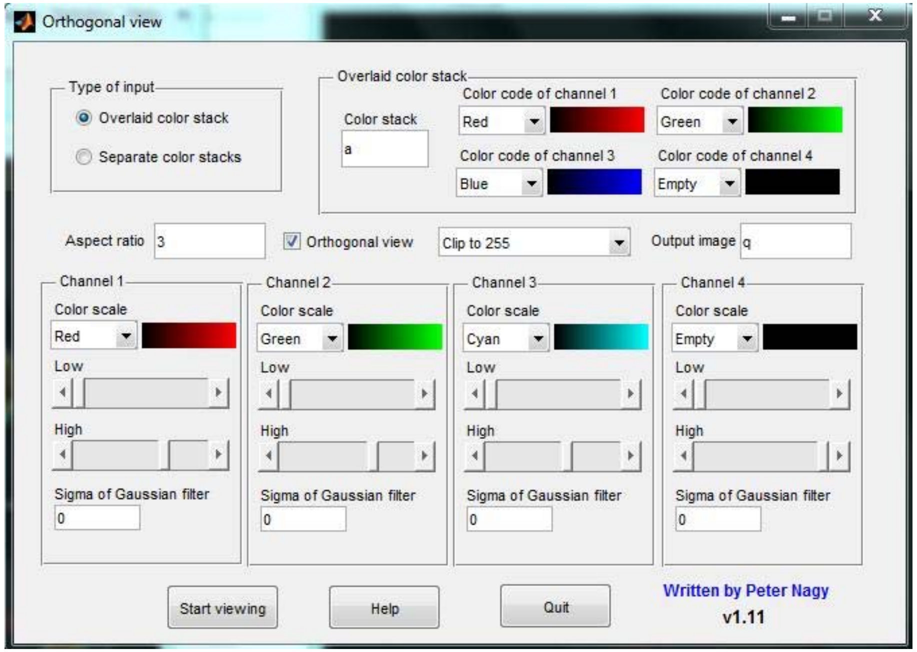
>3D:

Images with more than three dimensions are also possible, e,g. dimensions 1-3: x,y,z; 4th dimension: time; 5th dimension: wavelength.

Displaying 3D images 1.

Orthogonal view/sections:





Orthogonal view

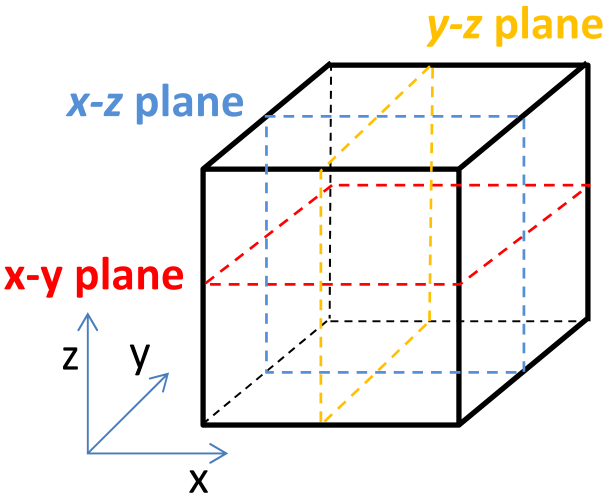
Type of input
● Overlaid color stack
○ Separate color stacks

Overlaid color stack
Color stack: a
Color code of channel 1: Red
Color code of channel 2: Green
Color code of channel 3: Blue
Color code of channel 4: Empty

Aspect ratio: 3
☑ Orthogonal view
Clip to 255
Output image: q

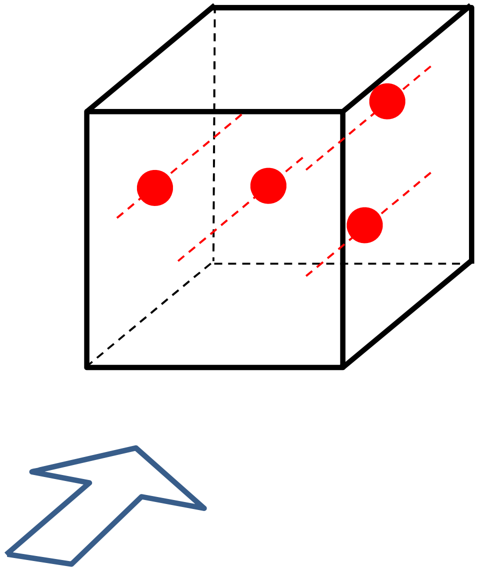
Channel 1	Channel 2	Channel 3	Channel 4
Color scale: Red	Color scale: Green	Color scale: Cyan	Color scale: Empty
Low	Low	Low	Low
High	High	High	High
Sigma of Gaussian filter: 0	Sigma of Gaussian filter: 0	Sigma of Gaussian filter: 0	Sigma of Gaussian filter: 0

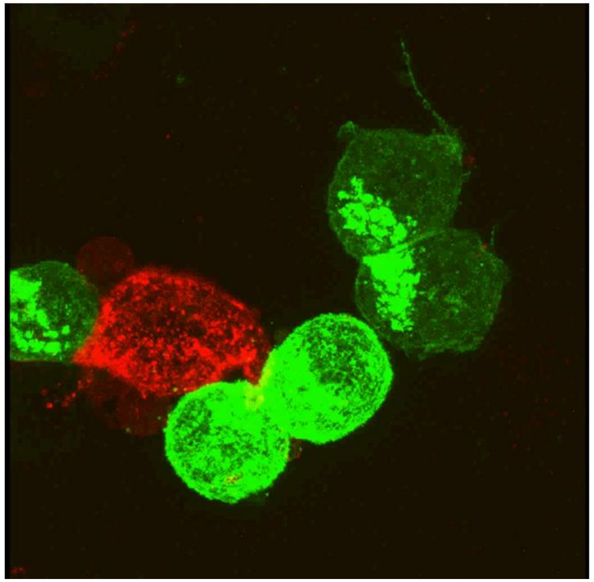
Written by Peter Nagy
v1.11



Displaying 3D images 2.

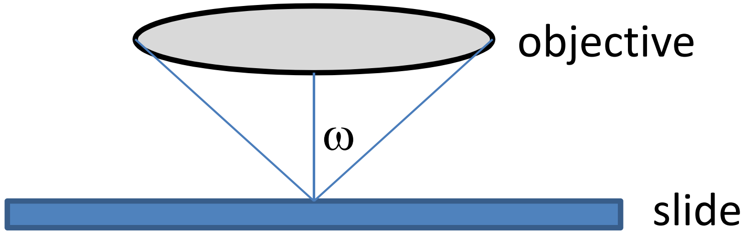
Maximum intensity projection:





- The objects are viewed from this direction.
- Along each projection line we look for the pixel which is the brightest and this pixel will be displayed in the 2D projection.
- Advantage: fast to calculate
- Disadvantage: weak 3D effect
- The level of 3D impression can be improved if the maximum intensity projection is displayed from several directions: “rotation”.

Optical and digital resolution power



Optical resolution power:

$$d = \frac{\lambda}{2NA} = \frac{\lambda}{2n\sin\omega}$$

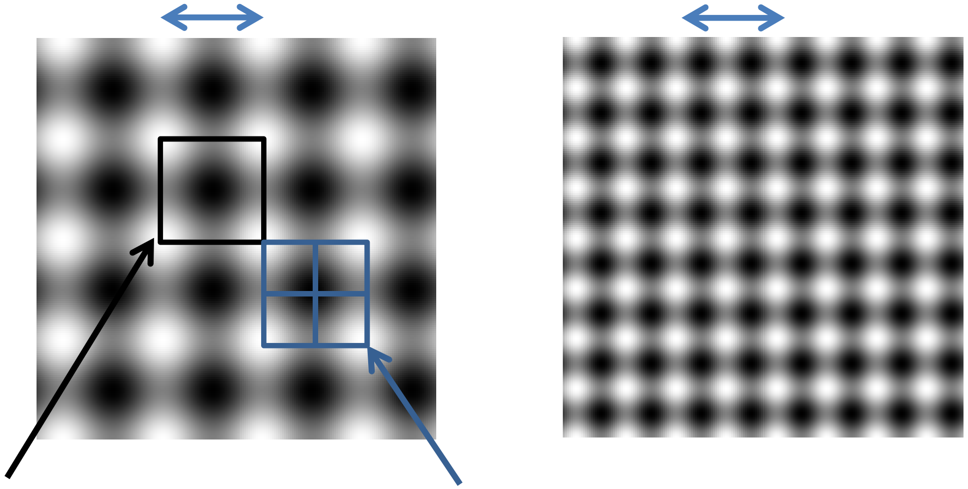
λ – wavelength

n – index of refraction of the medium between the objective and the object

Digital resolution power:

- A digital resolving power better than the optical resolution is useless.
- Approximate rule: the pixel size has to be approximately the same as the optical resolution.
- Accurate rule: **Nyquist sampling theorem**: if the largest frequency component of a function is *f*, then it can be reconstructed by sampling it at a rate of 2*f* ⇒ if the smallest optically resolvable distance is *d*, then the pixel size has to be *d*/2.

d – with this optical resolution power the perodic structure on the left is resolvable, the right one isn’t.

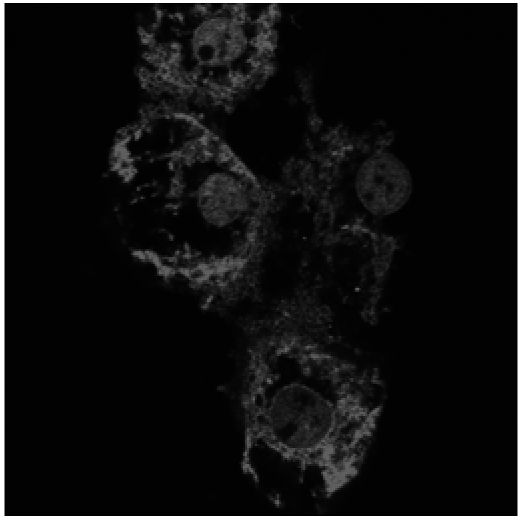


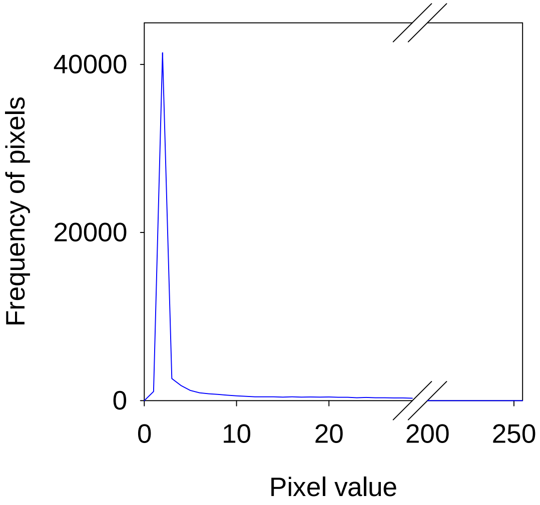
If the optical resolution is *d*:

Pixel size according to the approximate rule.

Pixel size according to the Nyquist theorem.

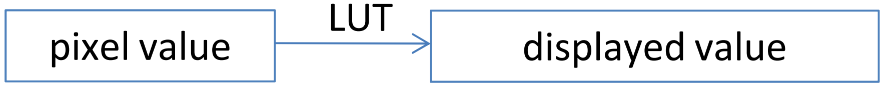
Histogram, LUT





The histogram on the left displays the distribution of pixel intensities **stored** in the images.

- The displayed color or gray-scale value of pixels is determined by the LUT (look-up table). The LUT assigns a display value to each stored value.
- The LUT does not influence the stored image content, only the display mode.



pixel value

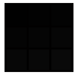
LUT

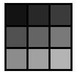
displayed value

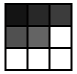
Display of gray-scale images is usually done on 256 levels of gray (0 – black, 255 – white), since the human eye cannot discriminate more gray levels.

Stored pixel values:

1	2	3
4	5	6
7	8	9



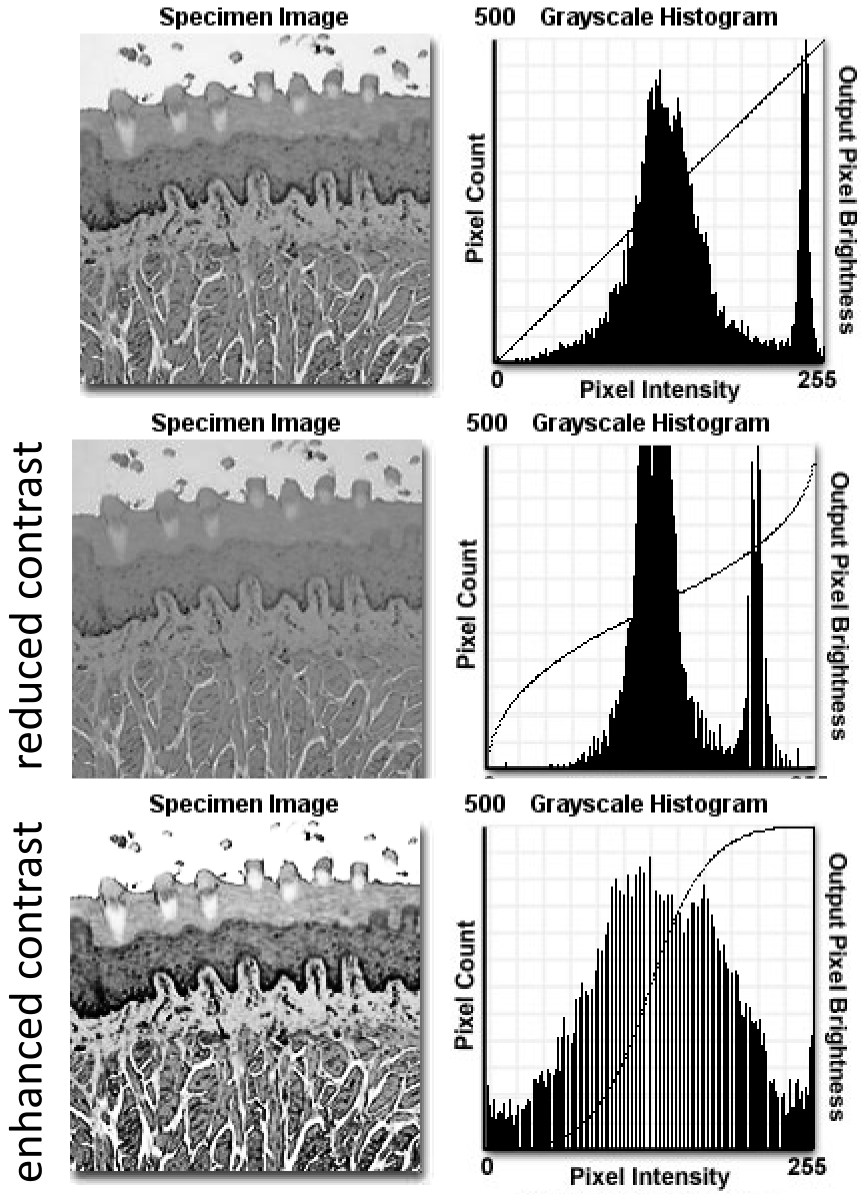




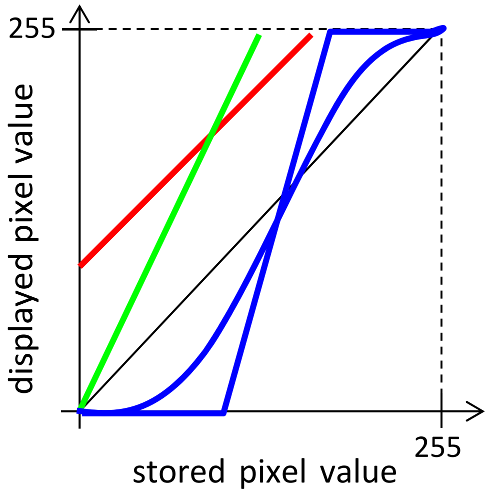
	1	2	3	4	5	6	7	8	9
LUT1	1	2	3	4	5	6	7	8	9
LUT2	20	40	60	80	100	120	140	160	180
LUT3	20	40	60	80	100	255	255	255	255

The effect of the LUT

The LUT can be given as a plot of displayed pixel values against stored pixel values.



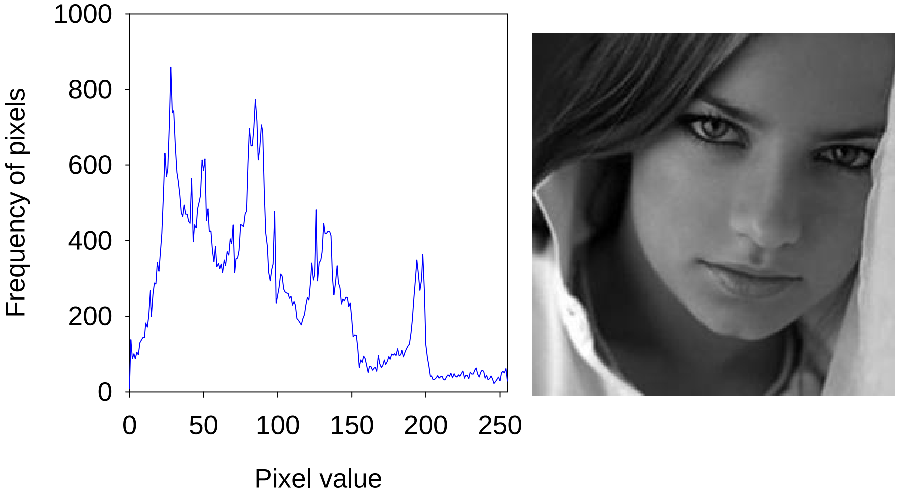
www.olympusmicro.com



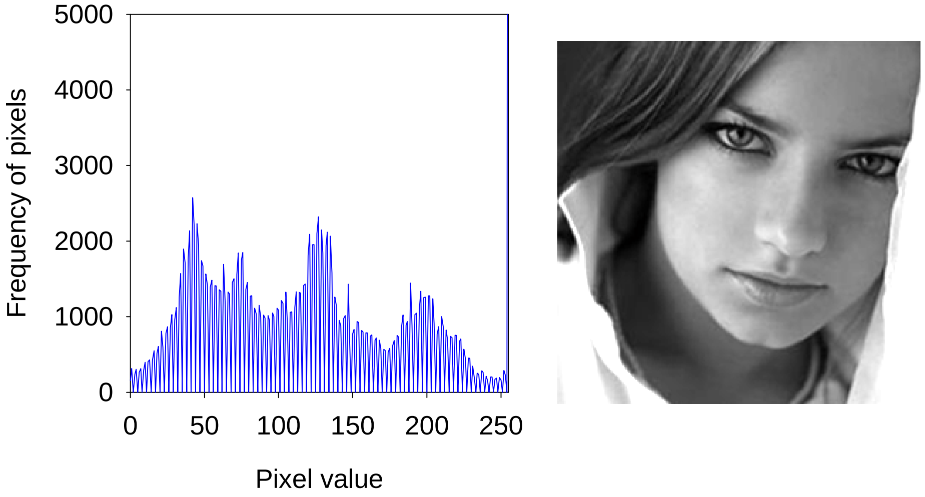
- increased brightness: the display value of each pixel is increased
- increased intensity: steeper relationship between the displayed and stored pixel values. The displayed value of pixels with high stored intensity is increased preferentially.
- increased contrast: the intensity range for pixels in the middle of the scale is increased.

Modifying the LUT with Corel PhotoPaint

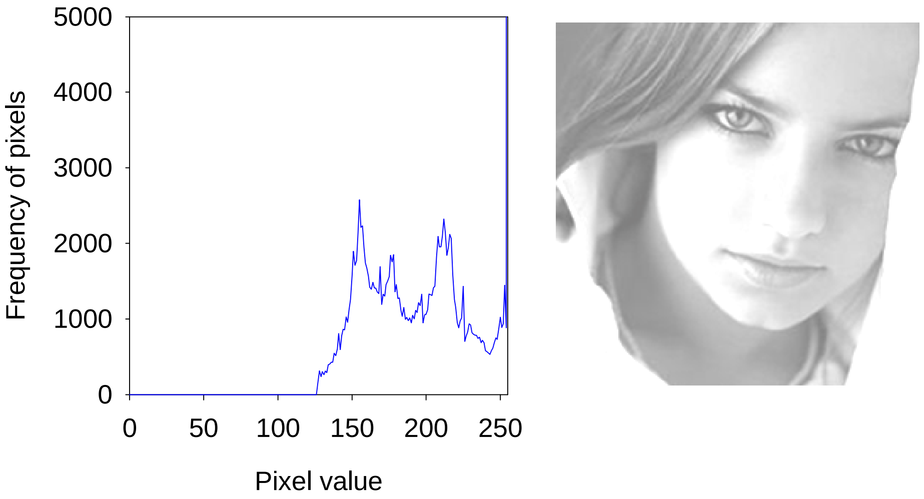
Original



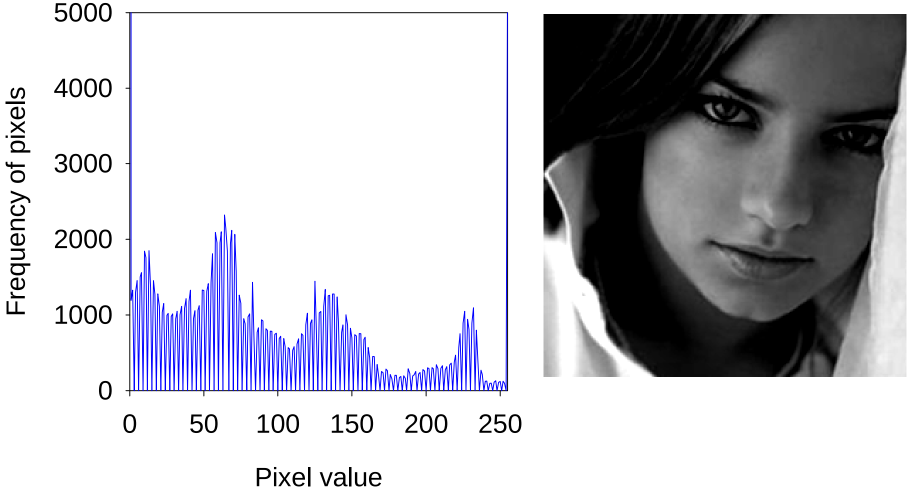
Intensity=50



Brightness=50

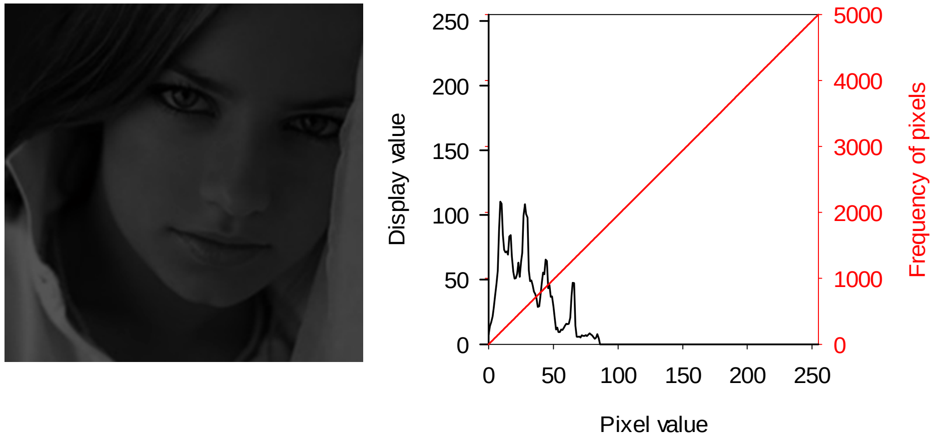


Contrast=50

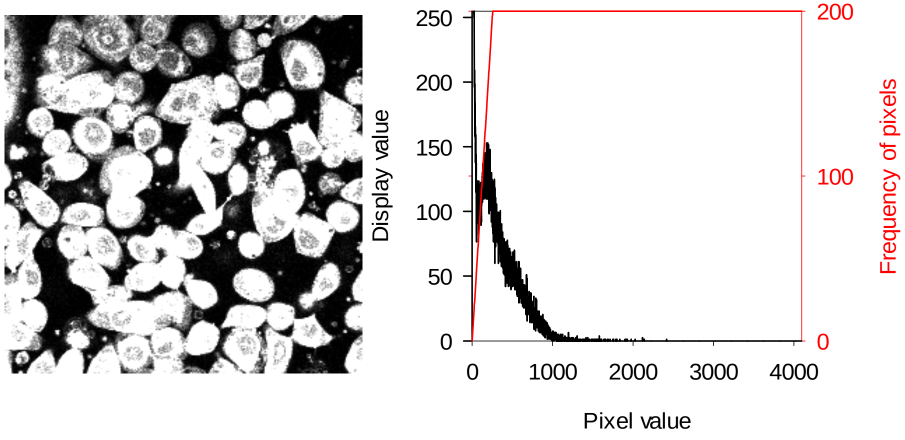


The effect of the LUT

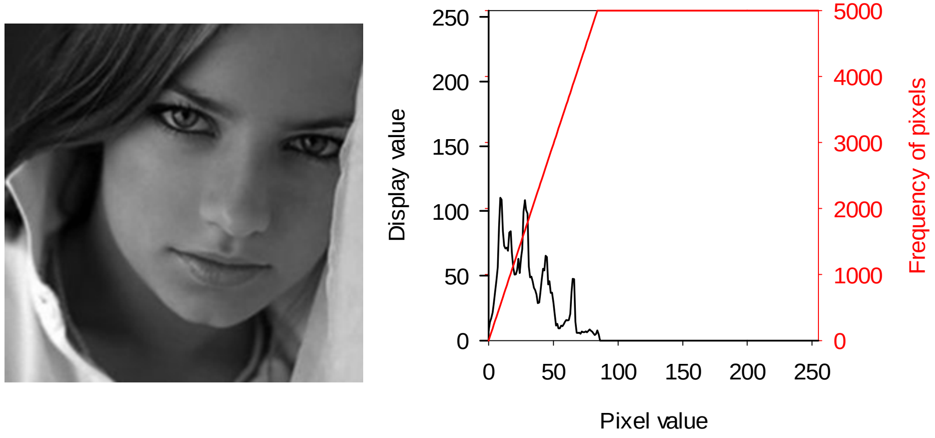
Original:



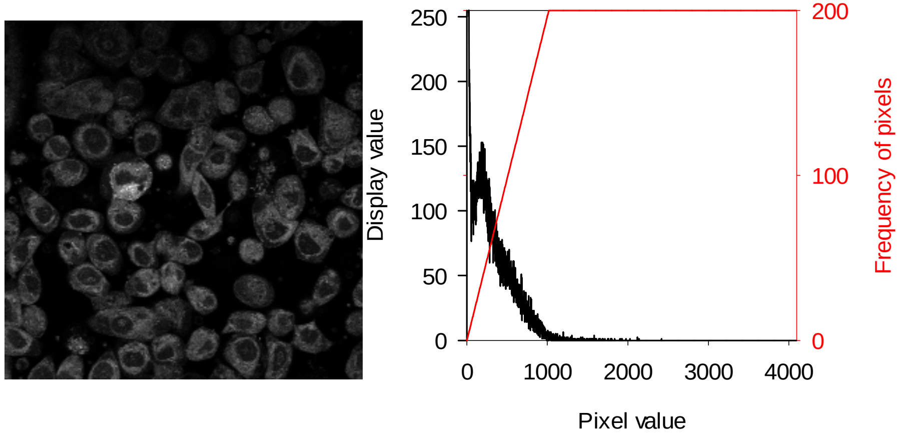
Original (stored on 12 bits):



Linear stretch:



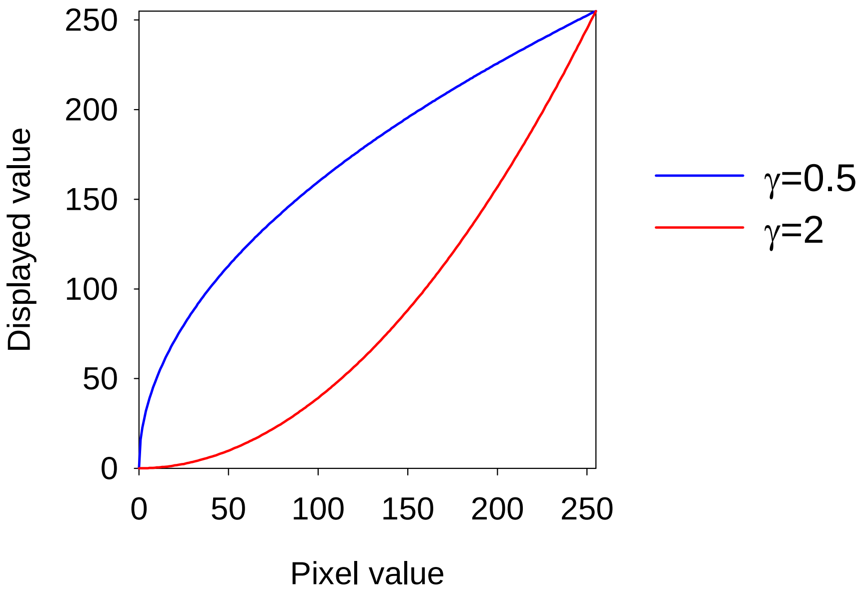
Linear stretch:



In linear stretching the the lowest and highest displayed intensities are assigned to the highest and lowest stored values and the range in between is distributed linearly.

Gamma correction

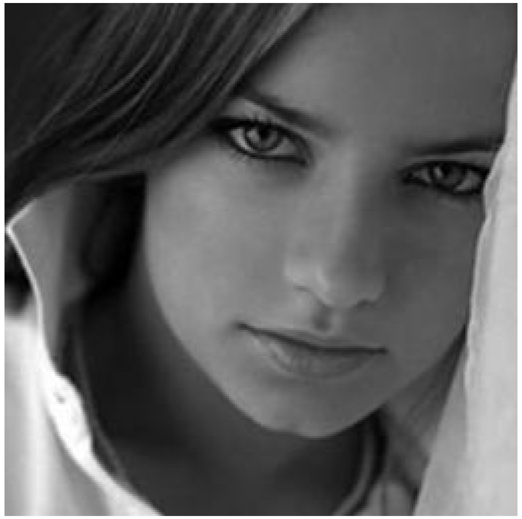
$$Pixel_{display} = const\ Pixel_{stored}^{\gamma}$$



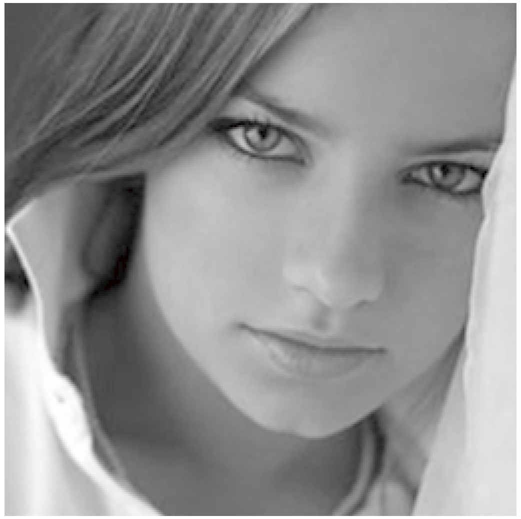
The displayed and stored pixel values are assigned to each other according to a power function.

$\gamma<1$ → gamma compression

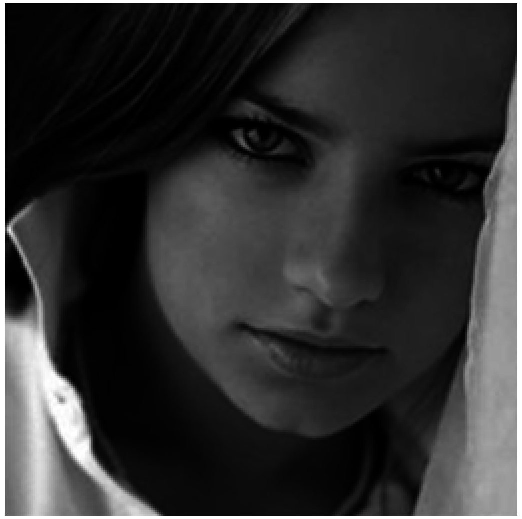
$\gamma>1$ → gamma expansion



eredeti

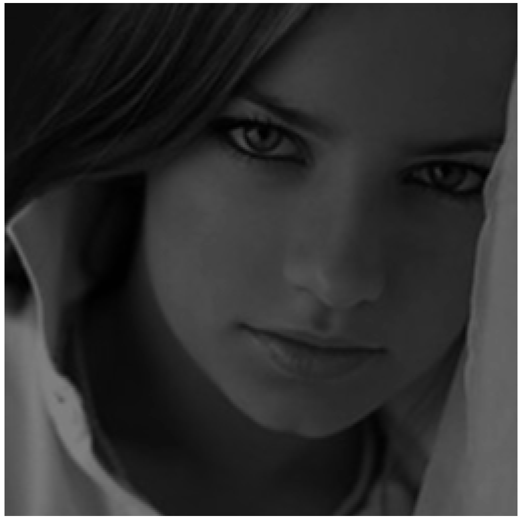


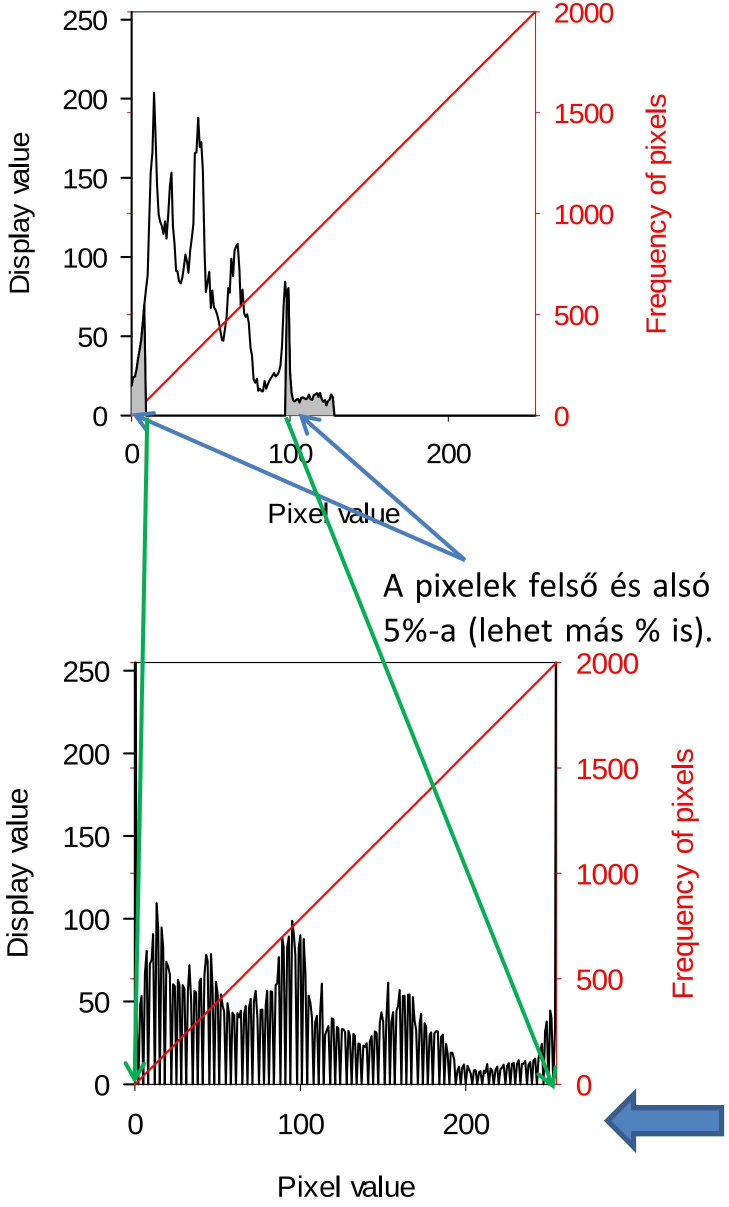
$\gamma=0.5$



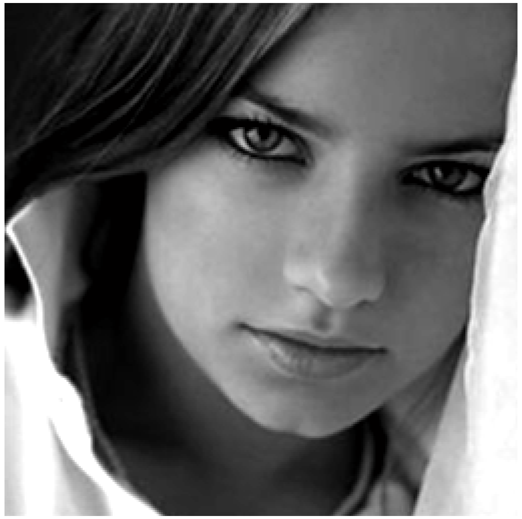
$\gamma=2$

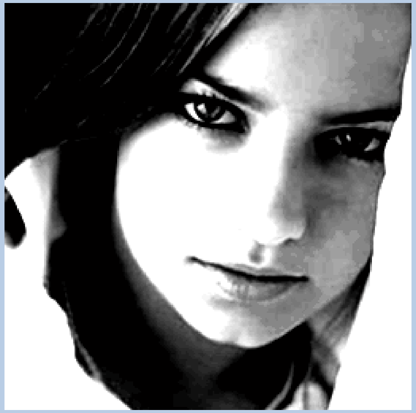
Contrast stretching: enhancing the contrast

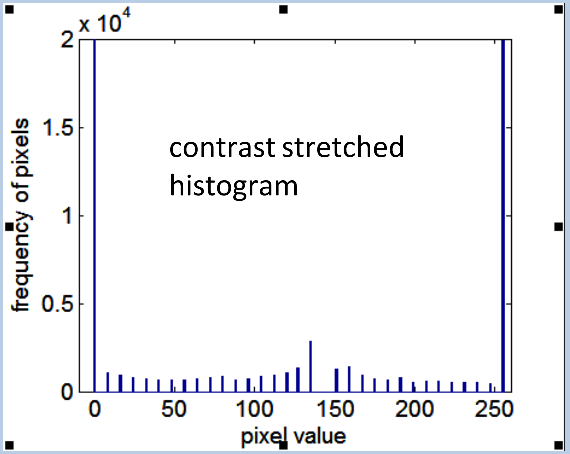




A pixelek felső és alsó 5%-a (lehet más % is).







Contrast stretching by dragging the lowest and highest 30% to 0 and 255, respectively → the majority of pixels will be either black or white.

0 and 255 is assigned to the 5th and 95th percentile and the range in between is distributed linearly.

Contrast stretching can be performed on the stored or the displayed (as above) pixel values. Modifying the stored pixel values for optimal display is only advised after evaluation of the image is completed.

Histogram equalization

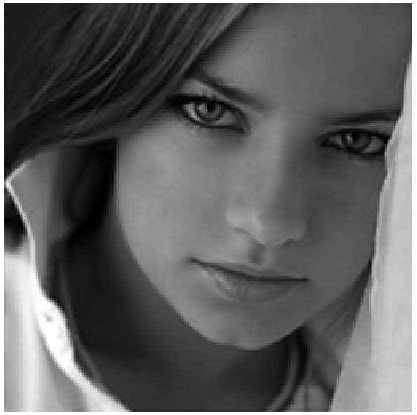
An algorithm ensuring that each intensity in the image is present with the same relative frequency. Intensities 0,1,…, I_{max} are present in the image and $p(f)$ is the distribution function of the intensity:

$$p(f) = P(I \leq f)$$

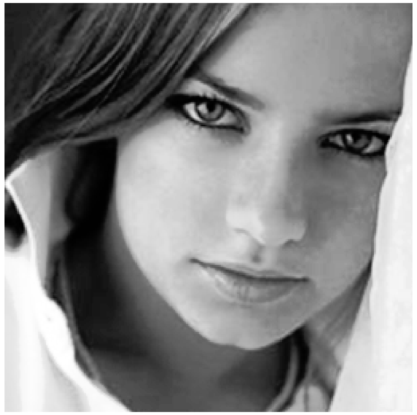
If intensities are displayed according to

$$I = I_{max}\, p(f)$$

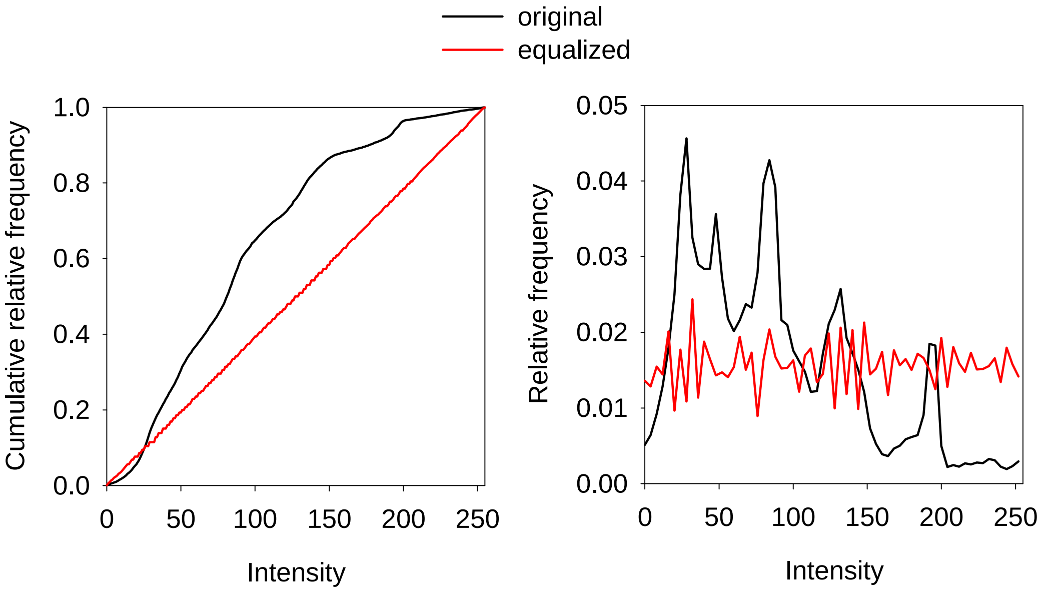
then the relative fraction of intensities lower than I will be I/I_{max}.



original



equalized



Filters

- A tool used during evaluation and noise reduction of images which calculates the new value of a pixel based on the intensities of the pixel and its neighbors according to a given function.
- Representing filters:
  - using a formula:

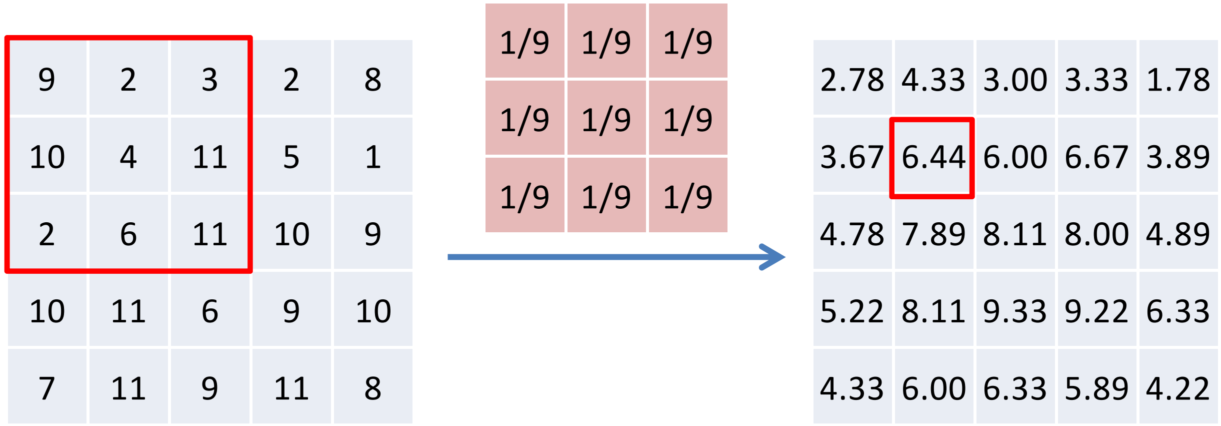
$$g_{x,y} = \sum_{k=-m}^{m} \sum_{l=-m}^{m} w_{k,l} f_{x+k,y+l}$$

$g_{x,y}$ – calculated new value of the pixel

$f_{x+k,y+l}$ – original values of the pixel and its neighbors

$w_{k,l}$ – weighting factor

  - by drawing the filter:



9	2	3	2	8
10	4	11	5	1
2	6	11	10	9
10	11	6	9	10
7	11	9	11	8

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

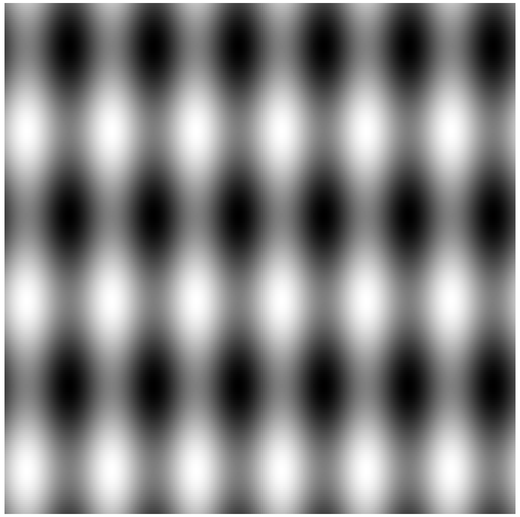
2.78	4.33	3.00	3.33	1.78
3.67	6.44	6.00	6.67	3.89
4.78	7.89	8.11	8.00	4.89
5.22	8.11	9.33	9.22	6.33
4.33	6.00	6.33	5.89	4.22

$$\frac{9+2+3+10+4+11+2+6+11}{9} = 6.44$$

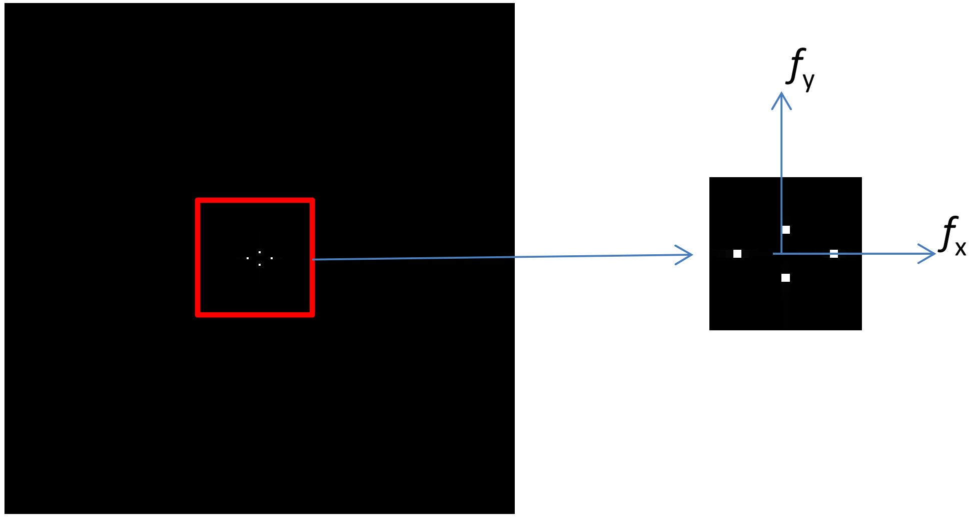
- The size of the above filter is 3×3, but it can be 5×5, 7×7, etc.
- There exist filters for smoothing (as above), edge detection, gradient detection, etc.

Fourier and inverse Fourier transformation

- In the previous slide application of filters in the spatial domain has been described.
- The display of images in the conventional, spatial domain shows pixel intensities at any given spatial location.
- It can be given what spatial frequency characterized the objects in the image, i.e. the image can be converted into the frequency domain.
- This procedure is called Fourier transformation:



Original image (f)



Fourier transformed image (F)

$$F_{k,l} = \sum_{x=1}^{n}\sum_{y=1}^{n} f_{x,y} e^{-2\pi i \frac{kx+ly}{n}} \qquad a_{k,l} = \sum_{x=1}^{n}\sum_{y=1}^{n} f_{x,y} \cos\left(2\pi \frac{kx+ly}{n}\right),\ b_{k,l} = -\sum_{x=1}^{n}\sum_{y=1}^{n} f_{x,y} \sin\left(2\pi \frac{kx+ly}{n}\right)$$

- The original image can be generated from the frequencies:

$$f_{x,y} = \frac{1}{n^2}\sum_{k=1}^{n}\sum_{l=1}^{n} F_{k,l} e^{2\pi i \frac{kx+ly}{n}} \qquad f_{x,y} = \frac{1}{n^2}\sum_{k=1}^{n}\sum_{l=1}^{n}\left(a_{k,l}\cos\left(2\pi\frac{kx+ly}{n}\right) - b_{k,l}\sin\left(2\pi\frac{kx+ly}{n}\right)\right)$$

Filters in the spatial and frequency domains

Filters can be applied in the frequency domain as well:

$$b = f * a$$

b – image generated by the filter
f – the filter
a – the original image
* - application of the filter (convolution).

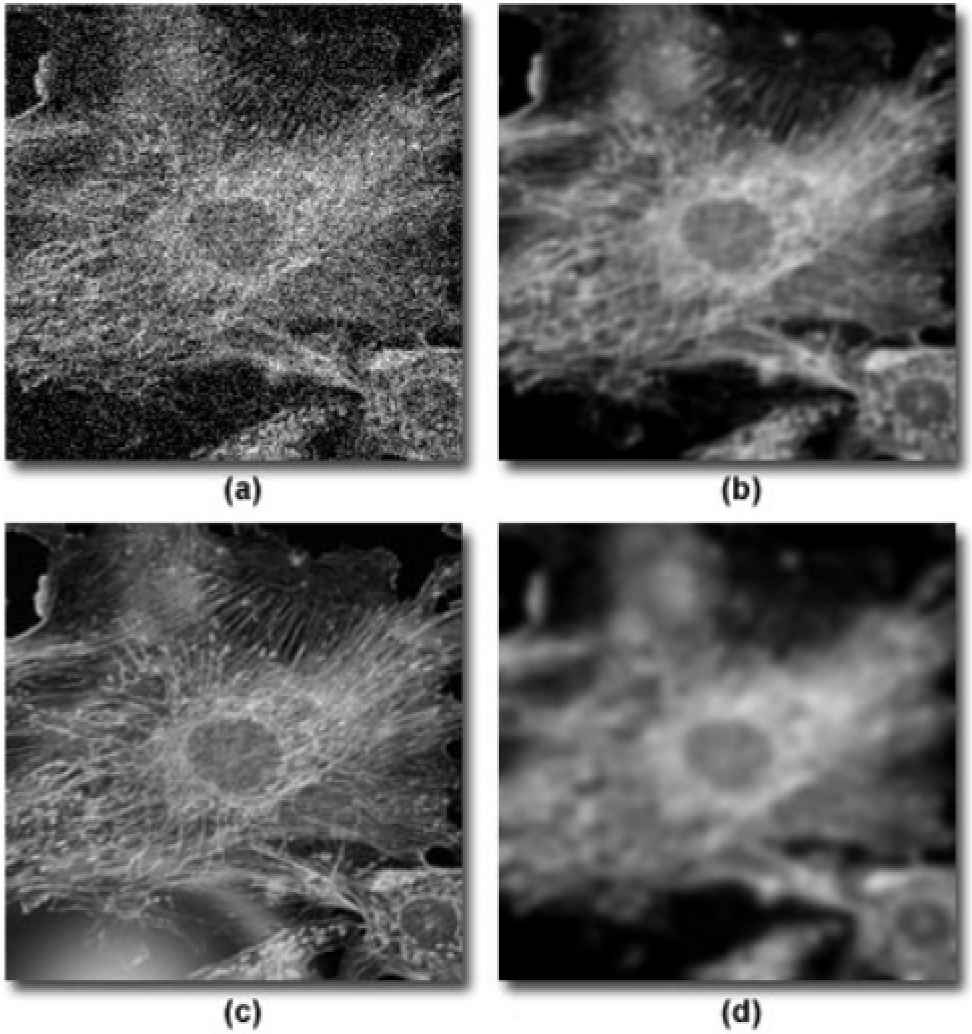
$$B = F \cdot A$$

B – Fourier transform of *b*
F – Fourier transform of the filter
A – Fourier transform of the original image
· - multiplication

$$b = invFT\left(B\right)$$

For many filters application in the frequency domain is advantageous due to faster calculation.

Image degrading factors during digital imaging



a. noise
b. scatter
c. glare
d. blur

ad a:

Noise has two sources:

- statistical nature of photon detection (Poisson distribution)
- imaging system (normal distribution)

ad b:

- Generated in the sample.
- The thicker the sample, the more pronounced it is.

ad c:

- Scatter and reflection generated in the imaging system of the microscope.
- Usually minimal in modern microscopes (if adjusted properly).

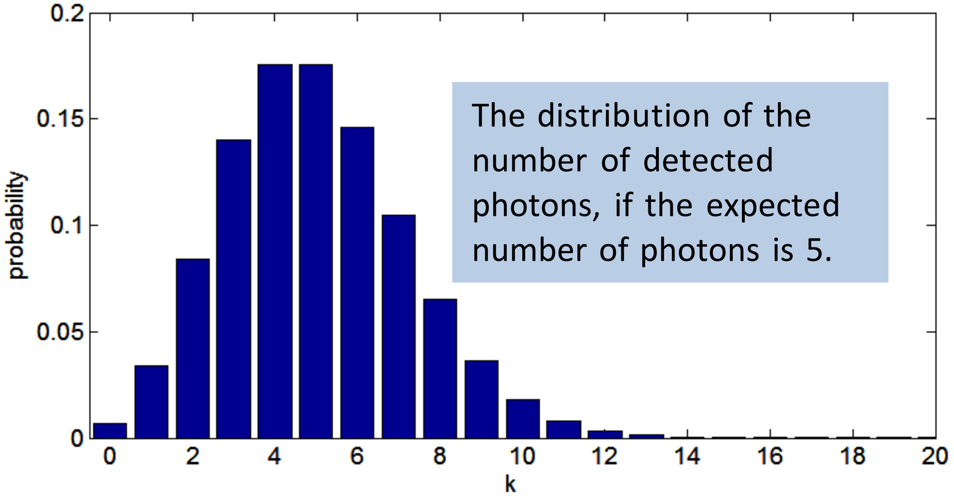
ad d:

- Due to diffraction as a result of the wave nature of light.
- It can be characterized by the PSF (point spread function).

http://www.olympusmicro.com/primer/digitalimaging/deconvolution/deconvolutionhome.html

Noise generated by photon detection

The number of detected photons (*n*) follows a Poisson distribution: $P(n=k)=\frac{N^k}{k!}e^{-N}$



The distribution of the number of detected photons, if the expected number of photons is 5.

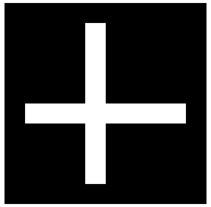
Expected number of photons: N

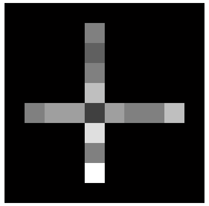
SD of photon number: $\sqrt{N}$

Relative error of photon number: $\frac{\sqrt{N}}{N}=\frac{1}{\sqrt{N}}$

The larger the expected photon number, the smaller the relative error is.

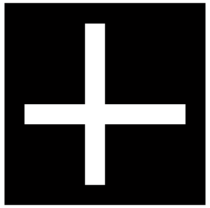
N=5

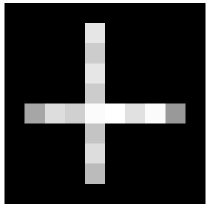




The images are displayed linear stretched, this is why the images on the left look equally bright.

N=50





Original image (without noise)

with Poisson noise added

Image restoration

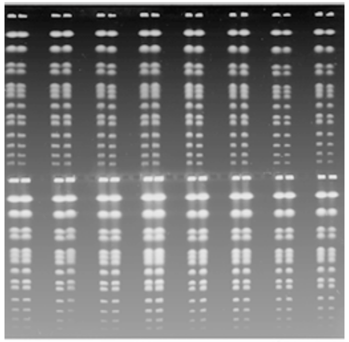
- During the acquisition of microscopic images considerable amount of noise is superimposed on the data.
- The aim of image restoration is to suppress noise for
  - the better representation of the object
  - the better look of the image
- Image quality is usually deteriorated by the following two factors:
  - noise → its effect is decreased by image restoring algorithms (noise reduction)
  - blur → its effect is decreased by deconvolution

Image restoration (suppression of noise)

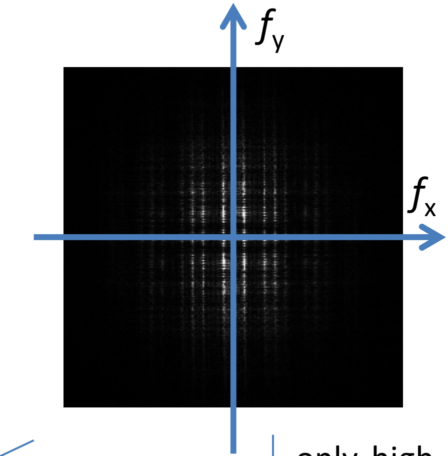
- By modeling the statistical nature of noise it can be removed almost completely, but this approach is rarely used.
- The principle of the most frequently used algorithms:
  - $relative\ error\ due\ to\ noise \sim \frac{1}{\sqrt{N}},\ N$ - average photon number
  - Therefore, it the new value of a pixel is calculated from many pixels
    - the relative extent of noise decreases because the number of photons, from which the intensity is calculated, in increased.
    - but the relative resolution is also decreased, since the intensity of a pixel is calculated from many neighboring pixels.

Low-pass and high-pass filters

Original image



Fourier transformation

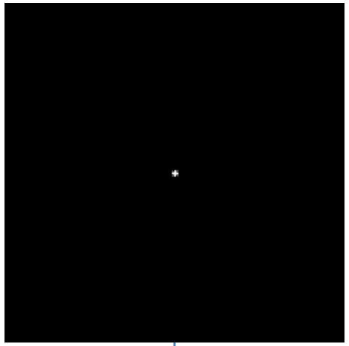


Low-pass filter: IFT is carried out with low frequency components, therefore small objects are removed.

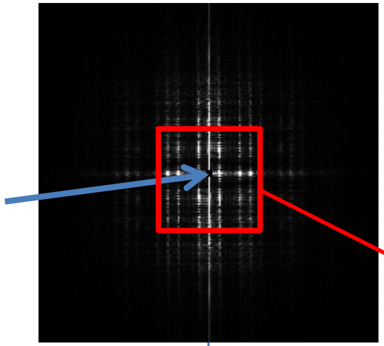
High-pass filter: IFT is carried out with high frequency components, therefore small objects appear without the background.

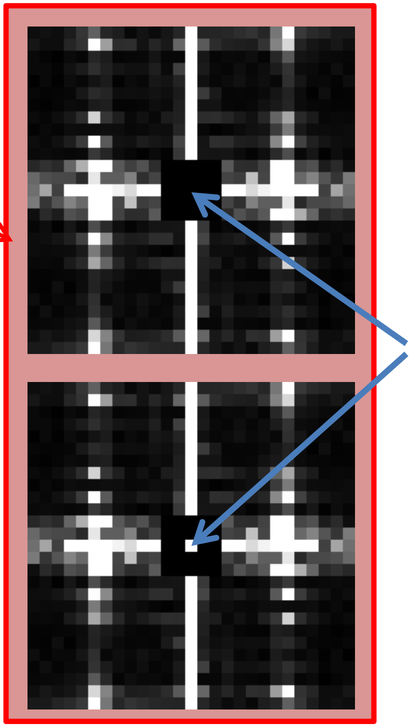
only low frequency components

only high frequency components



low frequency components were suppressed to zero

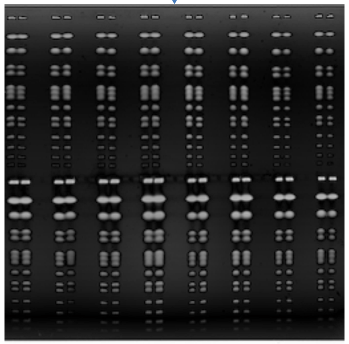




The zero frequency component of a Fourier transformed is the mean of the image, therefore it is often retained in high-pass filters so that the mean intensity of the image is not changed.

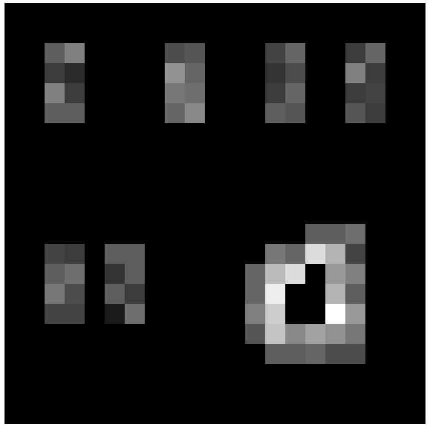
inverse Fourier transformation

low-pass filter

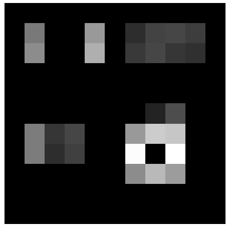


high-pass filter

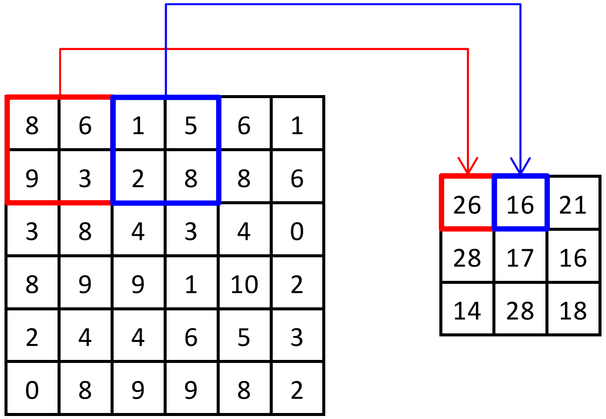
Image restoration, noise reduction: pixel pooling



Original image



Smoothed image

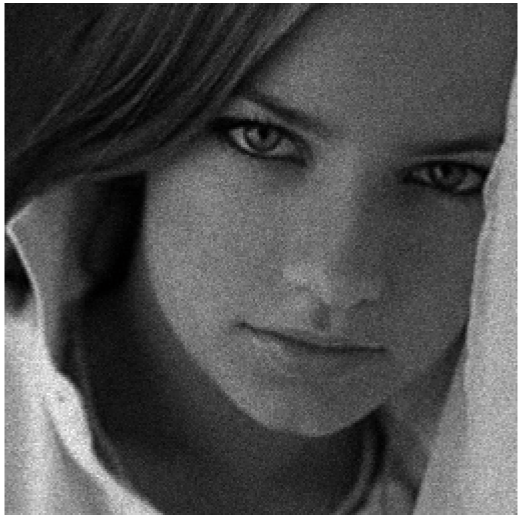


8	6	1	5	6	1
9	3	2	8	8	6
3	8	4	3	4	0
8	9	9	1	10	2
2	4	4	6	5	3
0	8	9	9	8	2

Original image

26	16	21
28	17	16
14	28	18

Smoothed image



Original image

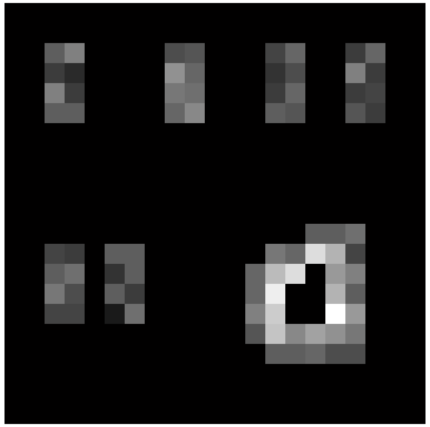
Smoothed image

- In the procedure the intensity of every 2×2 block in the original image is summed (pooled) and this summed intensity will be entered into the smoothed image.
- The size of the smoothed image will be half of the original.
- The block size can be different.

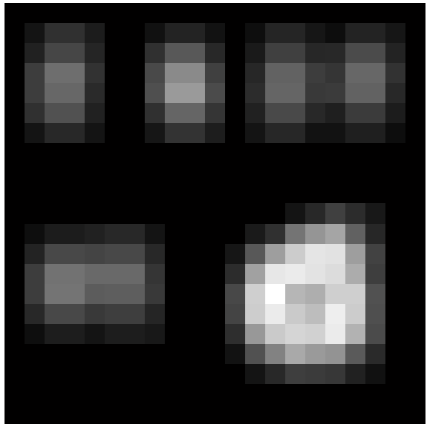
Image restoration, noise reduction: mean (average) filter

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

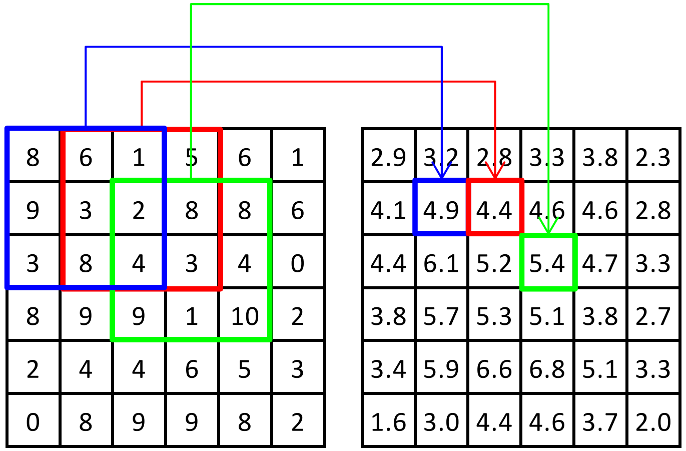
A filter implementing 3×3 averaging.



Original image



Smoothed image

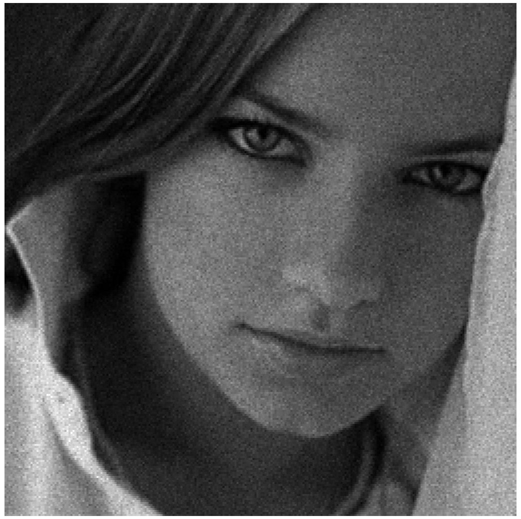


8	6	1	5	6	1
9	3	2	8	8	6
3	8	4	3	4	0
8	9	9	1	10	2
2	4	4	6	5	3
0	8	9	9	8	2

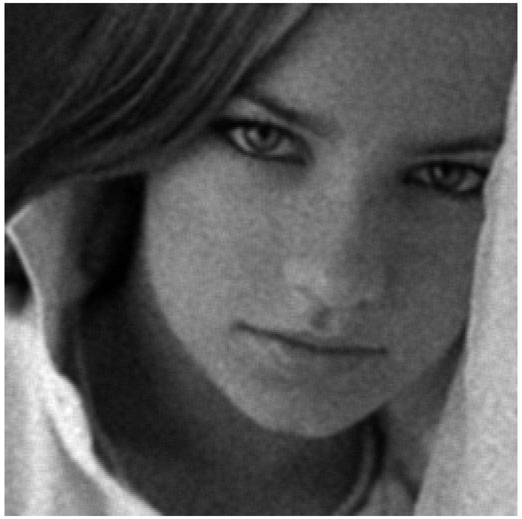
Original image

2.9	3.2	2.8	3.3	3.8	2.3
4.1	4.9	4.4	4.6	4.6	2.8
4.4	6.1	5.2	5.4	4.7	3.3
3.8	5.7	5.3	5.1	3.8	2.7
3.4	5.9	6.6	6.8	5.1	3.3
1.6	3.0	4.4	4.6	3.7	2.0

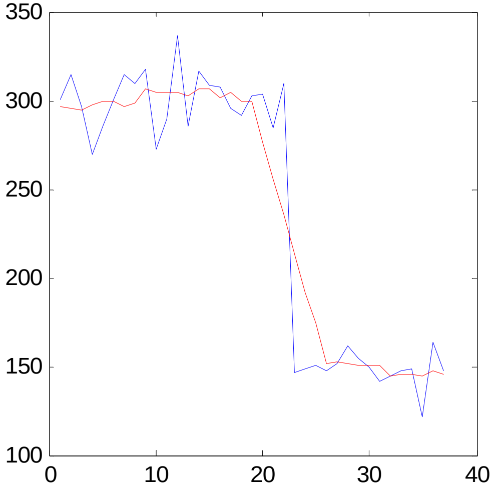
Smoothed image



Original image



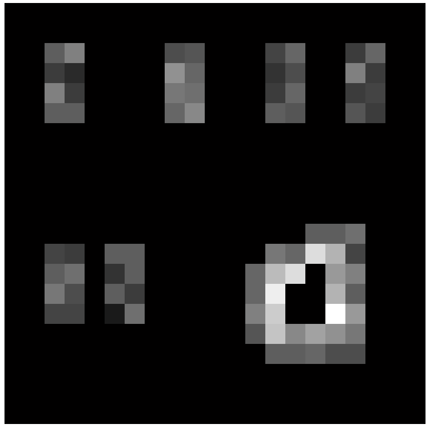
Smoothed image



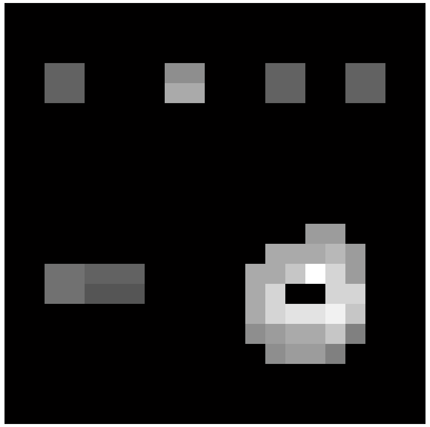
Original, mean filtered (window (kernel) size of 7)

- Each pixel is replaced by the mean of the neighboring pixels (3×3 pixels in the image above).
- The size of the smoothed image is the same as that of the original (as opposed to pixel pooling).
- Disadvantage: edges are suppressed.

Image restoration, noise reduction: median filter

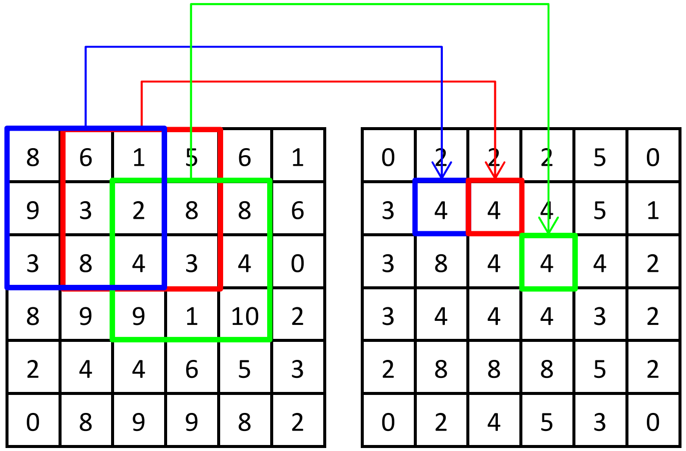


Original image



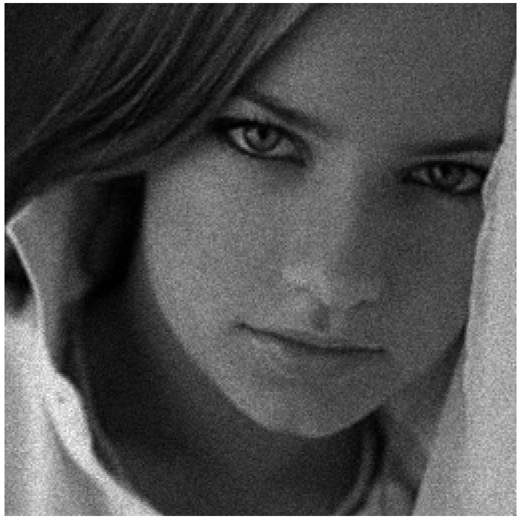
Smoothed image

The median of the given sample of 9 elements.

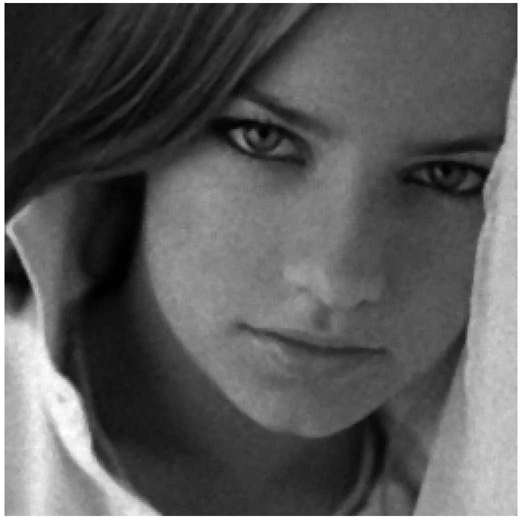


8	6	1	5	6	1
9	3	2	8	8	6
3	8	4	3	4	0
8	9	9	1	10	2
2	4	4	6	5	3
0	8	9	9	8	2

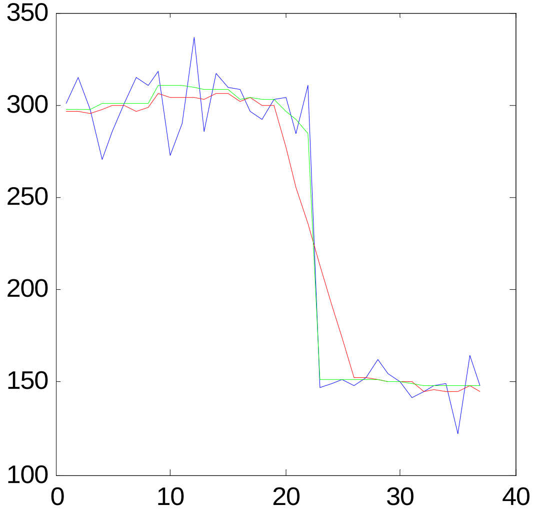
0	2	2	2	5	0
3	4	4	4	5	1
3	8	4	4	4	2
3	4	4	4	3	2
2	8	8	8	5	2
0	2	4	5	3	0



Original image



Smoothed image

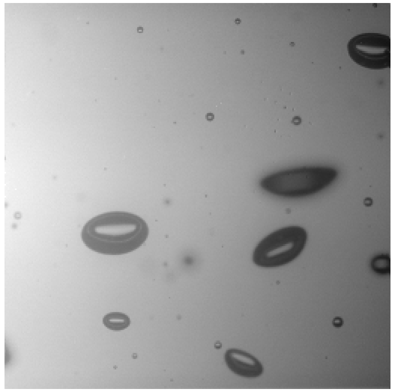


Original, mean filter (kernel size of 7), median filter (kernel size of 7)

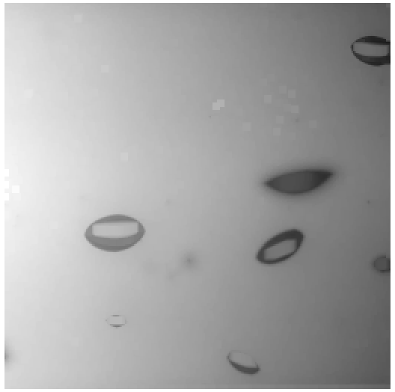
- Each pixel is replaced by the median of the neighboring pixels (3×3 pixels above).
- Edges are preserved much better than by the mean filter.

Image restoration, noise reduction: max-min (min of max) filter

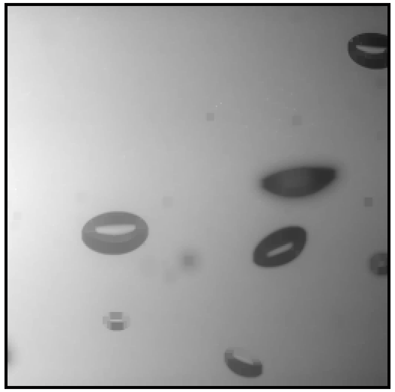
Original



Max (filter size: 5)



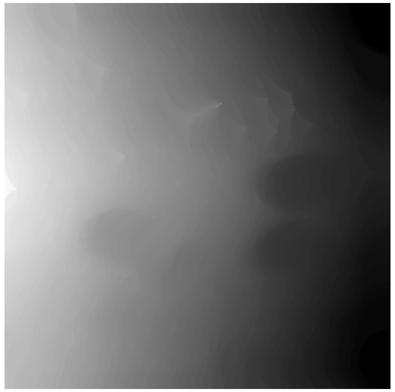
Min of Max (filter size: 5)



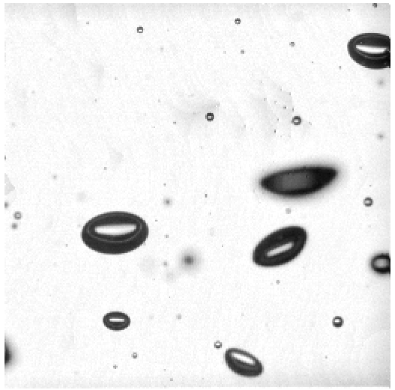
1. Maximum filtering: each pixel is replaced by the maximum of a neighborhood of given size thereby eliminating objects smaller than the filter size.
2. Minimum filtering is carried out on the output of the previous step shifting edges back to their original position.

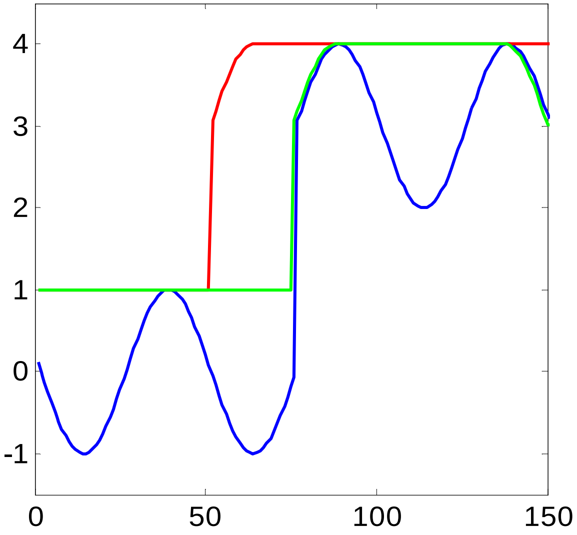
The max-min filter can be used for background subtraction:

Min of Max (filter size: 50)



Original– minOfMax(50)



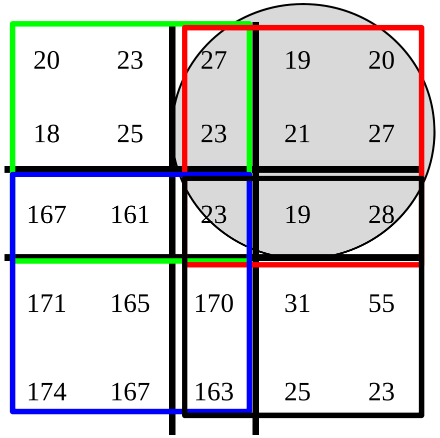


Original signal

Max

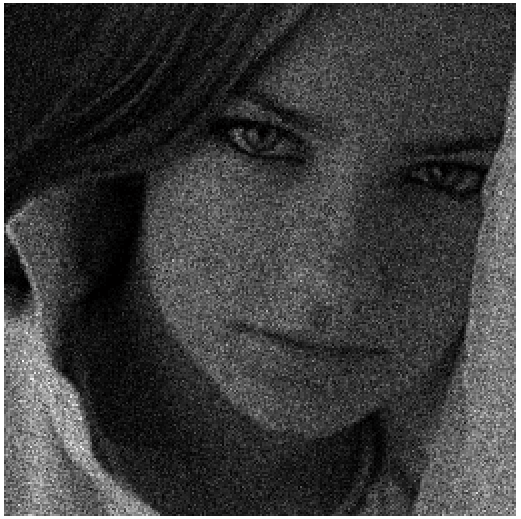
Min of Max

Image restoration, noise reduction: Kuwahara filter



20	23	27	19	20
18	25	23	21	27
167	161	23	19	28
171	165	170	31	55
174	167	163	25	23

The mean of the subwindow with smallest variance.



Kuwahara

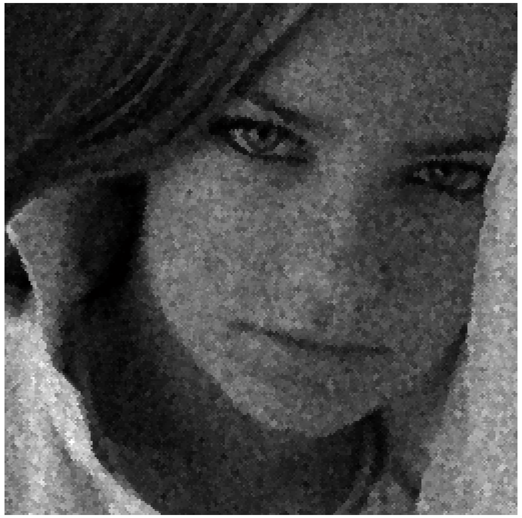
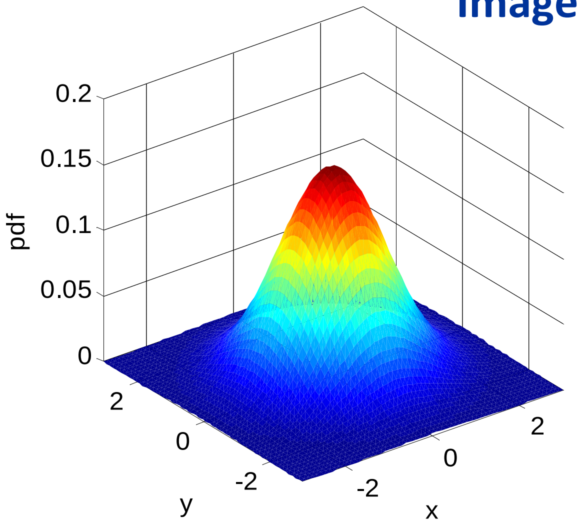
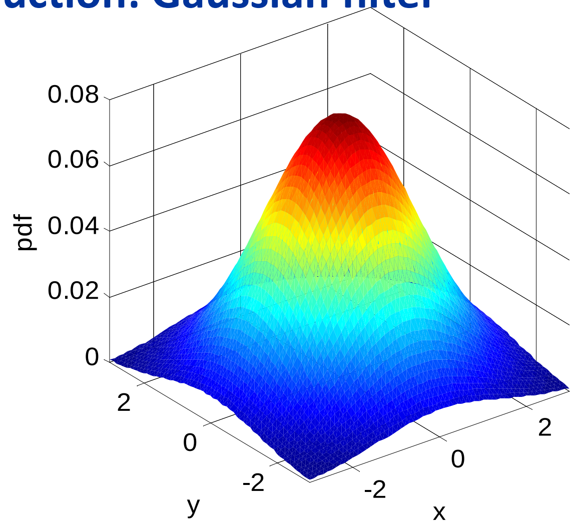


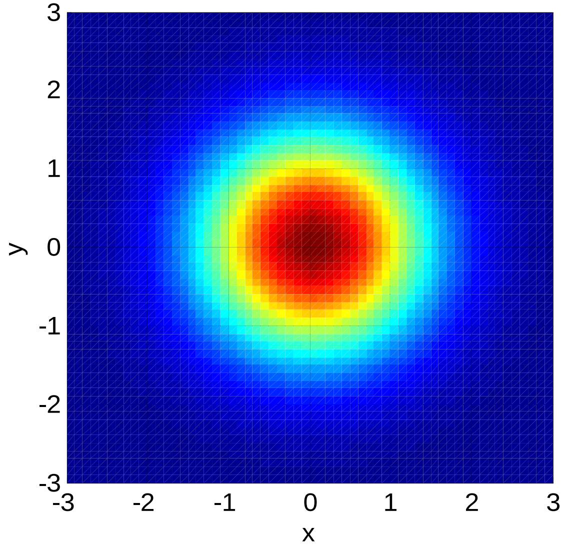
Image restoration, noise reduction: Gaussian filter



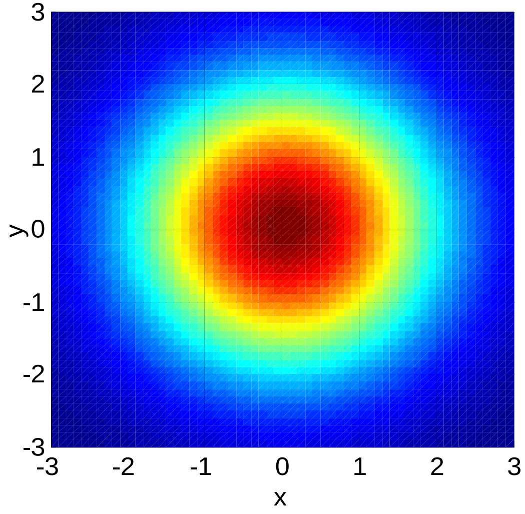
$$\mathrm{cov} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$



$$\mathrm{cov} = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$$



0.075	0.124	0.075
0.124	0.204	0.124
0.075	0.124	0.075

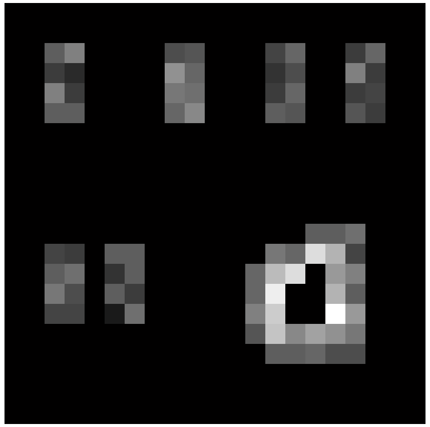


0.102	0.115	0.102
0.115	0.131	0.115
0.102	0.115	0.102

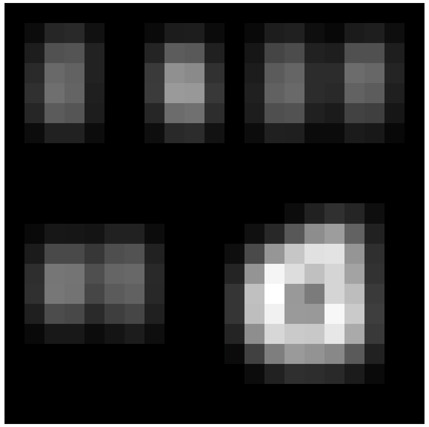
Digital version of the Gaussian which is used as a filter.

- Gaussian filters have two adjustable parameters:
  - filter size (how many pixels wide)
  - variance of the normal distribution
- Since the Fourier transformed of a Gaussian function is also a Gaussian, Gaussian filtering is a low-pass filter.

Image restoration, noise reduction: Gaussian filter



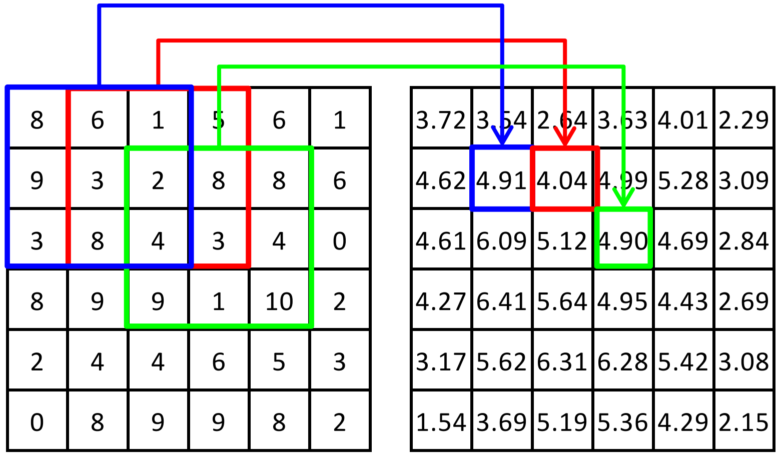
Original image



Smoothed image

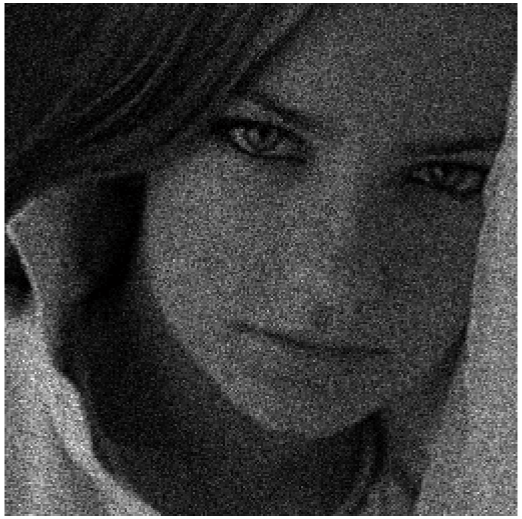
$$\mathrm{cov} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

0.075	0.124	0.075
0.124	0.204	0.124
0.075	0.124	0.075



8	6	1	5	6	1
9	3	2	8	8	6
3	8	4	3	4	0
8	9	9	1	10	2
2	4	4	6	5	3
0	8	9	9	8	2

3.72	3.54	2.64	3.63	4.01	2.29
4.62	4.91	4.04	4.99	5.28	3.09
4.61	6.09	5.12	4.90	4.69	2.84
4.27	6.41	5.64	4.95	4.43	2.69
3.17	5.62	6.31	6.28	5.42	3.08
1.54	3.69	5.19	5.36	4.29	2.15

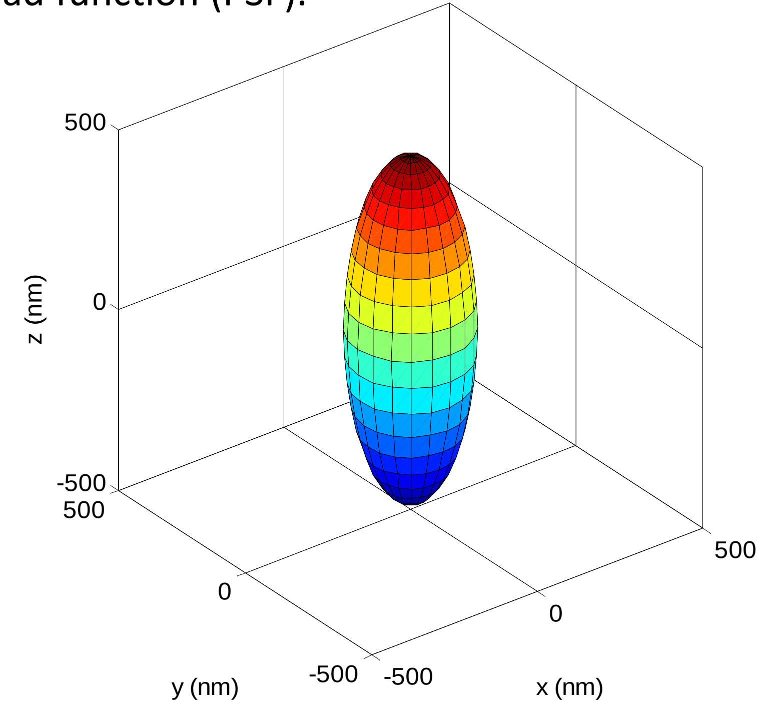


Original image

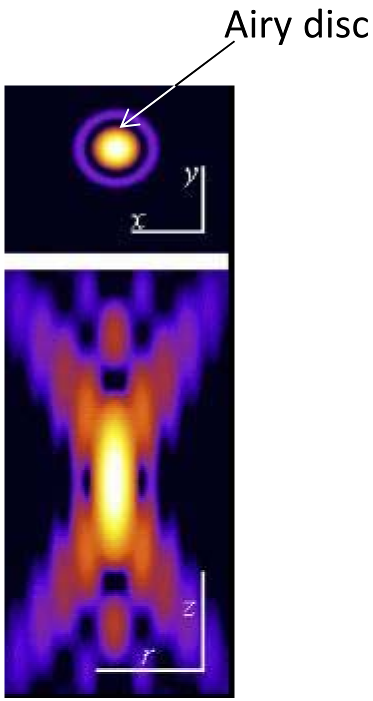
Smoothed image

The point spread function (PSF)

- The image of a point-like object will not be a point even in a perfectly functioning optical microscope due to the wave nature of light and the fact that only a fraction of emitted photons are detected.
- The image of a point-like object blurred as a result of the aforementioned factors is called the point spread function (PSF).



The PSD in three dimensions



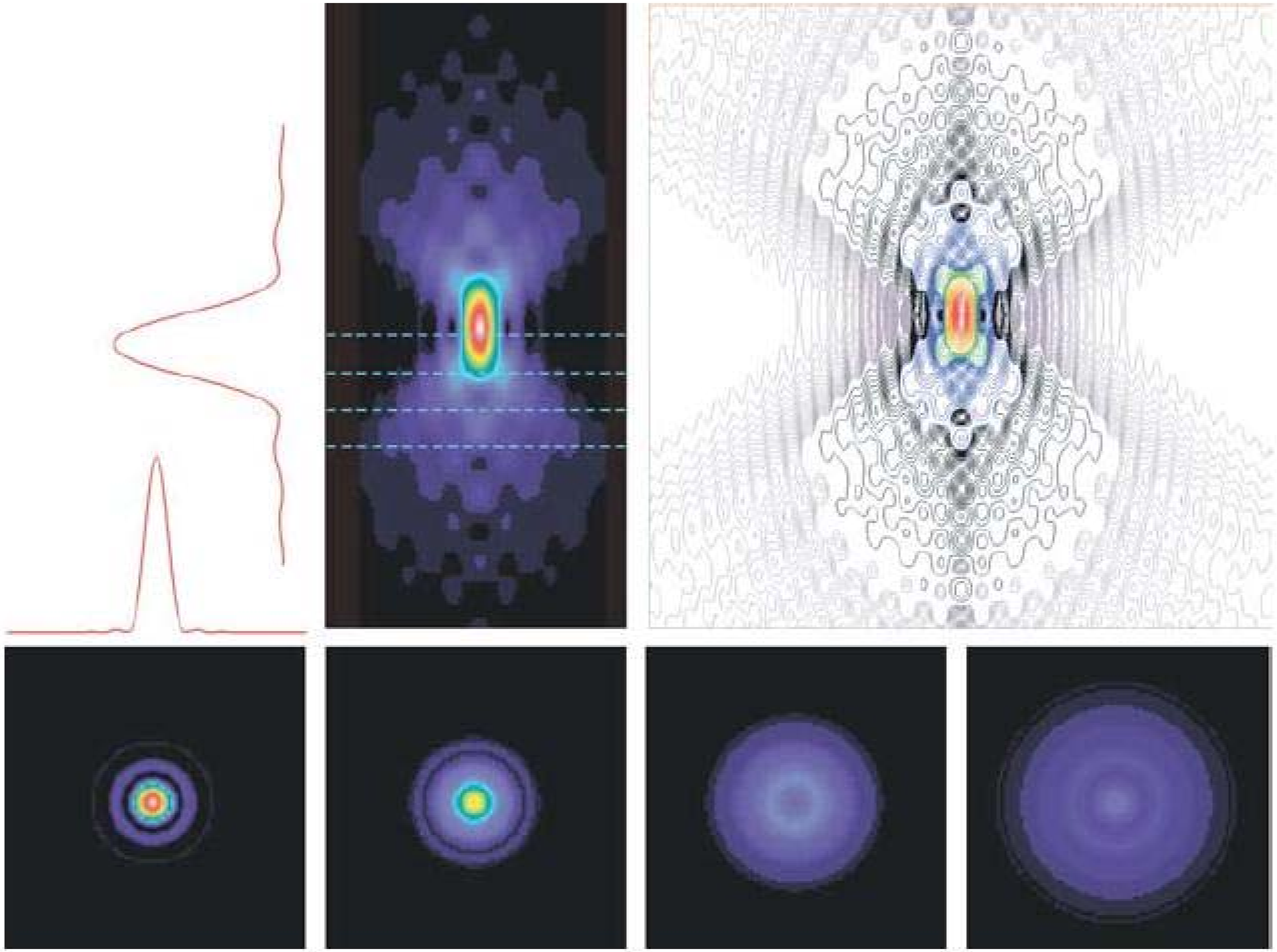
Two dimensional projections of the PSD

$$d_{lateral} = \frac{0.4\lambda}{NA} \qquad d_{axial} = \frac{1.4\lambda n}{NA^2}$$

λ – wavelength, *NA* – numerical aperture, *n* – index of refraction of the medium between the object and the objective

http://www.olympusmicro.com/primer/digitalimaging/deconvolution/deconvolutionhome.html

The PSF in three dimensions



Sibarita, J. B. 2005. Deconvolution microscopy. Advances in biochemical engineering/biotechnology 95:201-243.

The PSF

The PSF in an *x*-*y* plane can be adequately approximated by the following Bessel function:

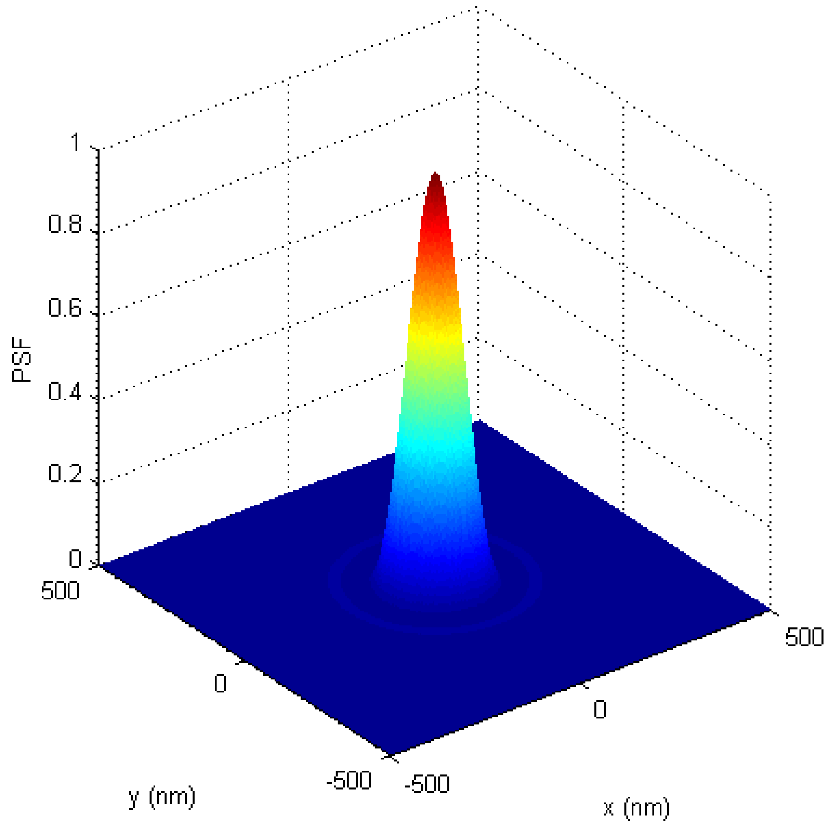
$$PSF = \left(\frac{2J_1\left(\frac{2\pi}{\lambda} r \tan\omega\right)}{\frac{2\pi}{\lambda} r \tan\omega} \right)^2$$

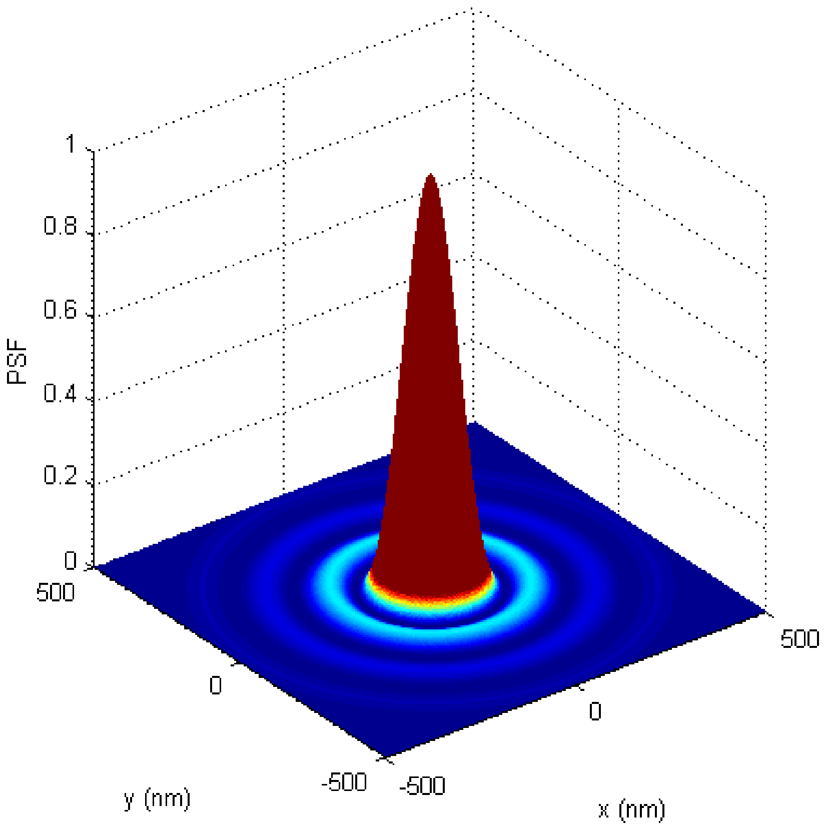
λ – wavelength

r – distance from the optical axis

ω – half-angle of the objective

J_1 – Bessel function of the first kind, of order 1

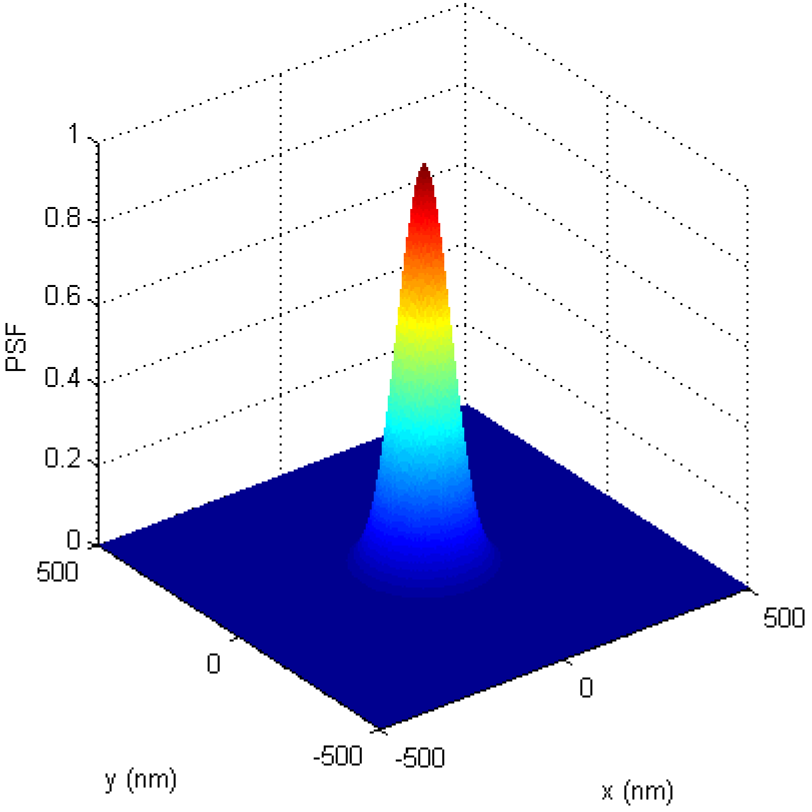




The PSF

The central maximum (Airy disc) of the PSF in an *x*-*y* plane can be approximated by a Gaussian:

$$PSF = e^{-2.5\left(\frac{2}{\lambda} r \tan\omega\right)^2}$$



Interpretation of optical resolution by convolution

Object

3	9	4	9	8	6	6	7
8	6	1	9	4	1	7	2
4	5	2	5	2	2	6	4
9	1	1	5	4	4	5	6
2	9	2	3	1	8	5	8
3	6	2	9	1	0	3	1
1	4	4	4	9	0	7	9
1	5	0	1	10	2	2	8

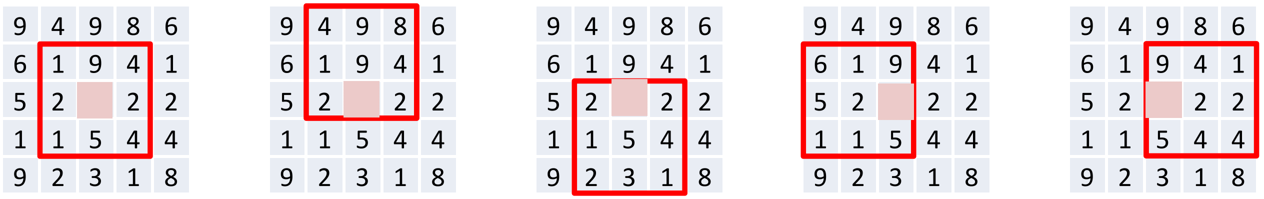
PSF (convolution kernel)

0.000	0.167	0.000
0.167	0.333	0.167
0.000	0.167	0.000

Image

0.000	0.500	1.500	0.667	1.500	1.333	1.000	1.000	1.167	0.000
0.500	3.833	5.167	4.500	6.500	5.833	4.500	5.333	3.667	1.167
1.333	4.833	5.833	3.833	6.167	4.667	3.500	4.833	3.667	0.333
0.667	5.000	3.833	2.667	4.667	3.167	2.833	5.000	3.667	0.667
1.500	4.167	4.333	2.000	3.833	3.333	4.500	5.167	4.833	1.000
0.333	4.167	4.833	3.167	3.833	3.000	4.333	5.667	4.667	1.333
0.500	2.500	5.000	4.167	4.667	3.500	2.000	3.167	3.667	0.167
0.167	1.667	4.000	3.000	5.167	5.500	3.000	4.667	5.667	1.500
0.167	1.333	2.500	1.667	2.667	5.333	2.667	3.500	4.500	1.333
0.000	0.167	0.833	0.000	0.167	1.667	0.333	0.333	1.333	0.000

- The image is generated by the convolving the object with the PSF.
- The image of the pixel with intensity 5 is determined by the pixel itself and its neighborhood:



9	4	9	8	6
6	1	9	4	1
5	2		2	2
1	1	5	4	4
9	2	3	1	8

9	4	9	8	6
6	1	9	4	1
5	2		2	2
1	1	5	4	4
9	2	3	1	8

9	4	9	8	6
6	1	9	4	1
5	2		2	2
1	1	5	4	4
9	2	3	1	8

9	4	9	8	6
6	1	9	4	1
5	2		2	2
1	1	5	4	4
9	2	3	1	8

9	4	9	8	6
6	1	9	4	1
5	2		2	2
1	1	5	4	4
9	2	3	1	8

$$5\cdot 0.33+2\cdot 0.167+2\cdot 0.167+9\cdot 0.167+5\cdot 0.167=4.67$$

$$image = object * PSF$$

$$image\left(x,y\right)=\sum_{i=-\infty}^{\infty}\sum_{j=-\infty}^{\infty} object\left(i,j\right)PSF\left(x-i,y-j\right)$$

Convolution-deconvolution

Object

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

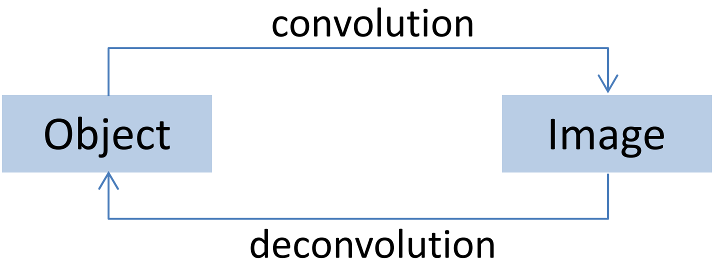
PSF (convolution kernel)

0.000	0.167	0.000
0.167	0.333	0.167
0.000	0.167	0.000

Image

0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
0.000	0.000	0.000	0.000	0.167	0.000	0.000	0.000	0.000	0.000
0.000	0.000	0.000	0.167	0.333	0.167	0.000	0.000	0.000	0.000
0.000	0.000	0.000	0.000	0.167	0.000	0.000	0.000	0.000	0.000
0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

- The fact that the image of a point-like object is blurred and is equivalent to the PSF can be interpreted according to the convolution theorem.
- The object can be restored from the image by deconvolution assuming the *PSF is known*.



Convolution-deconvolution

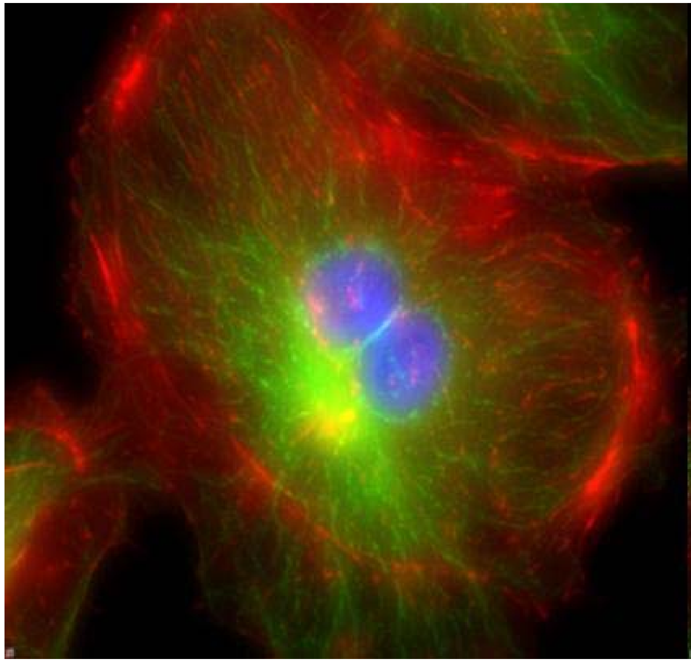
- Deconvolution in the spatial domain cannot be given as a closed-form expression.
- Both convolution and deconvolution can be carried out in the frequency domain:

	convolution	deconvolution
spatial domain	$object * PSF = image$	–
frequency domain	$\mathrm{FT}\left(object\right)\cdot\mathrm{FT}\left(PSF\right)=\mathrm{FT}\left(image\right)$	$\mathrm{FT}\left(image\right)./\mathrm{FT}\left(PSF\right)=\mathrm{FT}\left(object\right)$
	IFT $\downarrow$ $image$	IFT $\downarrow$ $object$

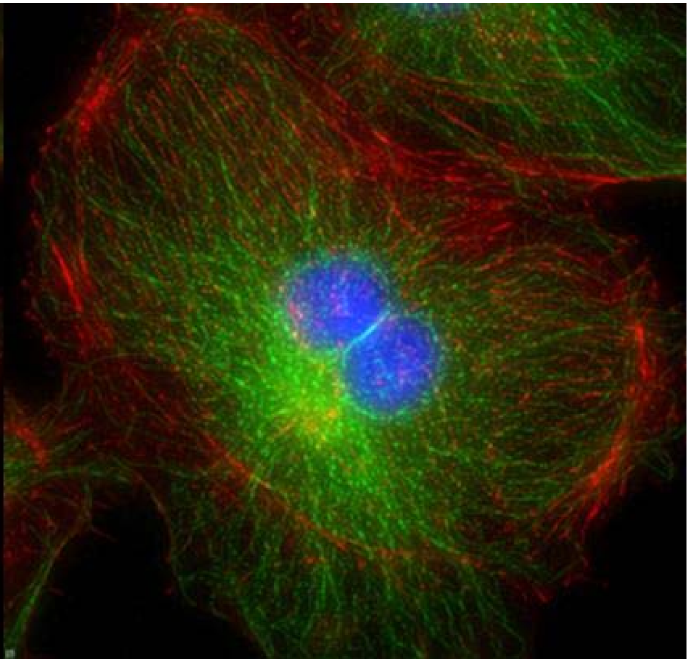
FT – Fourier transformation
IFT – inverse Fourier transformation
./ – element-wise division
. – element-wise multiplication („.*” in Matlab)

Deconvolution in three dimensions

- Since not only points in the given plane, but also above and below it contribute to an image point, deconvolution makes real sense only in 3D.
- In this case the image is generated by the 3D convolution of the object with the 3D PSF.
- Therefore, the object can be reconstructed by the 3D deconvolution of the image with the 3D PSF.
- The procedure can be applied to 3D image stacks recorded by conventional (non-confocal) microscopes → contrast is significantly, resolution is slightly increased.
- Determination of the 3D PSF is of central importance. Usually it is achieved by measuring the 3D image of fluorescent bead if the bead considered to be point-like.



Original confocal image



after deconvolution

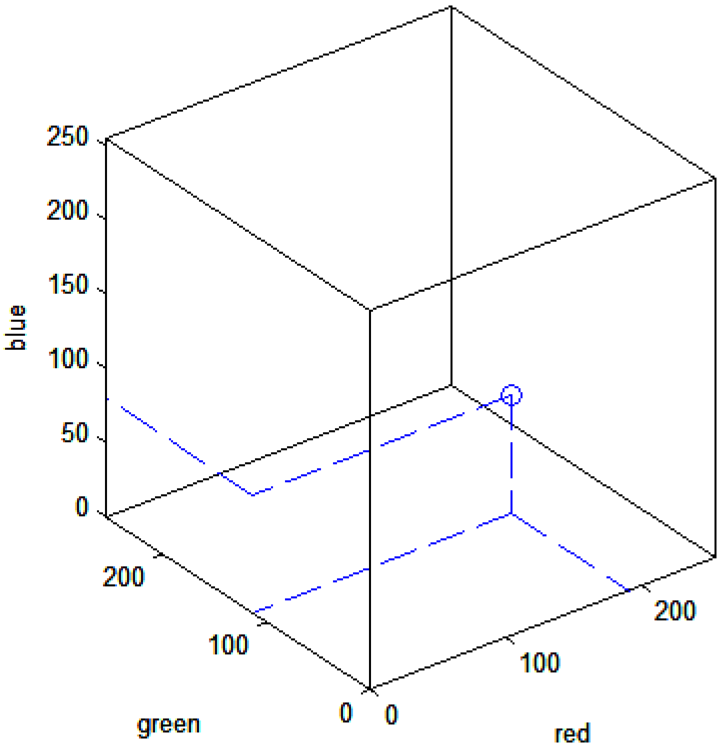
Color images

- There are three different color-sensitive receptors cells (cones) in the human eye which are sensitive for the red, green and blue spectral range.
- Therefore, color can be imagined as a function of three variables or a three dimensional space:

RGB color space:

- R: intensity of red [0-255] or [0-1]
- G: intensity of green [0-255] or [0-1]
- B: intensity of blue [0-255] or [0-1]

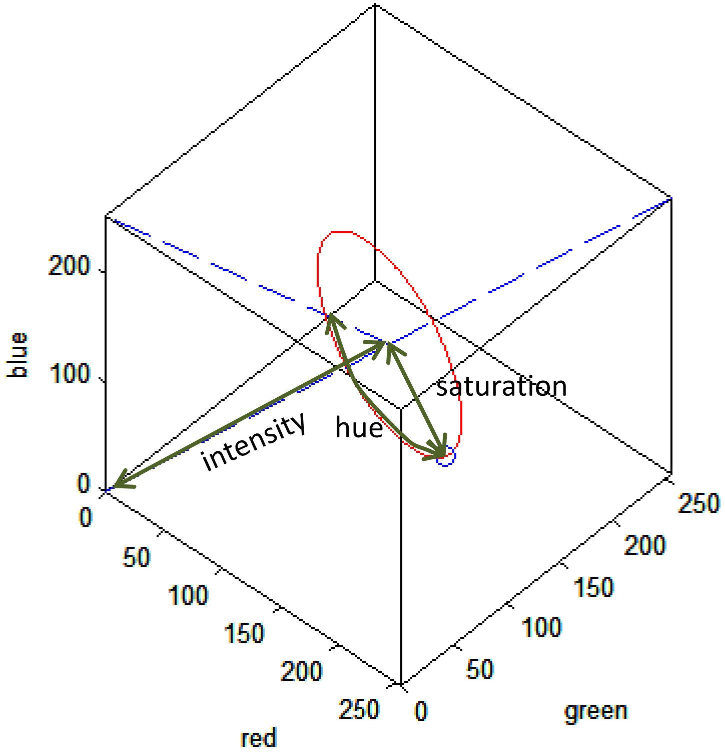
The RGB color model is used in most cases for the interpretation of color images.



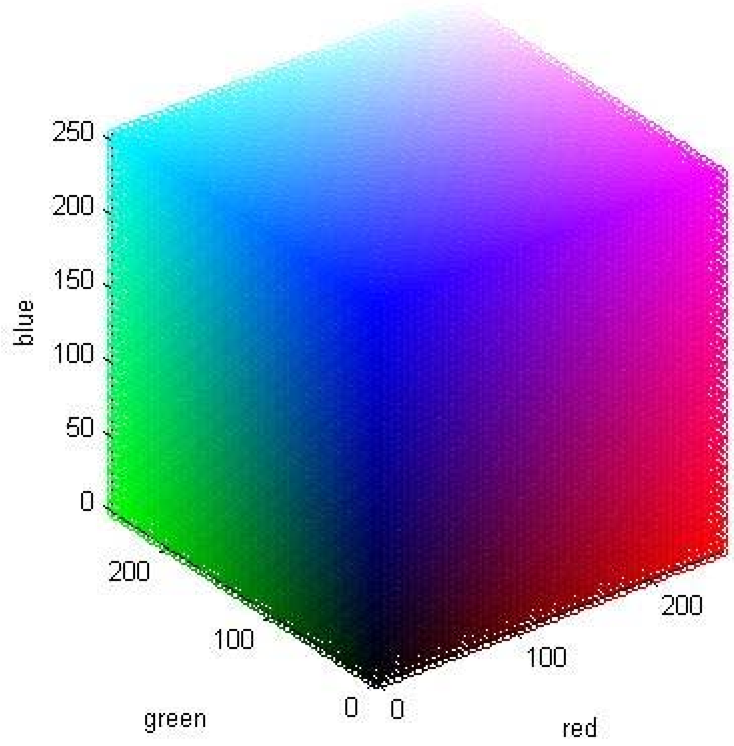
HSI color space (hue, saturation, intensity):

A circle is drawn perpendicular to the diagonal of the cube. The (R,G,B) point is on the circle.

- H: the angle made between the radius drawn to (R,G,B) and the line drawn from the origin of the circle to the blue corner
- S: the ratio of the radius of the circle to the line drawn from the circle origin through the (R,G,B) point to the edge of the cube (how far the (R,G,B) point is from the diagonal)
- I: the distance of the origin of the circle from (0,0,0)

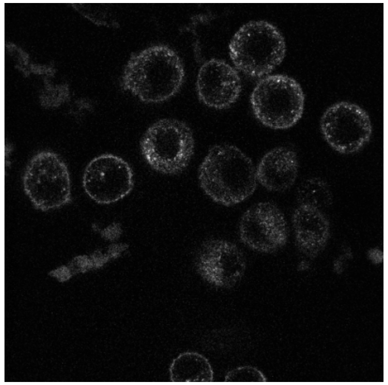


The RGB color space

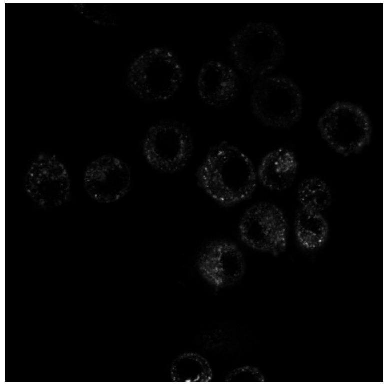


Color images, generation of color composite images

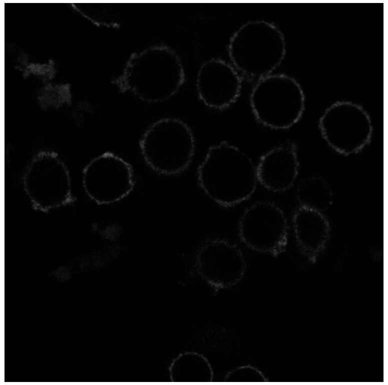
- During the investigation of biological samples the acquired images are usually gray-scale images, therefore color images arise only when images recorded in different channels (different wavelength ranges) are overlaid on each other.
- Original gray-scale images: Color image
  - first wavelength range → red channel
  - second wavelength range → green channel
  - third wavelength range → blue channel
- For better visibility different colors can be used, e.g. due to the bad visibility of blue on black cyan is often used instead of blue.
- It is advisable to contrast stretch the individual channels in order to increase the intensities to be emphasized to 255.



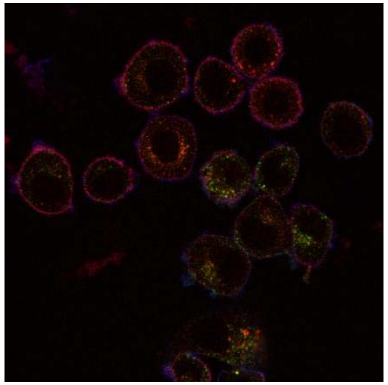
First channel → red



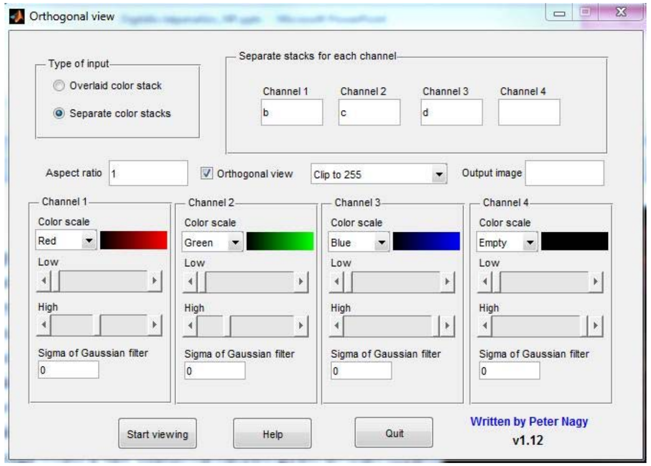
Second channel → green



Third channel → blue



Color composite image without contrast stretching



Orthogonal view

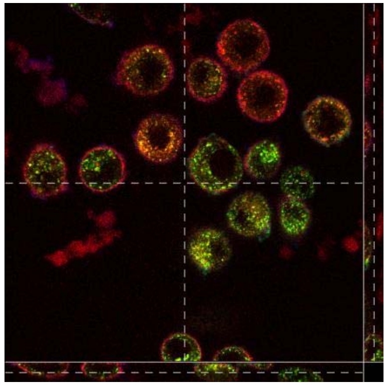
Type of input
○ Overlaid color stack
● Separate color stacks

Separate stacks for each channel
Channel 1: b
Channel 2: c
Channel 3: d
Channel 4:

Aspect ratio: 1
☑ Orthogonal view
Clip to 255
Output image:

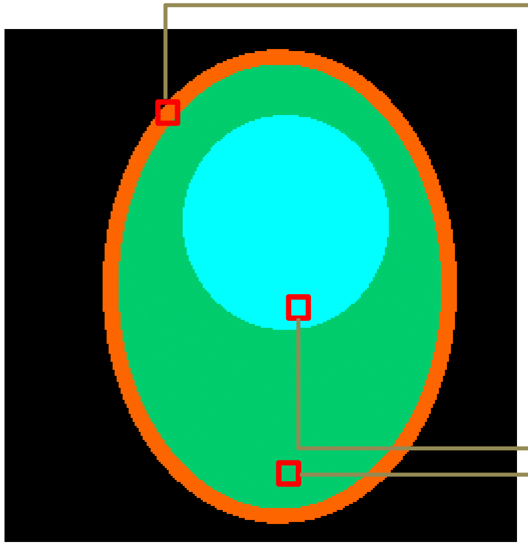
Channel 1 – Color scale: Red; Sigma of Gaussian filter: 0
Channel 2 – Color scale: Green; Sigma of Gaussian filter: 0
Channel 3 – Color scale: Blue; Sigma of Gaussian filter: 0
Channel 4 – Color scale: Empty; Sigma of Gaussian filter: 0

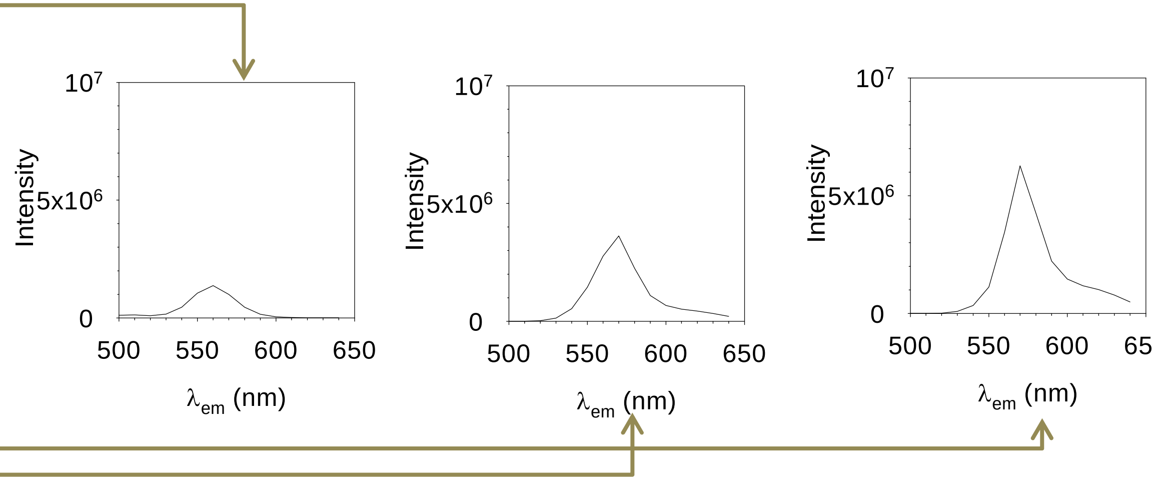
Color composite image after contrast stretching the red and green channels:



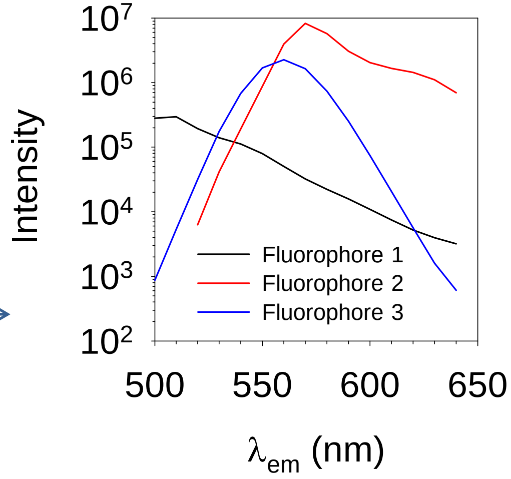
Spectral unmixing

- It is possible that the distribution of fluorophores cannot be determined based on the images acquired in different fluorescent channels if
  - the sample is labeled with dyes with overlapping spectra
  - different fluorophores are mixed in single pixels





- The spectrum, typically the emission spectrum, is recorded in each pixel of the sample.
- The sample has been labeled with three different fluorescent dyes whose spectra can be determined: →
- The fluorescence intensity in individual pixels is expressed by the following expression:



$$I(\lambda_1) = F(1)s(1,\lambda_1) + F(2)s(2,\lambda_1) + F(3)s(3,\lambda_1)$$

$F(1)$, $F(2)$, $F(3)$ – the relative quantity of individual dyes in a pixel

$s(1,\lambda_1)$, $s(2,\lambda_1)$, $s(3,\lambda_1)$ – the spectrum of individual dyes at a given wavelength

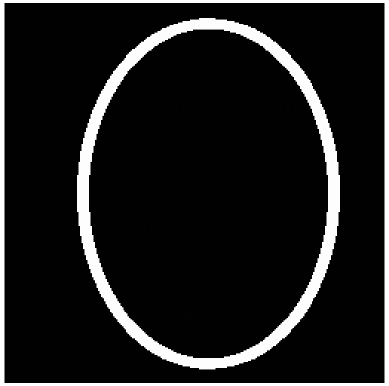
Spectral unmixing

- The previous set of equation in matrix form:

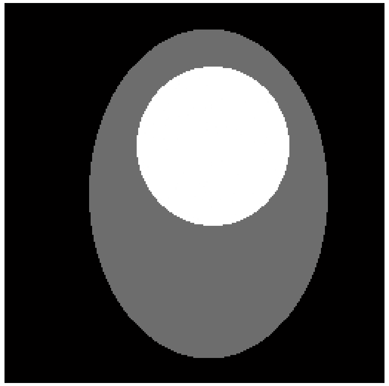
$$\begin{bmatrix} s(1,\lambda_1) & s(2,\lambda_1) & \dots & s(k,\lambda_1) \\ s(1,\lambda_2) & s(2,\lambda_2) & \dots & s(k,\lambda_2) \\ \dots & \dots & \dots & \dots \\ s(1,\lambda_n) & s(2,\lambda_n) & \dots & s(k,\lambda_n) \end{bmatrix} \begin{bmatrix} F(1) \\ F(2) \\ \dots \\ F(k) \end{bmatrix} = \begin{bmatrix} I(\lambda_1) \\ I(\lambda_2) \\ \dots \\ I(\lambda_n) \end{bmatrix}$$

- The above equation set is overdetermined, since there are *k* (in the above case 3) independent variables for *n* (in the above case 15) equations, therefore on approximate solution can be found with fitting.
- As a result the distribution of each dye can be determined:

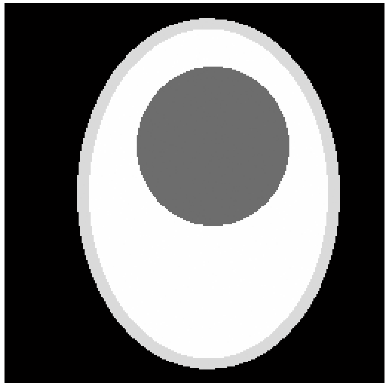
The distribution of fluorophore 1



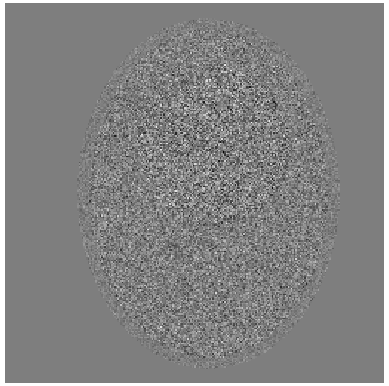
The distribution of fluorophore 2



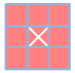
The distribution of fluorophore 3

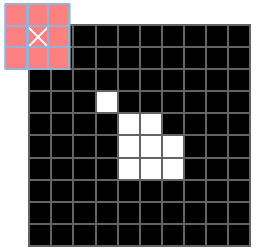


Error (residual) of the fitting



Mathematical morphology, morphological image processing: basic concepts

- Morphology: The collection of such tools which can be used for the description and representation of the shape and boundary (morphology) of objects in an image.
- Another image (set) has to be defined with which the operation is carried out. This is called the structuring element (S.E.) or kernel, e.g.: 
- The S.E. has a reference point (white ×) with which its position is defined.
- The S.E. is placed on every pixel of the image to be analyzed and an operation determined by the given operator is carried out.

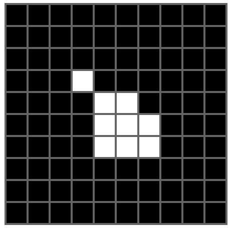


Morphological image processing: dilation

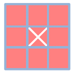
Binary dilation (⊕): The collection of those pixels on which the reference pixel of the structuring element is placed producing an overlap between the structuring element and the original object.

$$A \oplus B = \bigcup_{(x,y)\in A} \hat{B}_{x,y} = \left\{(x,y) \,\middle|\, \hat{B}_{x,y} \cap A \neq \varnothing\right\}$$

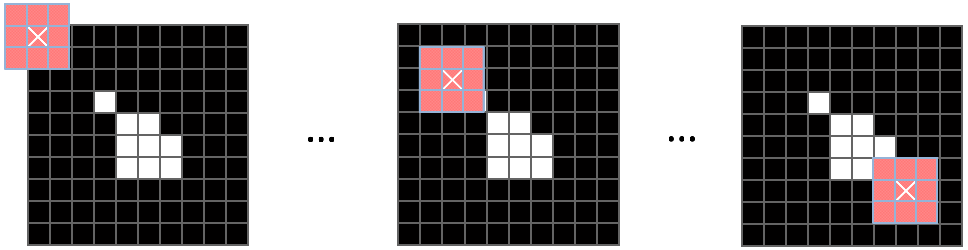
Original image (A):



S.E. (B):

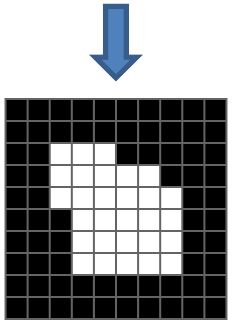


Dilation step-by-step:

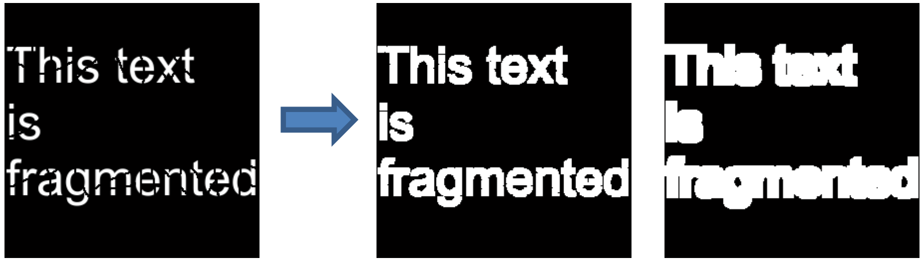


Overlap?	no	yes	yes
Part of $A \oplus B$?	no	yes	yes

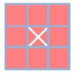
$A \oplus B =$

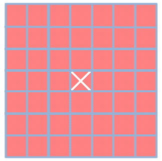


Holes smaller than the structuring element are removed:



Structuring element (not drawn to scale with the image):



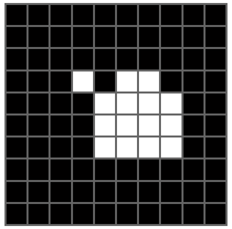


Morphological image processing: erosion

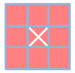
Binary erosion (⊖): The collection of those pixels on which the reference pixel of the structural element is place resulting in all the pixels of the structural element being inside the object.

$$A \ominus B = \left\{ (x, y) \middle| B_{x,y} \subset A \right\}$$

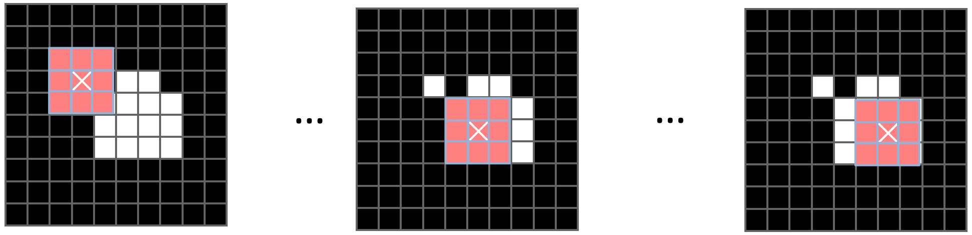
Original image (*A*):



S.E. (*B*):

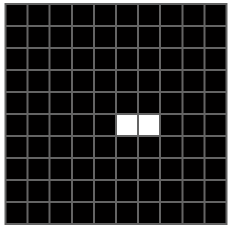


Erosion step-by-step:

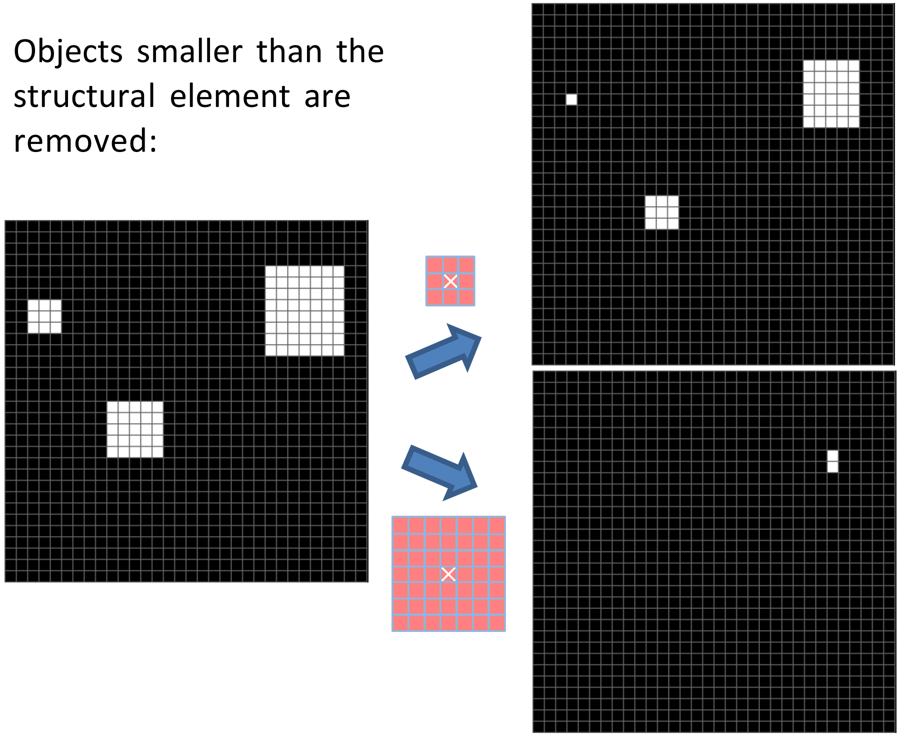


$B \subset A$?	no	yes	yes
Part of $A \ominus B$?	no	yes	yes

$A \ominus B =$



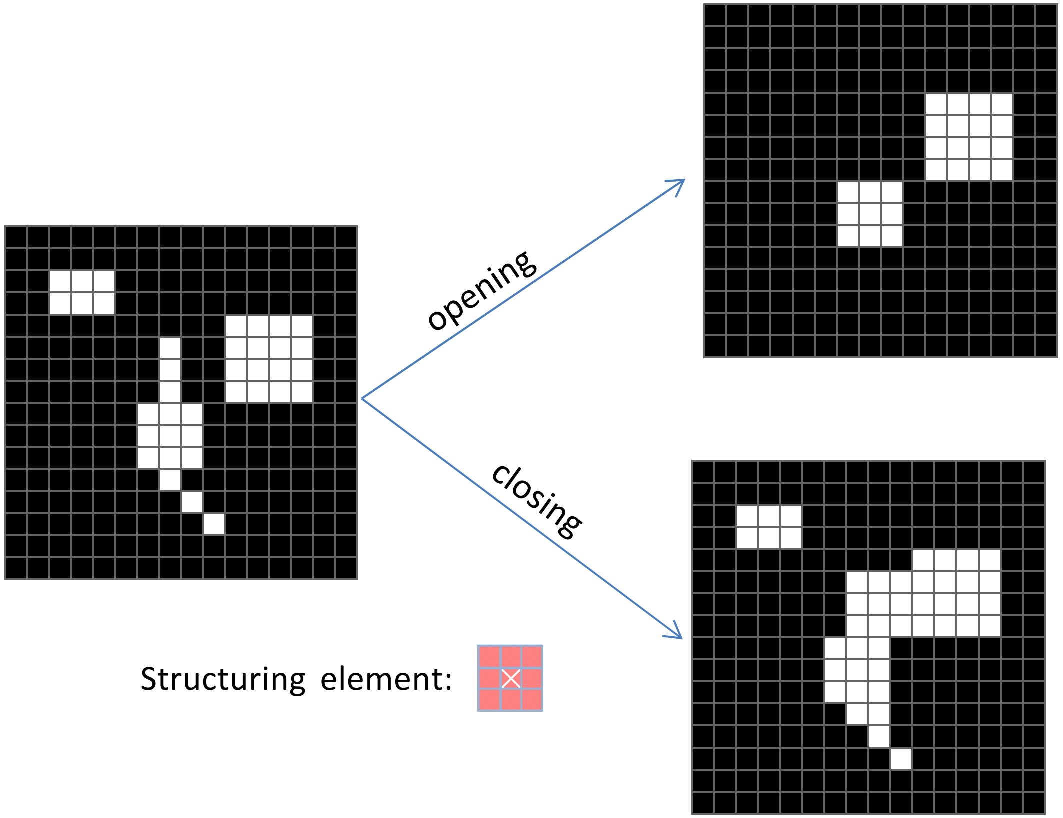
Objects smaller than the structural element are removed:



Morphological image processing: opening and closing

Aim: such an operator which removes holes or objects smaller than a given size without significantly distorting the shape of objects.
Dilation: removes holes smaller than the structuring element but **enlarges objects.**
Erosion: removes objects smaller than the structuring element but **shrinks objects.**

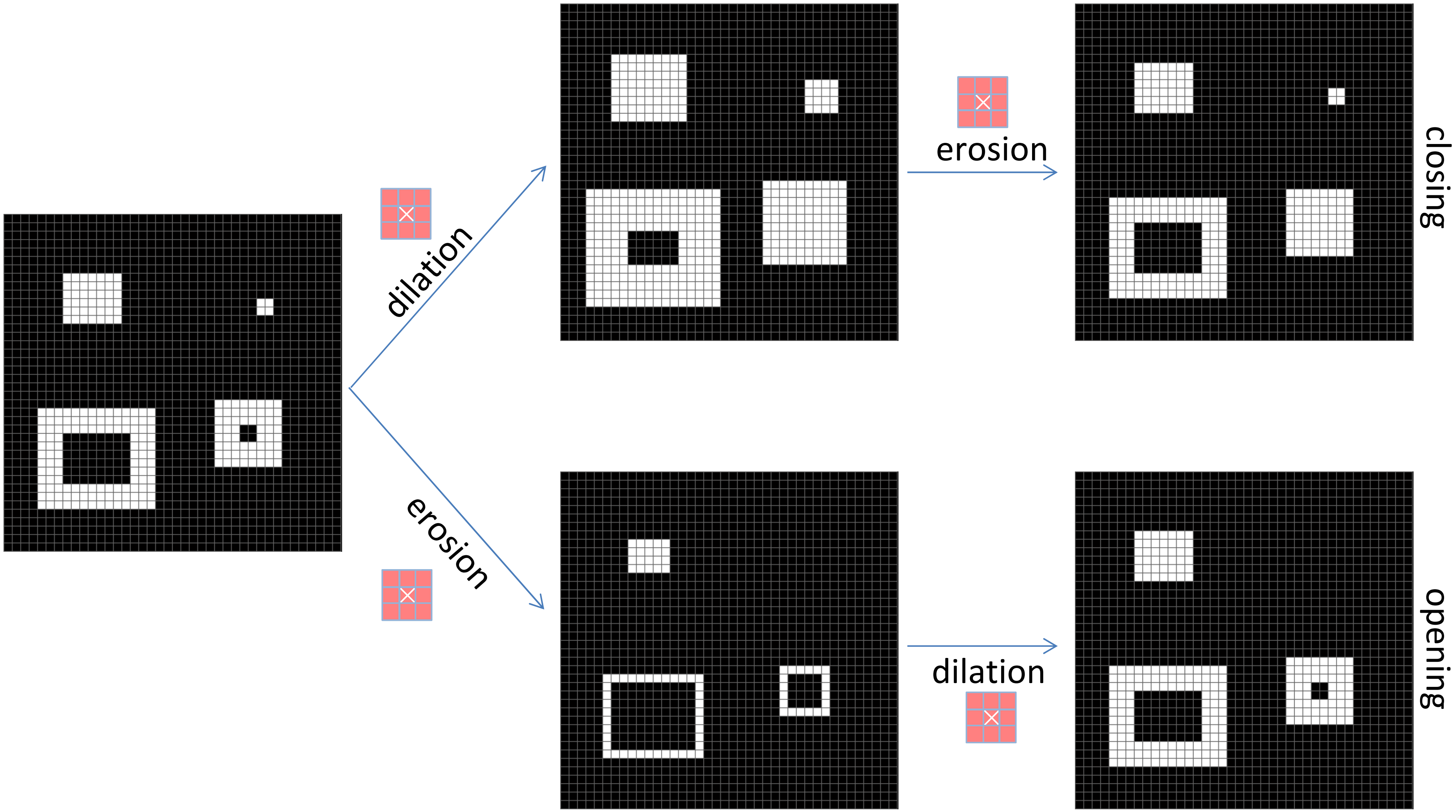
Opening (∘): dilation of the erosion: $A \circ B=(A \ominus B) \oplus B$
Closing (•): erosion of the dilation: $A \bullet B=(A \oplus B) \ominus B$



Objects or parts of objects smaller than the structuring element are removed, but the size of objects larger than the structural element is not changed.

Holes and gaps smaller than the structuring element are removed, but the size of objects is not changed otherwise.

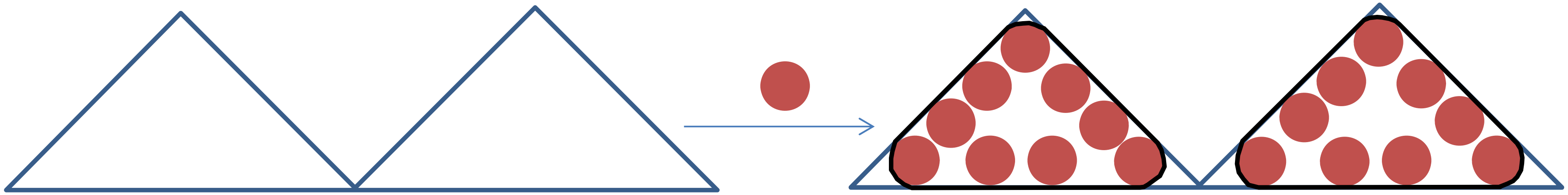
Morphological image processing: erosion, dilation, opening and closing



Morphological image processing: demonstration of opening and closing

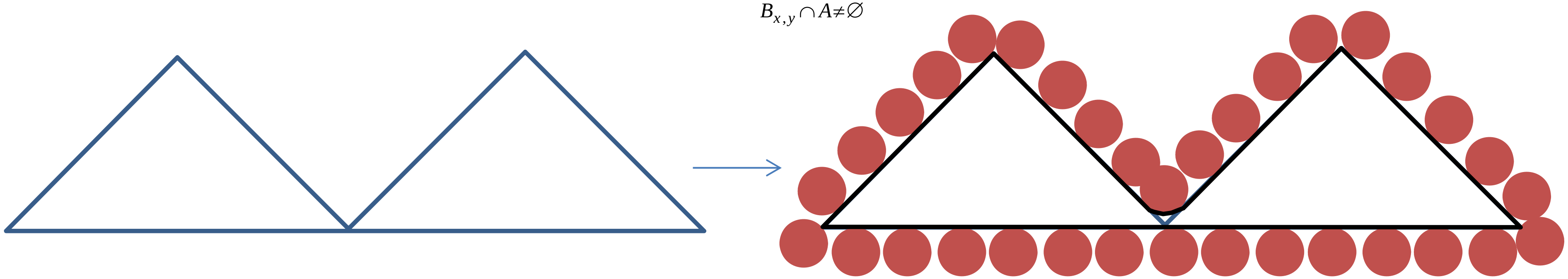
Opening: the union of the S.E. at all positions where the whole S.E. is inside the object. Opening removes those objects or parts of objects that cannot contain the S.E., smoothes contours, breaks thin connections and removes thin protrusions.

$$A \circ B = \bigcup_{B_{x,y} \subset A} B_{x,y}$$



Closing: the complement of the union of the S.E. at all positions where the S.E. is completely outside the object. Closing tends to smooth the contours of objects, but it joins narrow gaps and fills holes smaller than the S.E.

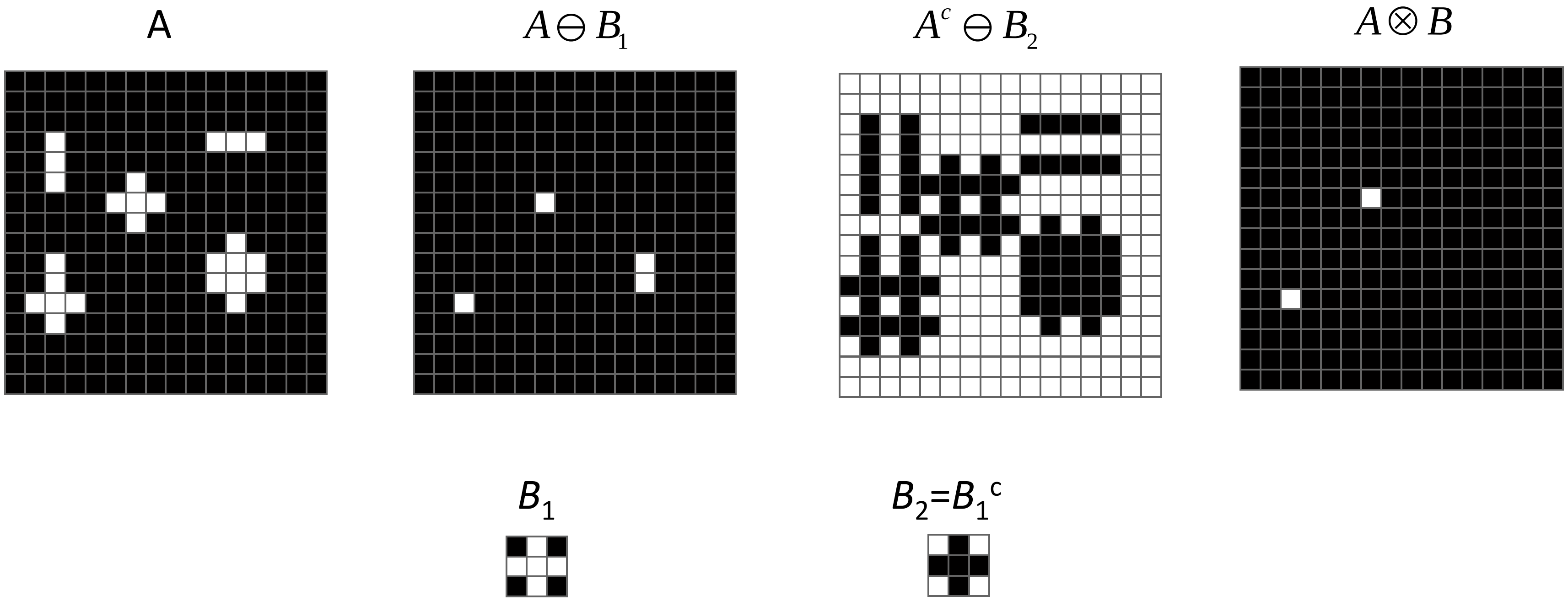
$$A \bullet B = \bigcup_{B_{x,y} \cap A \neq \varnothing} B_{x,y}$$



Morphological image processing: hit-or-miss transform

An operation which can be used to identify the presence of an object with a given shape.

$$A \otimes B = \left(A \ominus B_1 \right) \cap \left(A^c \ominus B_2 \right)$$



A

$A \ominus B_1$

$A^c \ominus B_2$

$A \otimes B$

B_1

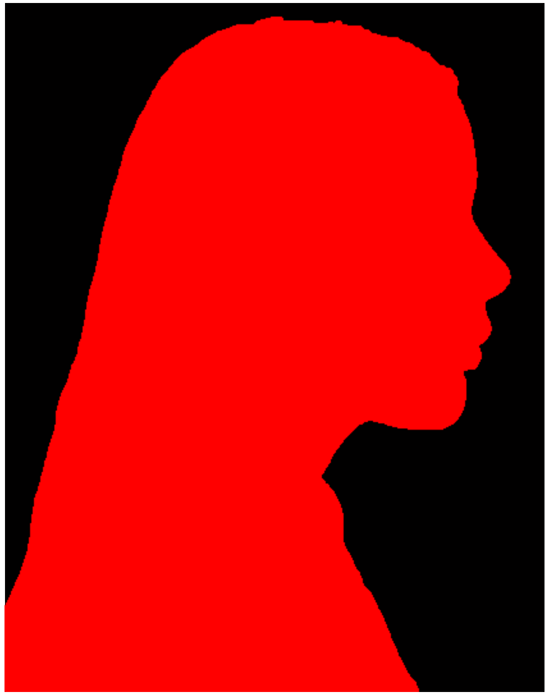
$B_2 = B_1^c$

The hit-or-miss transform identifies objects with a shape identical to structuring element B_1.
Superscript 'c' stands for the complement of a set.

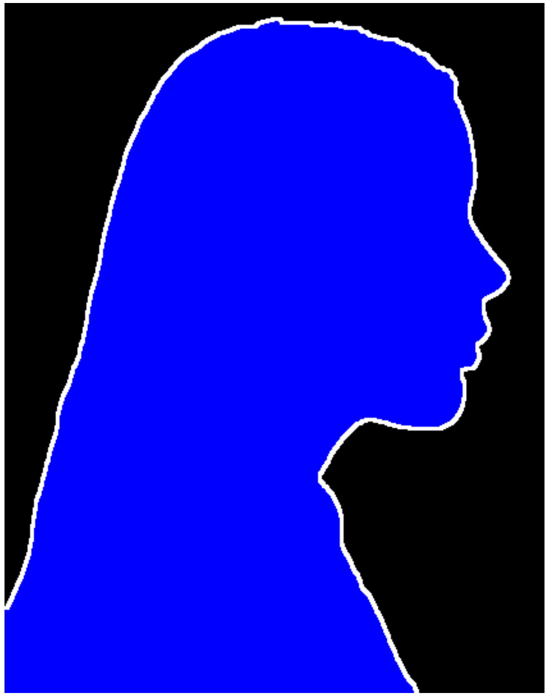
Morphological image processing: edge (boundary) detection

$$boundary(A) = A - (A \ominus B)$$

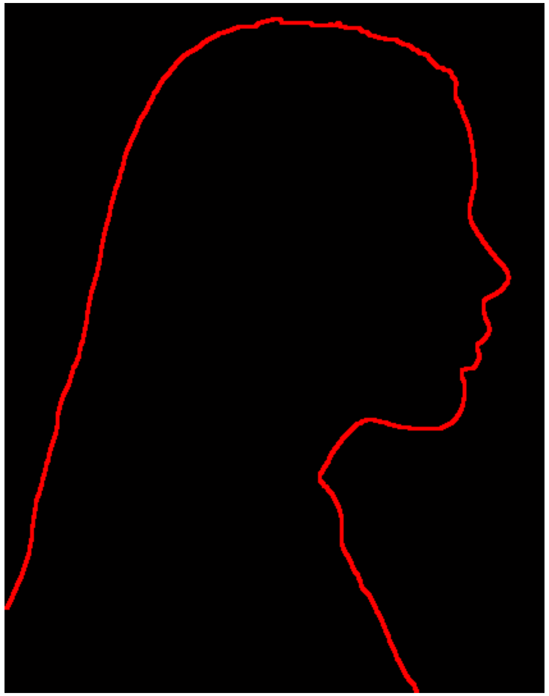
A



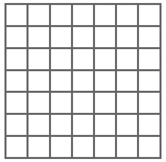
blue – $A \ominus B$
white – boundary of the original image



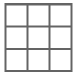
boundary



structuring element:
(thick boundary)



structuring element:
(thin boundary)



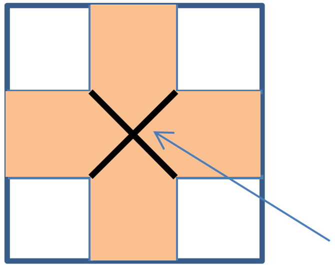
Morphological image processing: morphological operations on non-binary (gray-scale) images

Operation	On a binary image	On a gray-scale image
Dilation	$A\oplus B=\bigcup_{(x,y)\in A}\hat{B}_{x,y}=\left\{(x,y)\middle\vert\hat{B}_{x,y}\cap A\neq\varnothing\right\}$	$(A\oplus B)(x,y)=\max_{(k,l)\in B}\left(a(x+k,y+l)+b(k,l)\right)$
Erosion	$A\ominus B=\left\{(x,y)\middle\vert B_{x,y}\subset A\right\}$	$(A\ominus B)(x,y)=\min_{(k,l)\in B}\left(a(x+k,y+l)+b(k,l)\right)$
Opening	$A\circ B=\bigcup_{B_{x,y}\subset A}B_{x,y}$	$A\circ B=(A\ominus B)\oplus B$
Closing	$A\bullet B=\bigcup_{B_{x,y}\cap A\neq\varnothing}B_{x,y}$	$A\bullet B=(A\oplus B)\ominus B$

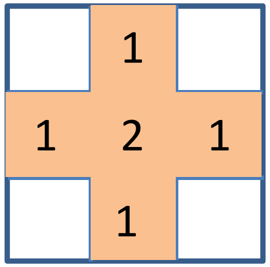
a – intensity of pixels in the image to be processed

b – values of the structuring element

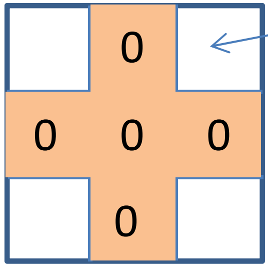
Gray-scale structuring element:



Defines which pixels to consider.



	1	
1	2	1
	1	

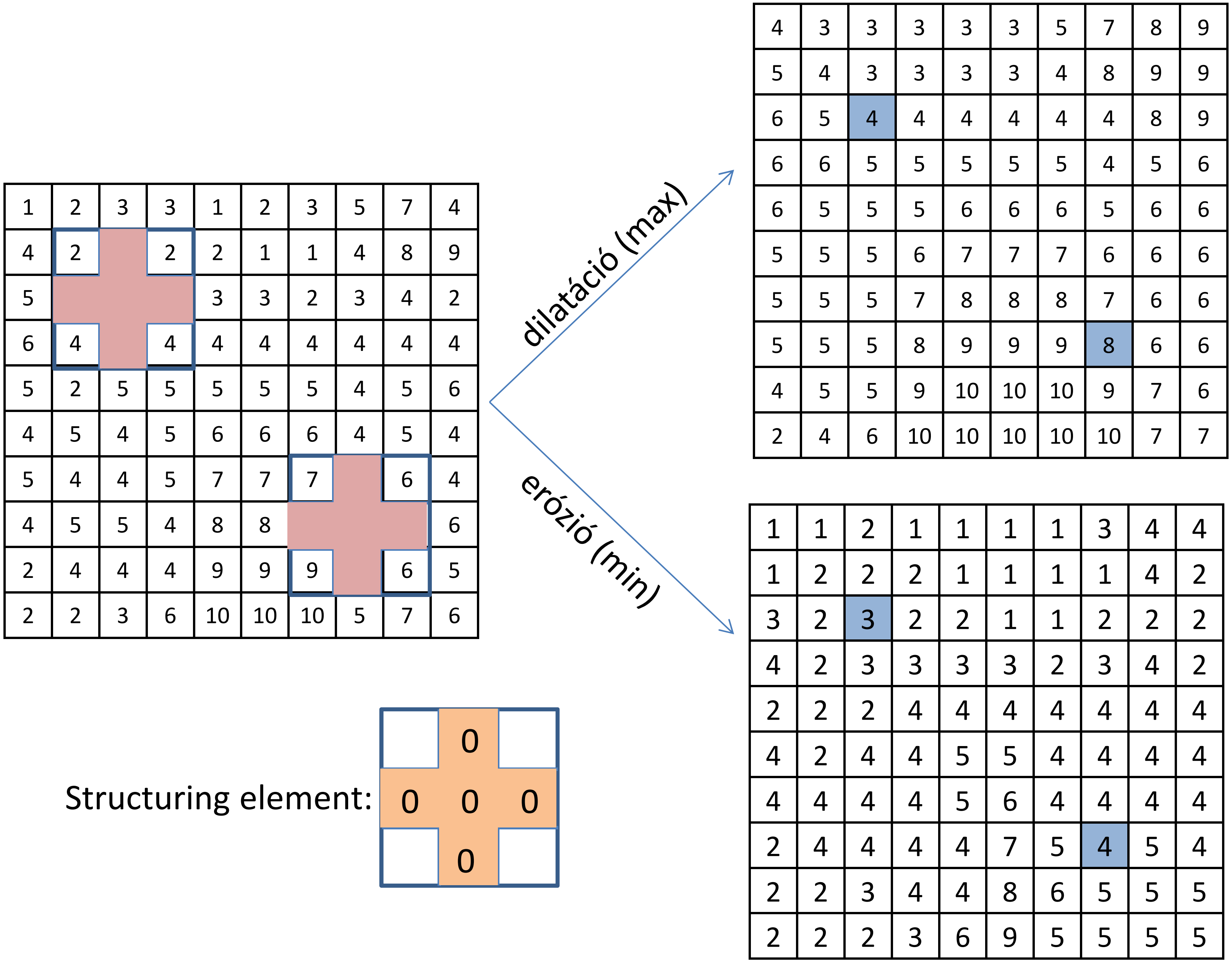


	0	
0	0	0
	0	

If all elements are zero, then it is a simple local maximum operator.

… and how many to add to the pixel intensities of the image.

Morphological image processing: demonstration of gray-scale operators



1	2	3	3	1	2	3	5	7	4
4	2		2	2	1	1	4	8	9
5				3	3	2	3	4	2
6	4		4	4	4	4	4	4	4
5	2	5	5	5	5	5	4	5	6
4	5	4	5	6	6	6	4	5	4
5	4	4	5	7	7	7		6	4
4	5	5	4	8	8				6
2	4	4	4	9	9	9		6	5
2	2	3	6	10	10	10	5	7	6

4	3	3	3	3	3	5	7	8	9
5	4	3	3	3	3	4	8	9	9
6	5	4	4	4	4	4	4	8	9
6	6	5	5	5	5	5	4	5	6
6	5	5	5	6	6	6	5	6	6
5	5	5	6	7	7	7	6	6	6
5	5	5	7	8	8	8	7	6	6
5	5	5	8	9	9	9	8	6	6
4	5	5	9	10	10	10	9	7	6
2	4	6	10	10	10	10	10	7	7

1	1	2	1	1	1	1	3	4	4
1	2	2	2	1	1	1	1	4	2
3	2	3	2	2	1	1	2	2	2
4	2	3	3	3	3	2	3	4	2
2	2	2	4	4	4	4	4	4	4
4	2	4	4	5	5	4	4	4	4
4	4	4	4	5	6	4	4	4	4
2	4	4	4	4	7	5	4	5	4
2	2	3	4	4	8	6	5	5	5
2	2	2	3	6	9	5	5	5	5

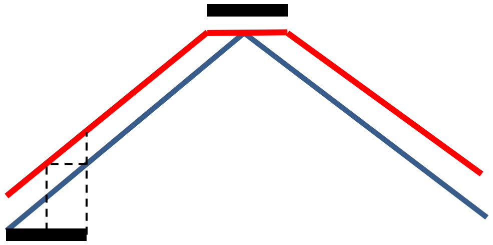
Structuring element:

	0	
0	0	0
	0	

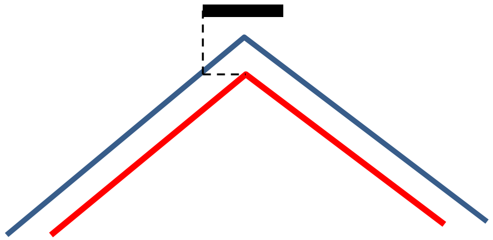
Morphological image processing: morphological operations on non-binary (gray-scale) images

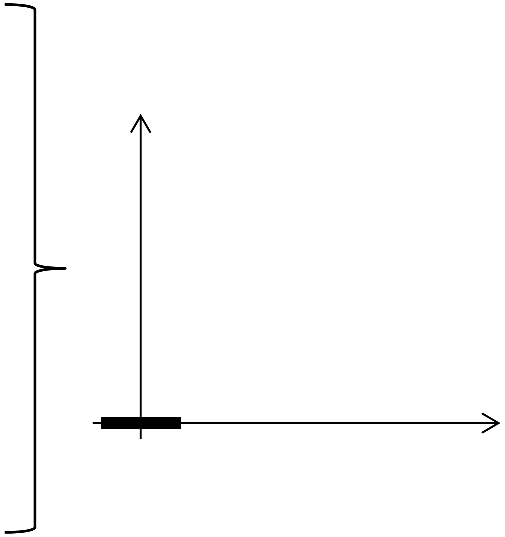
Operation   Demonstration on gray-scale images   Structuring element

Dilation

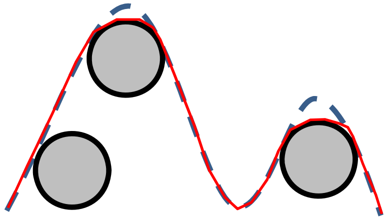


Erosion

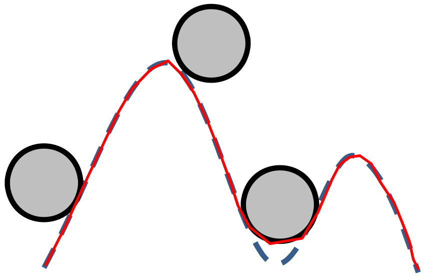


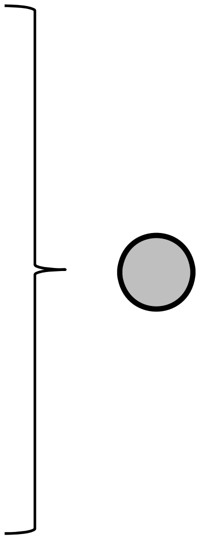


Opening



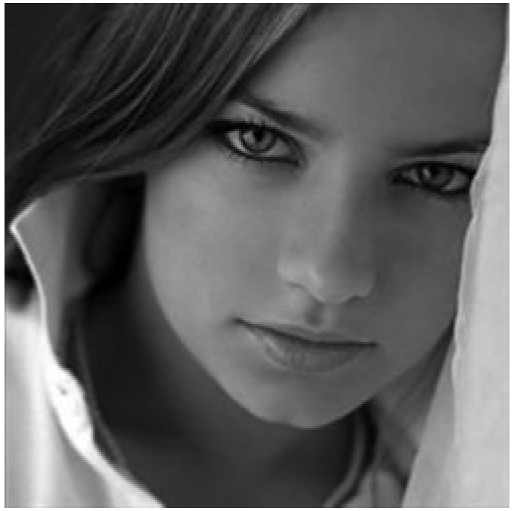
Closing



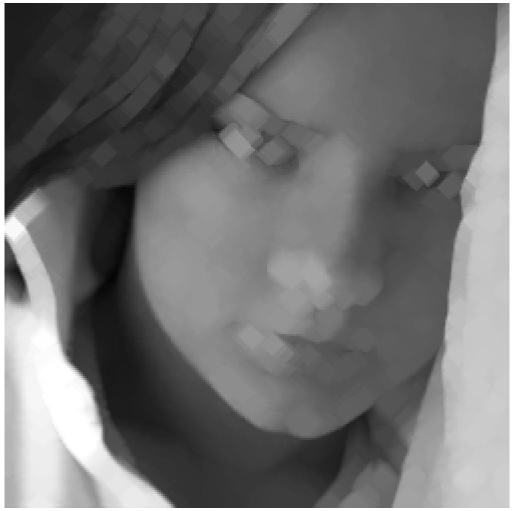


Morphological image processing: morphological operations on non-binary (gray-scale) images

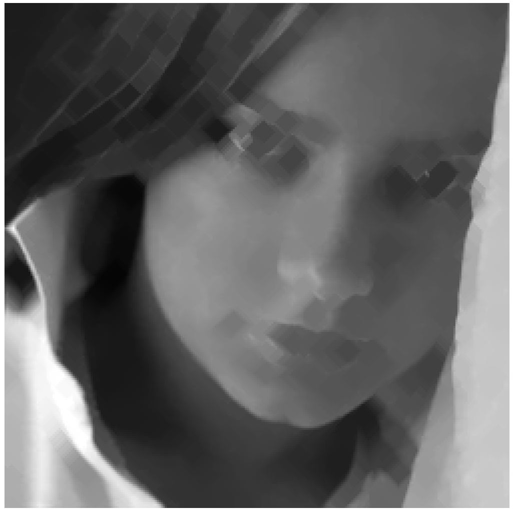
Original image



Dilation



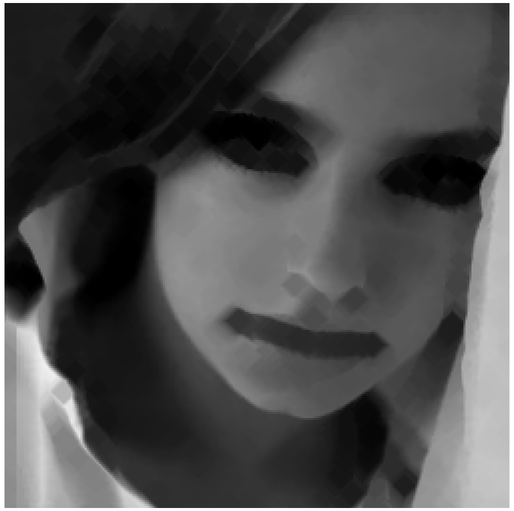
Closing



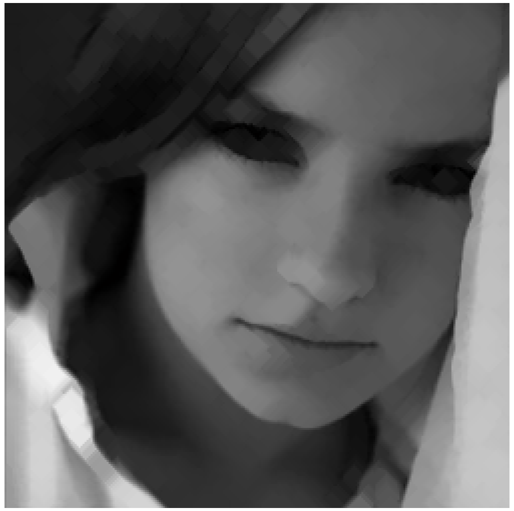
Structuring element:

□	□	□	□	□	■	□	□	□	□	□
□	□	□	□	■	■	■	□	□	□	□
□	□	□	■	■	■	■	■	□	□	□
□	□	■	■	■	■	■	■	■	□	□
□	■	■	■	■	■	■	■	■	■	□
■	■	■	■	■	■	■	■	■	■	■
□	■	■	■	■	■	■	■	■	■	□
□	□	■	■	■	■	■	■	■	□	□
□	□	□	■	■	■	■	■	□	□	□
□	□	□	□	■	■	■	□	□	□	□
□	□	□	□	□	■	□	□	□	□	□

Erosion



Opening



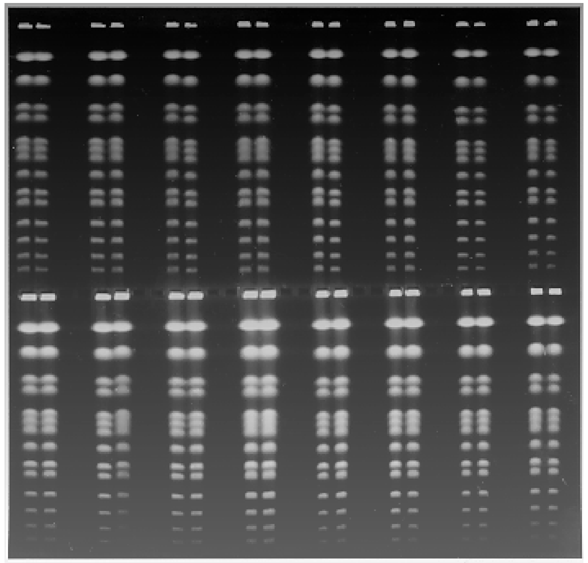
Morphological image processing: tophat transform of gray-scale images

Such an operation which removes objects (intensity trends) larger than the structuring element from the image.

$$tophat(A) = A - (A \bullet B)$$ enhancing dark objects on bright background

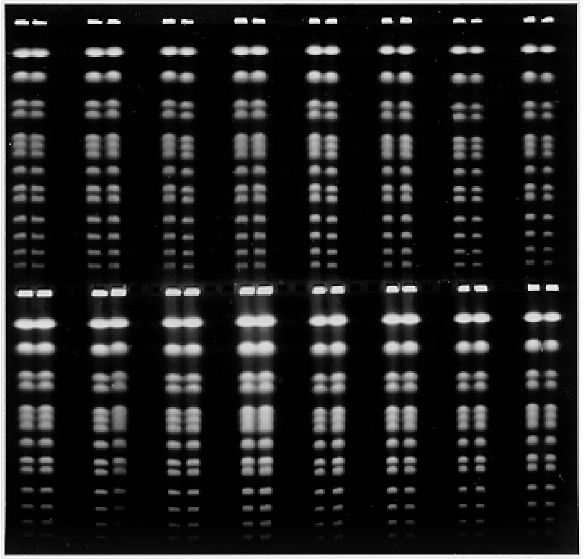
$$tophat(A) = A - (A \circ B)$$ enhancing bright objects on dark background

A



$A \circ B$

tophat(*A*)

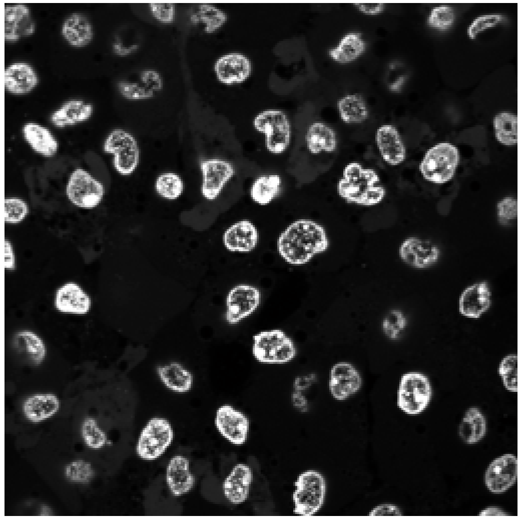


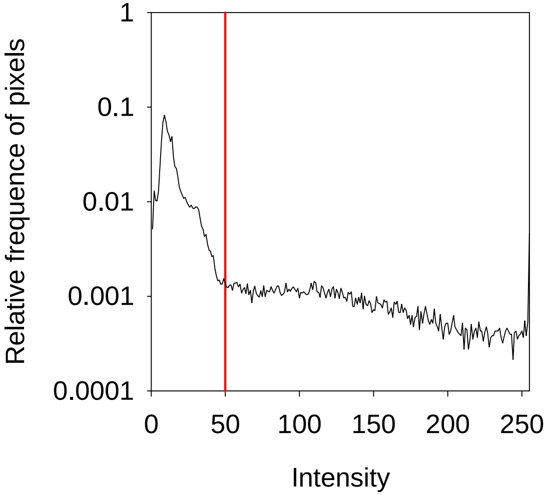
Segmentation

- Dividing the image pixels into different segments (groups, sets).
- Typically the image is divided into two segments:
  - foreground or objects: what is interesting for the evaluator
  - background: what is uninteresting for the evaluator
- Result of segmentation:
  - as many pixel sets (usually two) as the number of segments the image was divided into
  - the boundaries between the sets.
- Segmentation is often preceded by noise reduction and background subtraction.
- Segmentation is necessary because computers lack congenital shape and pattern recognition capabilities (see the image and numerical representation of the Mona Lisa).
- Different means of segmentation:
  - histogram-based segmentation, thresholding (manual, intermeans, maximum entropy, Otsu)
  - edge detection (LoG, Canny)
  - region-based segmentation (watershed, split and merge)

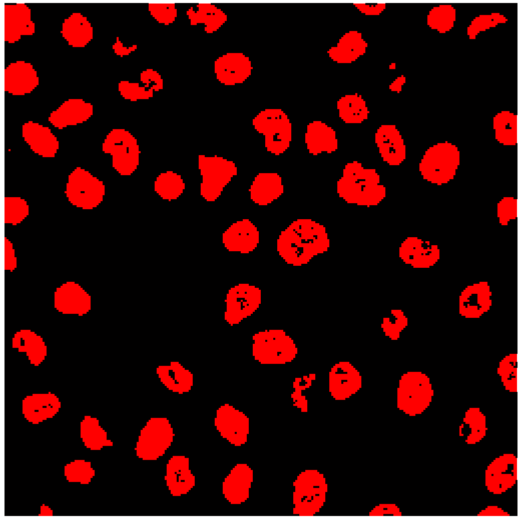
Segmentation: manual thresholding

- The first step in thresholding is the generation of the histogram of pixel intensities:





- In manual thresholding such an intensity value is chosen based on the inspection of the image or the histogram which seems to separate the foreground from the background in the best possible way.



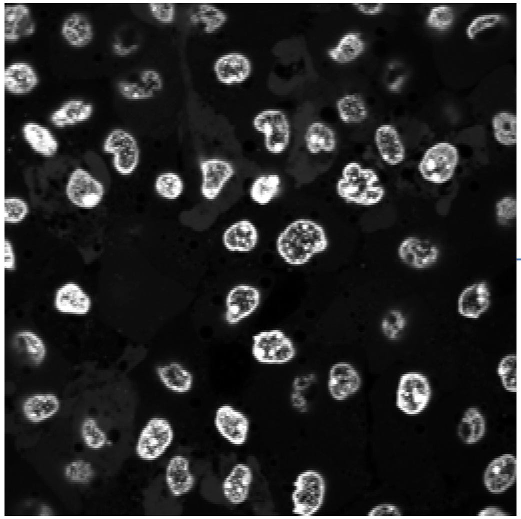
```
segmented=original>50;
```

Segmentation: intermeans algorithm

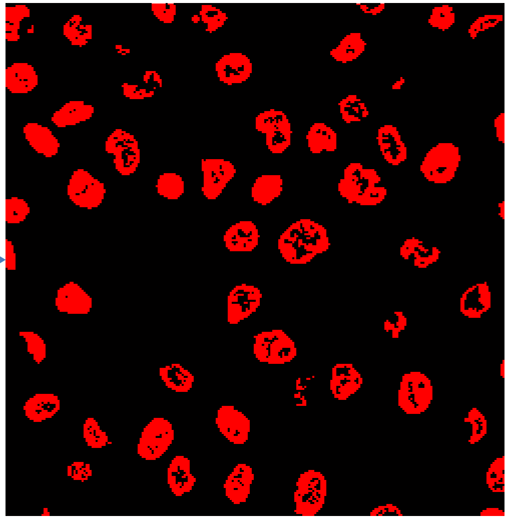
1. Initial threshold=median of the image
2. μ_1=mean of sub-threshold pixels, μ_2=mean of over-threshold pixels
3. Re-estimation of the threshold:

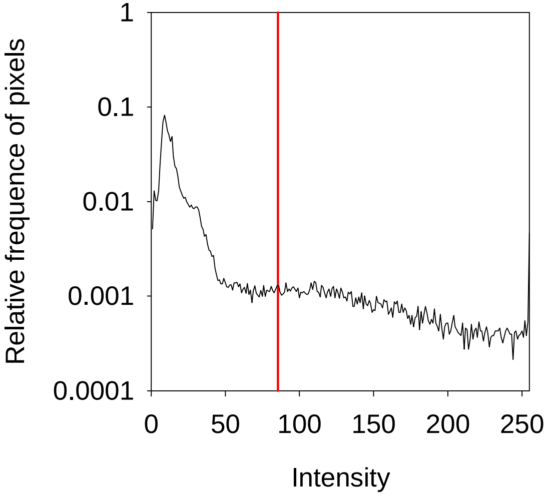
$$thr_{new} = \frac{\mu_1 + \mu_2}{2}$$

4. Repeat steps 2 and 3 until convergence.



threshold:
85.5





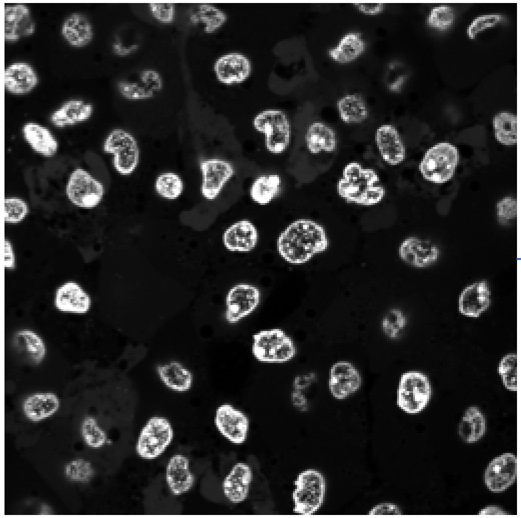
Segmentation: maximum entropy algorithm

In information theory entropy characterizes the uncertainty (and thus information content) of a random variable or set. According to the Shannon formula the entropy of random variable *k*:

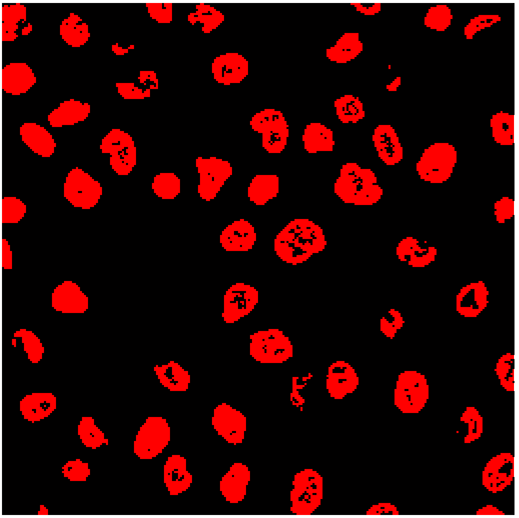
$$H\left(k\right) = -\sum_{i} p_i \log_2 p_i$$

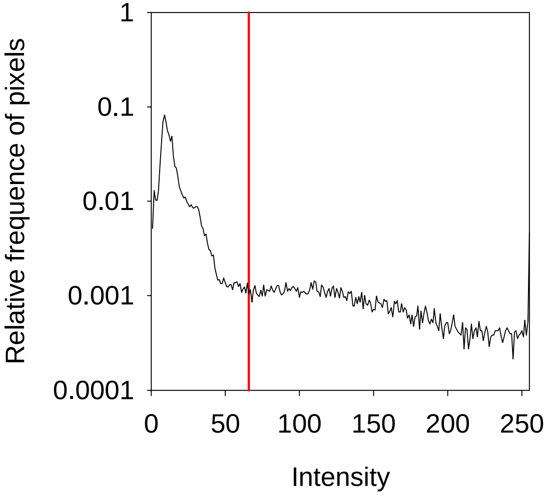
The algorithm:

1. For all possible thresholds (from 0 to the maximum intensity) determine the entropy of sub-threshold pixels.
2. From all the thresholds choose the one which is associated with the largest entropy.



threshold:
66





Segmentation: Otsu's method

$$SS_{tot} = \sum_{i \in all\ pixels} (x_i - \bar{x})^2 = \sum_{j \in all\ groups} \sum_{k \in all\ pixels\ in\ group_j} (x_{j,k} - \bar{x}_j + \bar{x}_j - \bar{x})^2 =$$

$$= \sum_j \sum_k (x_{j,k} - \bar{x}_j)^2 + 2\sum_j \sum_k (x_{j,k} - \bar{x}_j)(\bar{x}_j - \bar{x}) + \sum_j \sum_k (\bar{x}_j - \bar{x}_j)^2 =$$

$$= \sum_j SS_{intra,j} + \sum_j n_j (\bar{x}_j - \bar{x}_j)^2 = \boxed{\sum_j n_j S_j^2} + \boxed{\sum_j n_j (\bar{x}_j - \bar{x}_j)^2}$$

$$\sum_j \sum_k (x_{j,k} - \bar{x}_j)(\bar{x}_j - \bar{x}) = \sum_j (\bar{x}_j - \bar{x}) \underbrace{\sum_k (x_{j,k} - \bar{x}_j)}_{\text{zero in each group}} = 0$$

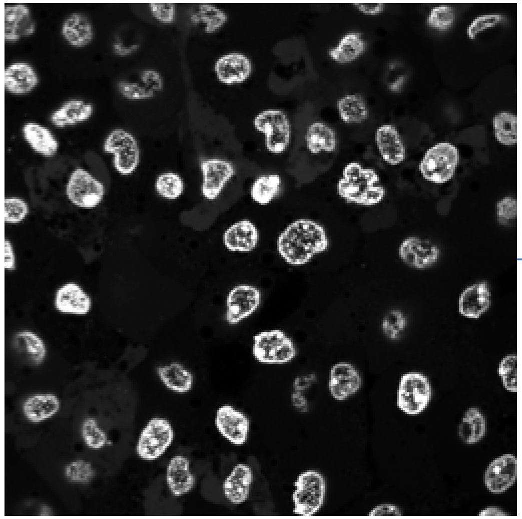
within groups squared deviations

between groups squared deviations

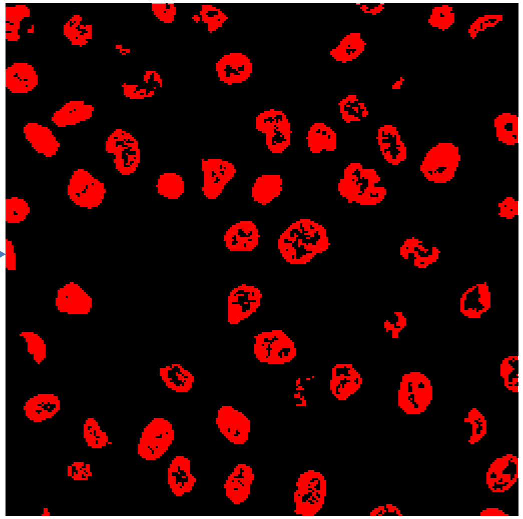
SS – sum of squares

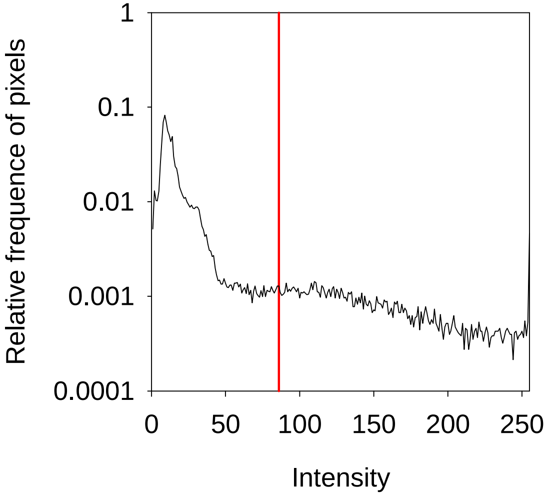
- The essence of Otsu's method is to maximize the between groups variance (or SS) which is equivalent to minimizing the within groups variance (or SS), i.e. such pixels are assigned to a group which are similar to each other regarding their intensity.
- Algorithm:

1. For all possible thresholds calculate the within groups SS (blue box).
2. Choose that threshold for which the within groups SS is minimal.

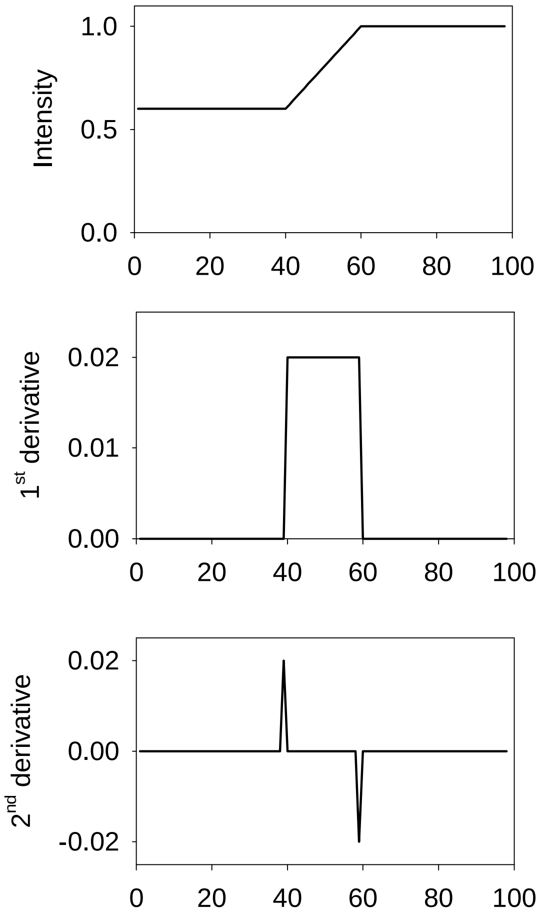


threshold: 66





Segmentation: Laplace operator



- The first and second derivative of a function can be used for edge detection.
- The second derivative of a function of two variables (Laplace operator):

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

- Demonstration of the second derivative:

Function	*f*(1)	*f*(2)	*f*(3)
First derivative	*f*(2)-*f*(1)	*f*(3)-*f*(2)	
Second derivative	$f(3)-f(2)-\left(f(2)-f(1)\right)=$ $=f(3)+f(1)-2f(2)$		

- According to the above the *x* and *y* components of the Laplace operator:

$$\frac{\partial^2 f}{\partial x^2} = f\left(x+1, y\right) + f\left(x-1, y\right) - 2f\left(x, y\right)$$

$$\frac{\partial^2 f}{\partial y^2} = f\left(x, y+1\right) + f\left(x, y-1\right) - 2f\left(x, y\right)$$

- Digital filters implementing the Laplace operator:

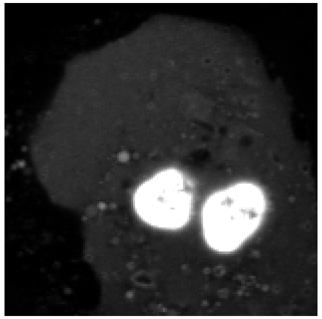
only considering the *x* and *y* directions:

0	-1	0
-1	4	-1
0	-1	0

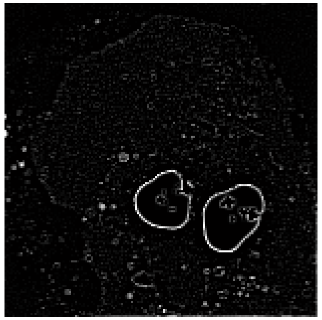
considering the diagonal directions as well:

-1	-1	-1
-1	8	-1
-1	-1	-1

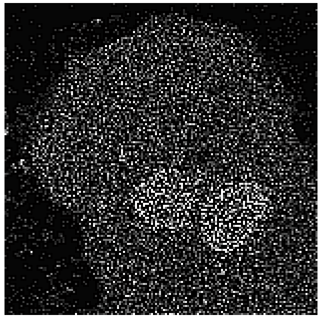
Segmentation: Laplace vs. LoG (Laplacian of Gaussian)



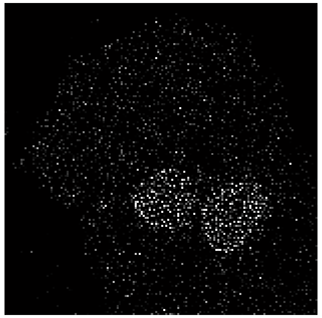
-1	-1	-1
-1	8	-1
-1	-1	-1



The Laplace operator works well on non-noisy images…



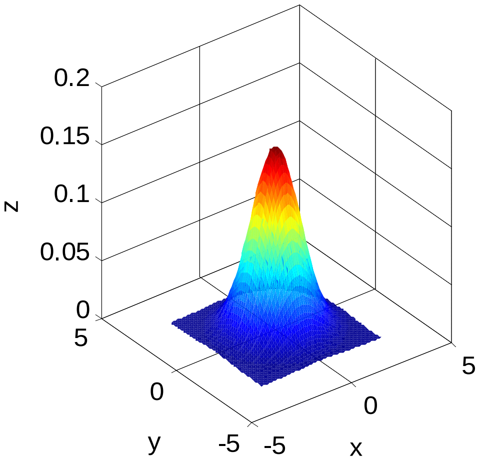
-1	-1	-1
-1	8	-1
-1	-1	-1



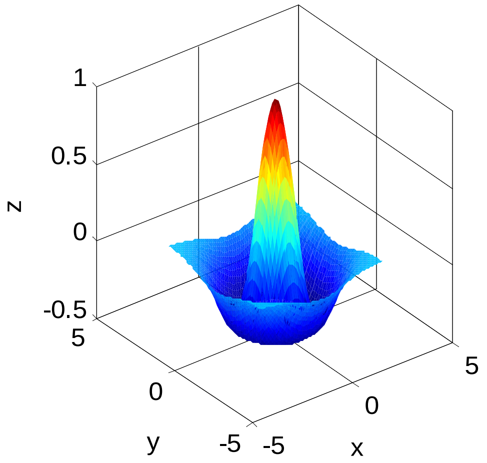
but it fails on the noisy version of the same image.

The Laplace operator is very sensitive to noise, therefore its application has to be preceded by smoothing (low-pass filter) → Gaussian smoothing followed by the Laplace filter = LoG

Segmentation: LoG (Laplacian of Gaussian)



Gaussian function



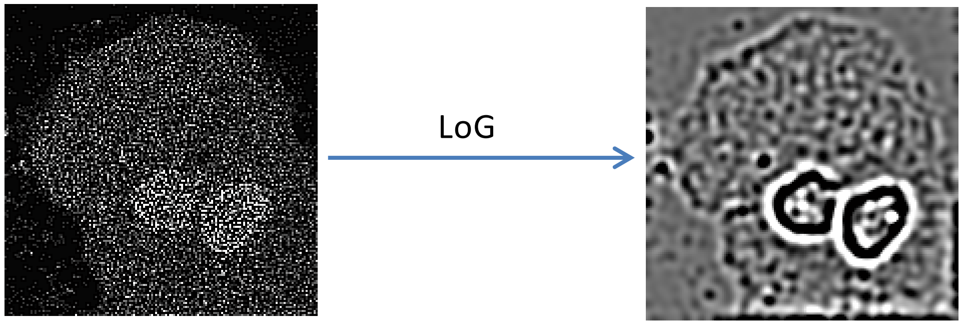
Second derivative of the Gaussian function=LoG

If the image is processed with a digital filter corresponding to the second derivative of the Gaussian (LoG), noise reduction and edge detection are carried out at the same time.

A digital filter implementing the LoG operation:

0	0	-1	0	0
0	-1	-2	-1	0
-1	-2	16	-2	-1
0	-1	-2	-1	0
0	0	-1	0	0

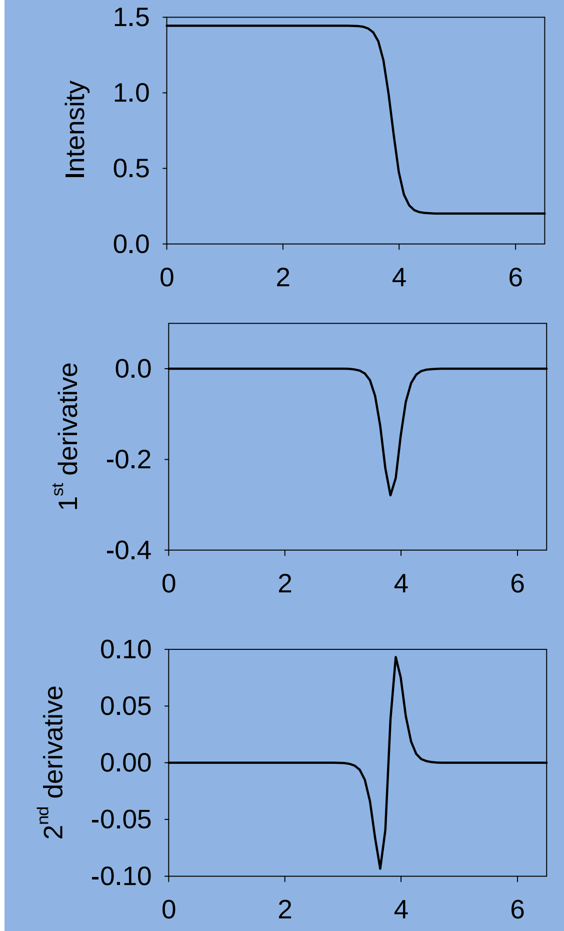
- Depending on how noisy the image is the following parameters can be adjusted:
  - the size of the kernel (filter). 5×5 in the case shown on the right.
  - the variance of the Gaussian function: the more the variance, the more pronounced the noise suppressive effect is.

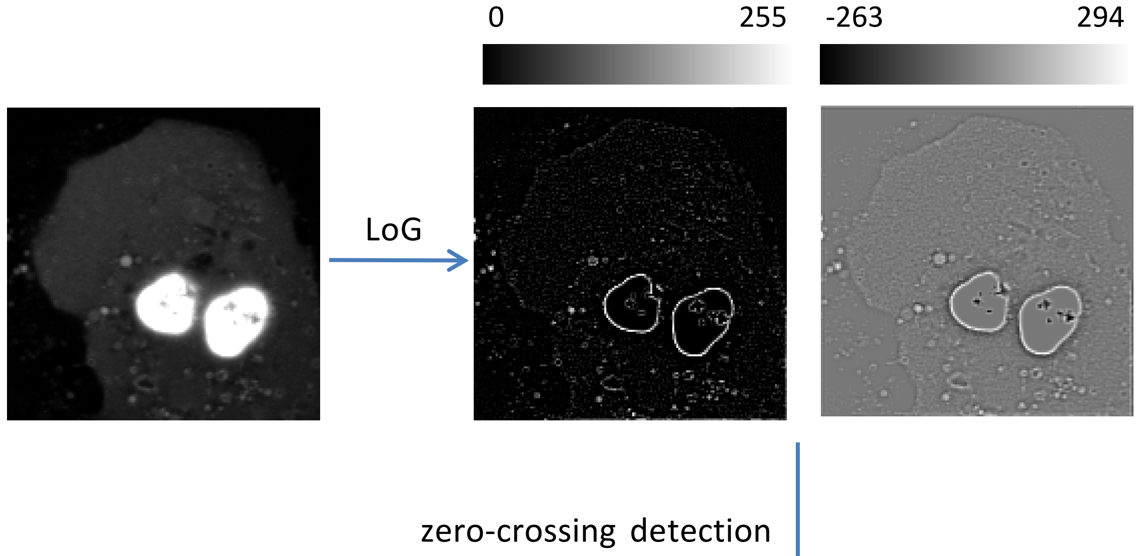


Segmentation: zero crossing of the LoG transform

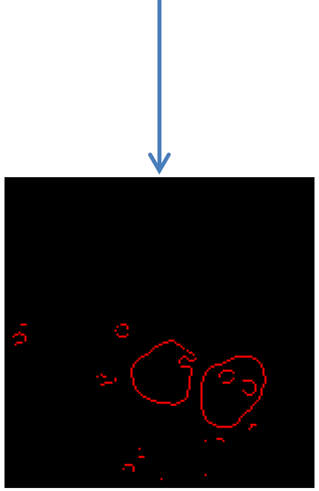
Although the intensity of the LoG-filtered image marks object boundaries, it does not classify pixels to edge/non-edge categories (e.g. 1 – edge pixel, 0 – non-edge pixel).

Zero crossing of the second derivative coincides with the boundary..





Object boundaries can be unequivocally highlighted by detecting the zero crossing of the LoG-transformed image:
zero-crossing pixel – 1
all other pixels l – 0



Segmentation based on the first derivative (gradient)

- The first derivative of a function of two variables (e.g. a 2D image) in the *x* and *y* directions is defined by the following equations:

$$G_x = \frac{\partial f}{\partial x},\ G_y = \frac{\partial f}{\partial y}$$

- G_x and G_y can be determined with the following digital operators according to Prewitt and Sobel:

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

Prewitt

G_x

-1	0	1
-1	0	1
-1	0	1

$z_3 + z_6 + z_9 - z_1 - z_4 - z_7$

G_y

-1	-1	-1
0	0	0
1	1	1

$z_6 + z_7 + z_8 - z_1 - z_2 - z_3$

Sobel

G_x

-1	0	1
-2	0	2
-1	0	1

$z_3 + 2z_6 + z_9 - z_1 - 2z_4 - z_7$

G_y

-1	-2	-1
0	0	0
1	2	1

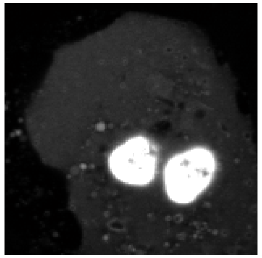
$z_6 + 2z_7 + z_8 - z_1 - 2z_2 - z_3$

- The absolute value of the gradient can be calculated exactly and approximated by the following formulas:

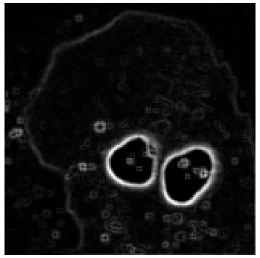
$$|G| = \sqrt{G_x^{\ 2} + G_y^{\ 2}} \approx |G_x| + |G_y|$$

The approximation is used because it requires much less computation, but nowadays this consideration is less important due to the availability of fast computers.

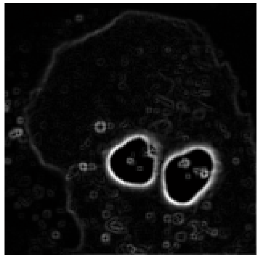
original



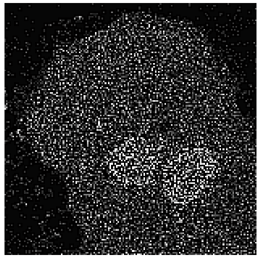
Prewitt



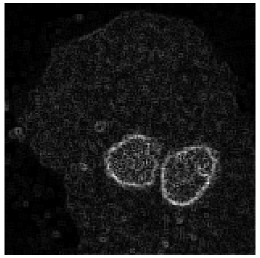
Sobel



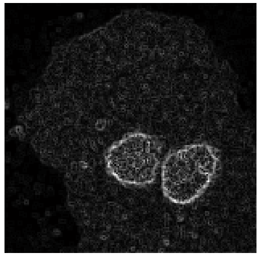
original



Prewitt



Sobel



Segmentation: Canny edge detection

1. Smoothing with a Gaussian filter, e.g. σ=2, kernel size=5×5 →

0.015	0.021	0.024	0.021	0.015
0.021	0.031	0.035	0.031	0.021
0.024	0.035	0.040	0.035	0.024
0.021	0.031	0.035	0.031	0.021
0.015	0.021	0.024	0.021	0.015

2. Calculation of the gradient using the Sobel operator.

3. Calculation of the direction of the gradient in each pixel: $\Theta = \arctan\left(\dfrac{G_y}{G_x}\right)$

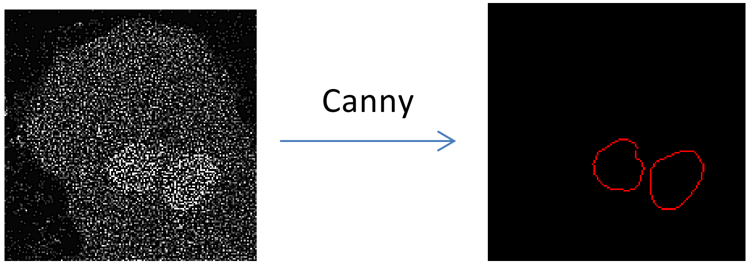
4. Non-maximum suppression:
   - round the gradient direction to integer multiples of 45° (horizontal, vertical, diagonal)
   - compare the gradient of the current pixel to that of the pixel in the direction of the gradient
   - if the gradient of the current pixel is larger than its neighbor’s, then keep it, otherwise zero it

   In this way the edge will be shrunk to a width of one pixel.

2↑	3↑	5↑	4↑	6↑
4↑	5↑	7↑	6↑	7↑
5↑	6↑	4↑	3↑	2↗
3↑	4↑	3↑	1↗	1↗

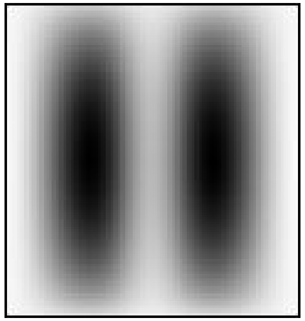
5. Double thresholding:
   - gradient pixels larger than the high threshold are maintained (strong boundary)
   - pixels with gradient values smaller than the low threshold are zeroed
   - pixels with gradient values in between the two thresholds are marked for further analysis (weak boundary)

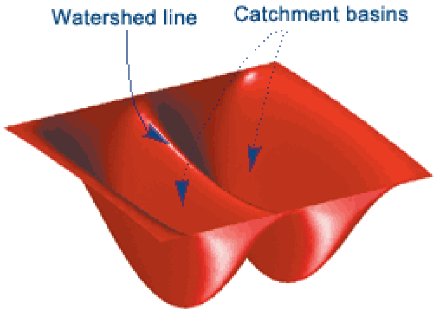
6. Analysis of weak boundaries: if a weak boundary is connected to a strong boundary, it is retained, otherwise it is deleted.



Segmentation: watershed algorithm

- The image has to be imagined as a 3D surface where height is proportional to intensity.

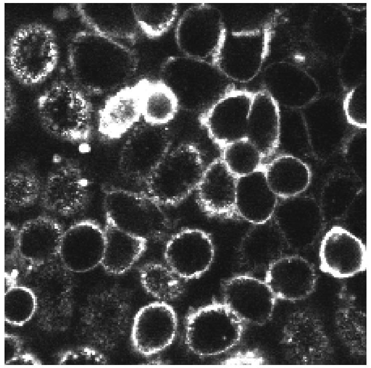




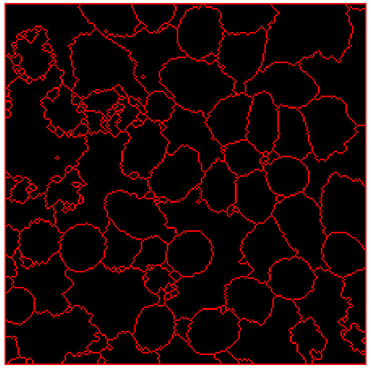
- If water is allowed to flow onto this height map from above, it will find the local minima. From a catchment basin always the same local minimum. The boundary between catchment basins corresponds to the boundary between objects in the image.
- The automatic version of the algorithm tends to find too few or too many boundaries depending on the adjustable parameters.
- Therefore, it is much more efficient if the user marks every cell in the image (seeded watershed segmentation) and then the algorithm will find exactly the same number of cells as seeds placed on the image.

http://www.mathworks.com/company/newsletters/articles/the-watershed-transform-strategies-for-image-segmentation.html

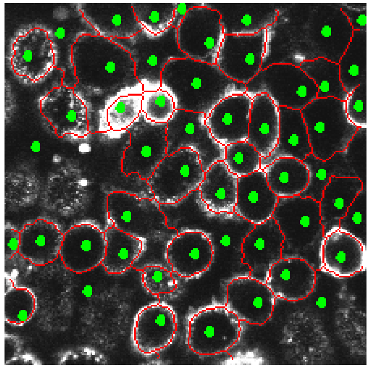
Segmentation: watershed algorithm



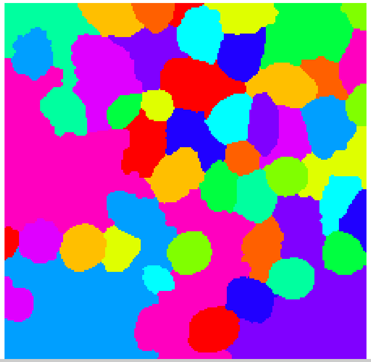
Original image



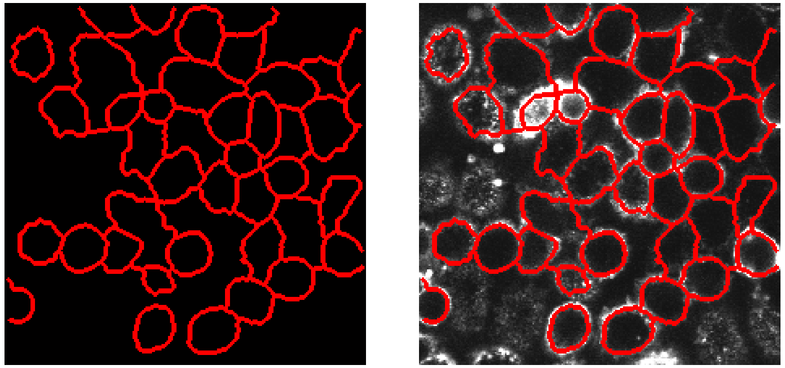
Automatic watershed segmentation



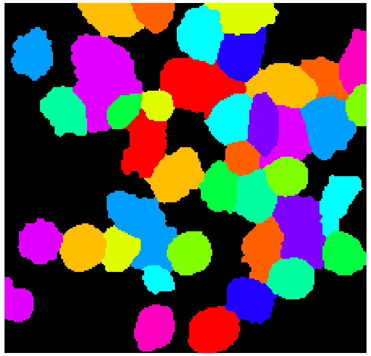
Manually-seeded watershed segmentation with the seeds



Primary segmentation with the manually-seeded algorithm



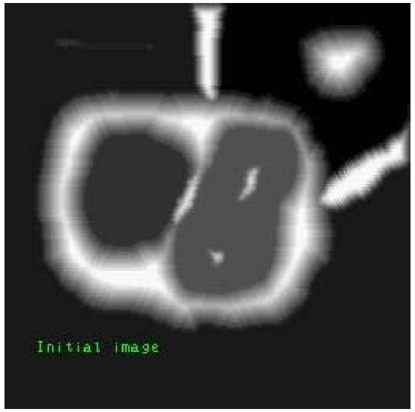
Result of the manually-seeded watershed segmentation: membranes on the left, membranes overlaid on the original image on the right.



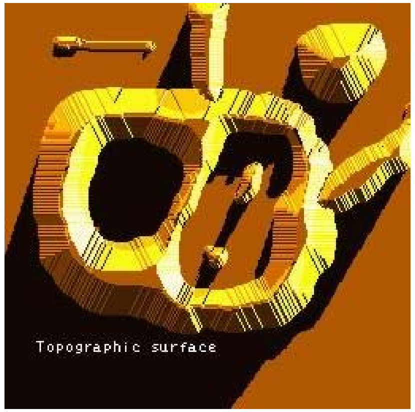
Deletion of non-cell objects, unification of other objects

Segmentation: watershed algorithm

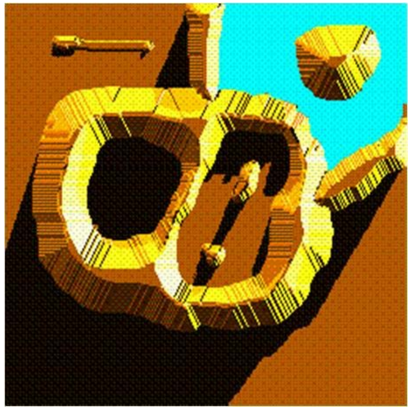
fluorescence image (gray-scale)



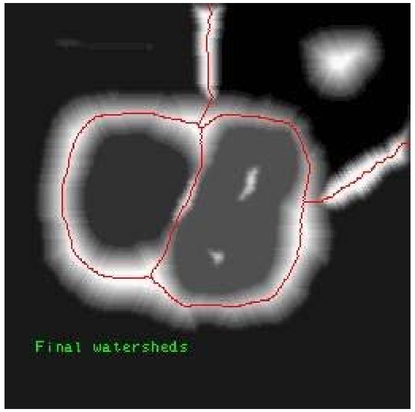
topographical map

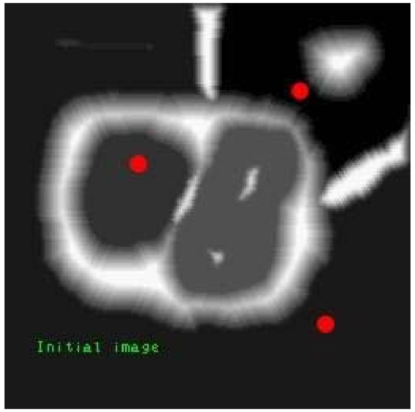


the image is inundated in order to find boundaries

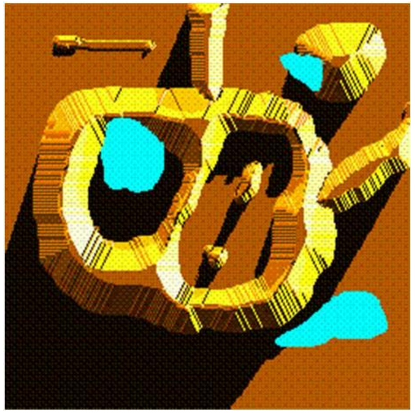


identified boundaries

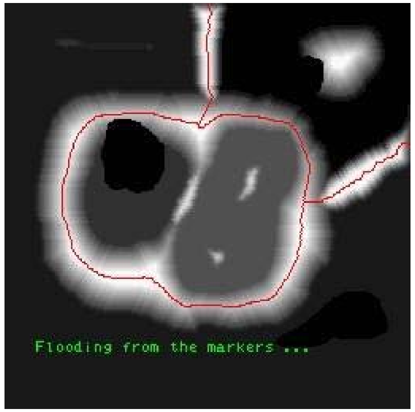




the user places seeds on the image in order to mark objects



inundation is started from the seeds

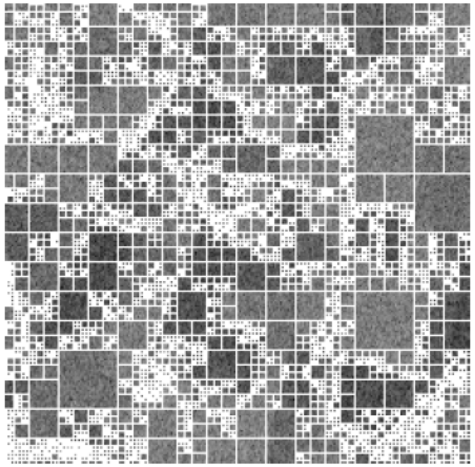


identified boundaries

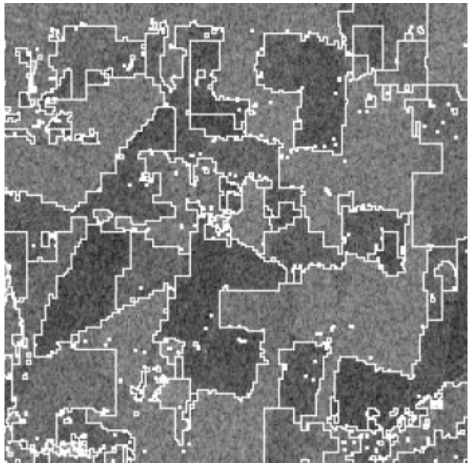
Segmentation: split and merge

1. Choose a value to which the variance of objects will be compared during the algorithm.
2. Splitting phase:
   1. Calculate the variance of the image. If it is over the value chosen in point 1, then split the image into four quadrants.
   2. If the variance of any quadrant is larger than the value chosen in point 1, then split it into four quadrants.
   3. This process has to be repeated until the variance of all sets (quadrants) is under the chosen threshold value.
3. Merging:
   1. For every neighboring area check if merging two neighboring areas results in a set whose variance is below the threshold. If yes, merge them.
   2. The process has to be repeated with the newly generated merged areas as well until no further merging is possible according to the above rule.

Original



Result of the splitting phase



The final result of the merging phase

Segmentation: k-means clustering

1. For every pixel there are *n* different measurements (e.g. pixel intensities at λ_{em}=520 nm, λ_{em}=670 nm, etc.). *n* can be one. The following data set is available:

	measurement 1	measurement 2	…	measurement n
pixel 1	*f*(1,1)	*f*(1,2)	…	*f*(1,n)
pixel 2	*f*(2,1)	*f*(2,2)	…	*f*(2,n)
pixel 3	*f*(3,1)	*f*(3,2)	…	*f*(3,n)
…	…	…	…	…
pixel k	*f*(k,1)	*f*(k,2)	…	*f*(k,n)

2. Decide how many clusters to identify (*N*).
3. For every cluster determine the initial mean:

	measurement 1	measurement 2	…	measurement n
cluster 1	μ(1,1)	μ(1,2)	…	μ(1,n)
cluster 2	μ(2,1)	μ(2,2)	…	μ(2,n)
cluster 3	μ(3,1)	μ(3,2)	…	μ(3,n)
…	…	…	…	…
cluster N	μ(N,1)	μ(N,2)	…	μ(N,n)

4. For every pixel determine to the mean of which cluster it is closest, i.e. to each pixel one of the numbers (1…N) is assigned.
5. The means of the clusters created in point 4 are calculated.
6. The loop is continued until no pixel changes clusters.

Determination of the distance of points: Mahalanobis distance

The distance of a vector x=(x_1,x_2,x_3,..,x_n) from another vector μ=(μ_1,μ^2,μ^3,…μ_n) can be determined by the following formula:

$$D(\mathbf{x},\boldsymbol{\mu})=\sqrt{(\mathbf{x}-\boldsymbol{\mu})\mathbf{V}^{-1}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}}$$

where **V** is the variance-covariance matrix, -1 and T in the superscript stand for matrix inversion of transposition, respectively.

The variance-covariance matrix:

$$\begin{bmatrix} \sigma_{1,1}{}^2 & \sigma_{1,2}{}^2 & \dots & \sigma_{1,n}{}^2 \\ \sigma_{2,1}{}^2 & \sigma_{2,2}{}^2 & \dots & \sigma_{2,n}{}^2 \\ \dots & \dots & \dots & \dots \\ \sigma_{n,1}{}^2 & \sigma_{n,2}{}^2 & \dots & \sigma_{n,n}{}^2 \end{bmatrix}, \sigma_{x,y}{}^2 = M\left[(x-\mu_x)(y-\mu_y)\right]$$

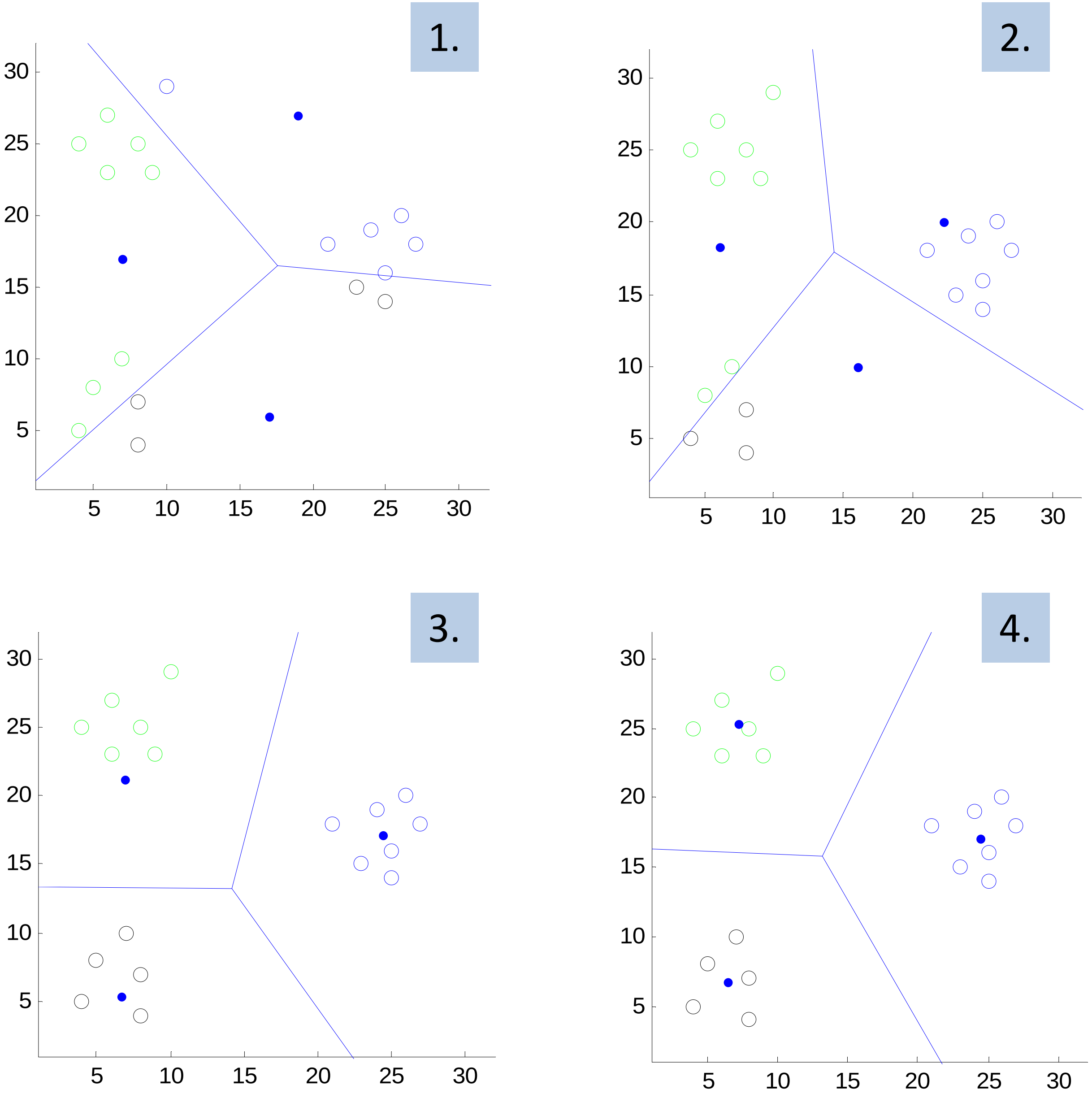
$\sigma_{1,1}$ is the variance of the 1st measurement, $\sigma_{1,2}$ is the covariance of the 1st and 2nd measurements.

diagonal matrix normalized Euclidean distance identity matrix Euclidean distance

$$\text{If}\begin{bmatrix} \sigma_{1,1}{}^2 & 0 & \dots & 0 \\ 0 & \sigma_{2,2}{}^2 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \sigma_{n,n}{}^2 \end{bmatrix} \Rightarrow D(\mathbf{x},\boldsymbol{\mu})=\sqrt{\sum_i \frac{(x_i-\mu_i)^2}{\sigma_{i,i}{}^2}}$$

$$\text{Ha}\begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 1 \end{bmatrix} \Rightarrow D(\mathbf{x},\boldsymbol{\mu})=\sqrt{\sum_i (x_i-\mu_i)^2}$$

Segmentation: k-means clustering



The k-means clustering algorithm converges in four steps in this example of clustering involving two variables.

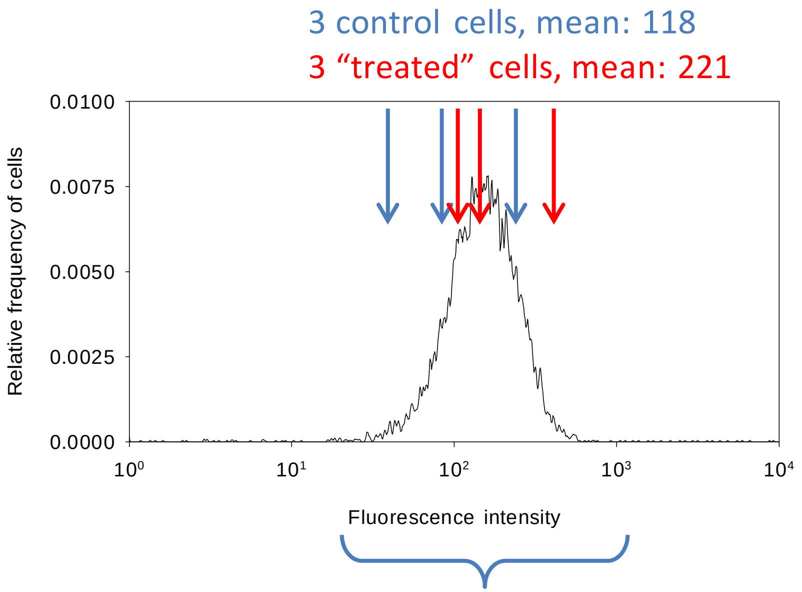
Background subtraction

Algorithms which remove the background intensity and only the specific fluorescence intensity of objects is retained.

- constant background subtraction: the same value is subtracted from every pixel in the image
- all high-pass filters: the background contains low frequency components which can be removed with a high-pass filter
- max-min filter
- tophat filter

Measurements

- After the identification of objects (foreground) by segmentation the last step of image analysis is typically the extraction of quantitative information.
- Why is quantitative evaluation necessary?
  - so that the conclusion drawn from the images is not subjective
  - only few cells can be measured by microscopy making microscopy statistically less robust
    - → this can be mitigated by quantitative evaluation
    - → it is advisable to measure and evaluate as many cells as possible so that the standard error of the statistics decreases

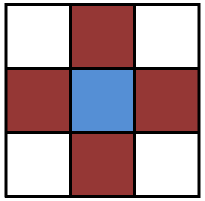


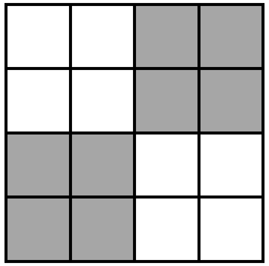
It is quite common that biological samples show a variability of 1-2 orders of magnitude. If conclusions are drawn from the analysis of a couple of cells, mistakes can easily be committed.

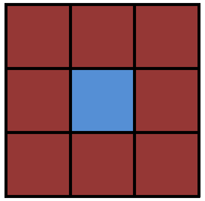
Measurements: 4- or 8-connectedness, the Freeman chain code

From the standpoint of measurements connectivity is an important concept, i.e. pixels in what kind of relation are considered to be neighbors:

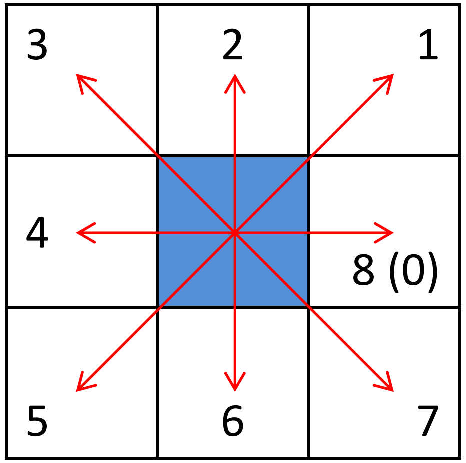
- 4-connectedness:
  a pixel has four neighbors (only those which touch it along the edges)
- 8-connectedness:
  a pixel has eight neighbors (at the corners and along the edges)



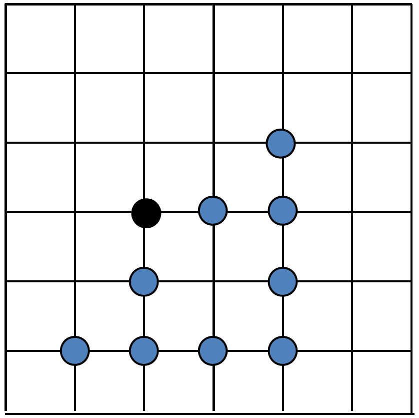




Based on 4-connectedness the above counts as two objects, based on 8-connectedness it is one object.



Starting at a given contour pixel and moving clockwise, we give each pixel a code based upon its direction from the previous pixel.

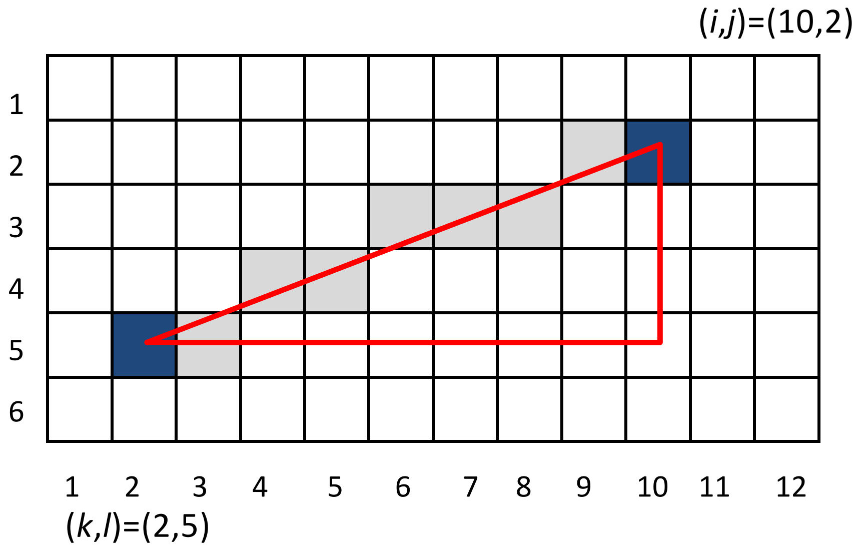


The above circumference is described by the following Freeman chain code starting from the black point:

8→1 →6 →6 →6 →4 →4 →4 →1 →2

Measurements: length estimation

What is the distance between the two blue pixels?



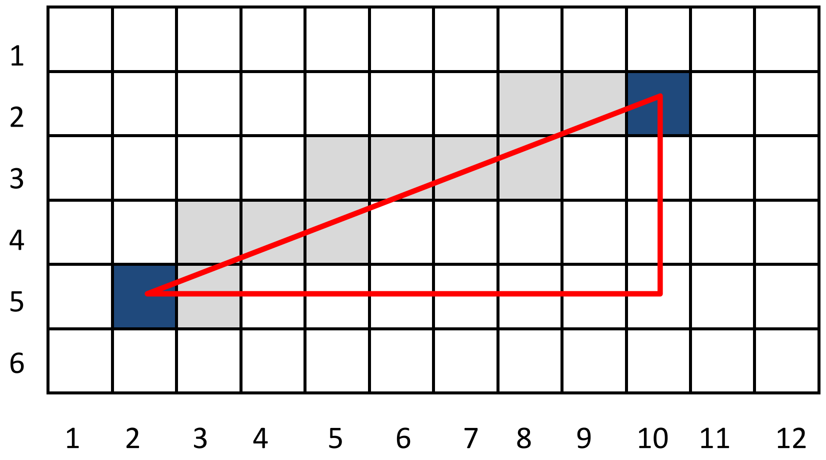
1. Euclidean distance:

$$\sqrt{(i-k)^2+(j-l)^2}=\sqrt{(10-2)^2+(2-5)^2}=8.5$$

2. Chessboard distance:

- the number of steps needed by a king to move from one place to the other
- the shortest chain of 8-connected pixels connecting the two points

$$\max\left(\left|i-k\right|,\left|j-l\right|\right)=8$$



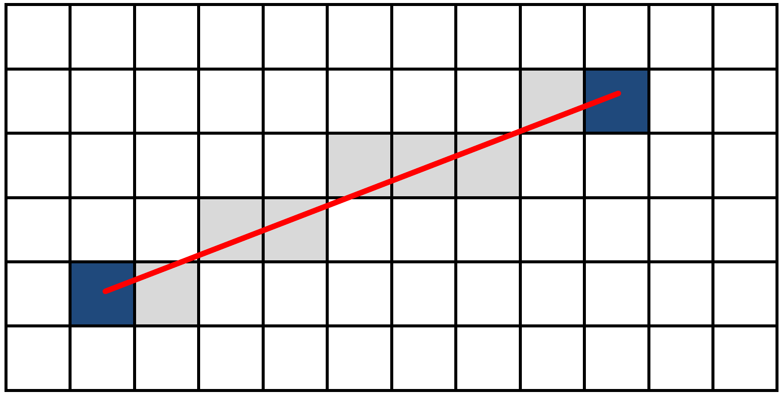
3. City-block distance:

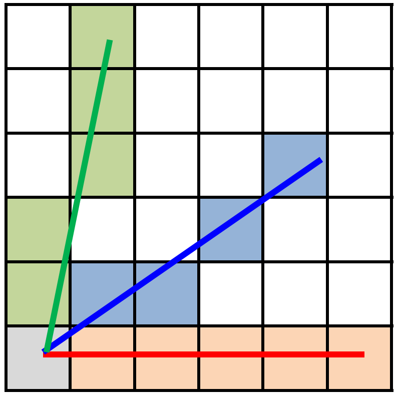
- the shortest chain of 4-connected pixels connecting the two points

$$\left|i-k\right|+\left|j-l\right|=11$$

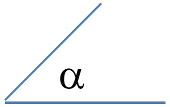
Measurements: estimating the length of a curve or line

8-connected neighborhood





lines of length *d* at different angles



Number of pixels:

- 5·cos 0°=5
- 5·cos 30°=4
- 5·sin 80°=5

$$n_{pixel} = d\cos\alpha,\ ha\ \ \alpha \le 45^{\circ}$$

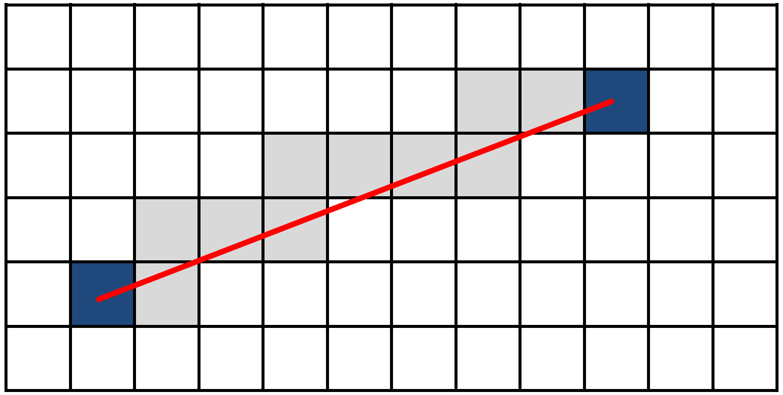
$$n_{pixel} = d\sin\alpha,\ ha\ \ \alpha > 45^{\circ}$$

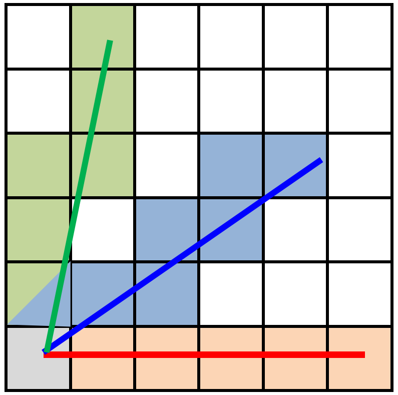
The mean of the above between 0° and 45°:

$$\frac{4}{\pi}\int_0^{\pi/4}\cos\alpha\ d\alpha = \frac{4}{\pi\sqrt{2}} = 0.9$$

If a line contains *n* pixels, its length is *n*/0.9.
The length of the above line: 8/0.9=8.88

4-connected neighborhood





$$n_{pixel} = d\left(\cos\alpha + \sin\alpha\right)$$

Number of pixels:

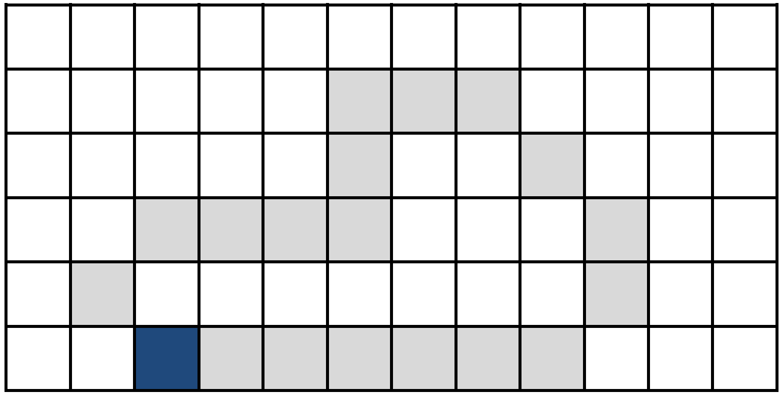
- 5·(cos 0° + sin 0°) =5
- 5·(cos 30° + sin 30°) =7
- 5·(cos 80° + sin 80°) =6

The mean of the above between 0° and 45°:

$$\frac{4}{\pi}\int_0^{\pi/4}\left(\cos\alpha + \sin\alpha\right)\ d\alpha = \frac{4}{\pi} = 1.273$$

If a line contains *n* pixels, its length is *n*/1.273.
The length of the above line is: 8/1.273=6.28

Measurements: estimating the length of a curve

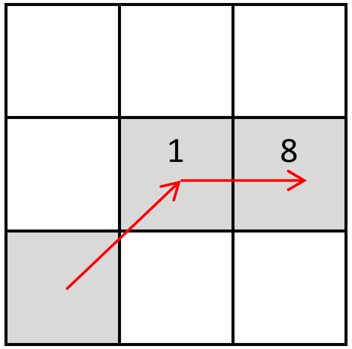


parallel steps – even Freeman chain code
diagonal steps – odd Freeman chain code

	steps parallel to the grid (14)	steps diagonal to the grid (5)	number of corners (3)	length of the above circumference
number of pixels	1	1	0	19
Freeman (1970)	1	$\sqrt{2}$	0	21.071
Proffit (1979)	0.984	1.340	0	20.476
Vossepoel (1982)	0.980	1.406	-0.091	20.477

Corner:

- the number of even-odd and odd-even transitions in the Freeman chain code.
- steps of a chess knight



$$D_{count} = n_e + n_o$$

$$D_{Freeman} = n_e + \sqrt{2} n_o$$

$$D_{Proffit} = 0.984 n_e + 1.340 n_o$$

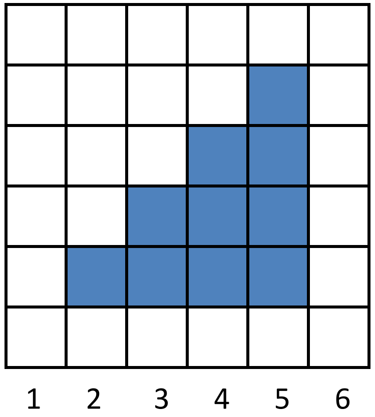
$$D_{Vossepoel} = 0.98 n_e + 1.406 n_o - 0.091 n_c$$

n_e – number of even Freeman chain codes
n_o – number of odd Freeman chain codes
n_c – number of corners

Measurements: area estimation

What is the area of the triangle on the right?

- estimation 1: A_1=number of pixels=10
- estimation 2: A_2= ½ (3x3)=4.5 (according to the formula for the area of a triangle if the length of an edge=end point - starting point=5-2=3)
- estimation 3: A_3= ½ (4x4)=8 (according to the formula for the area of a triangle if the length of an edge=number of pixels=4)

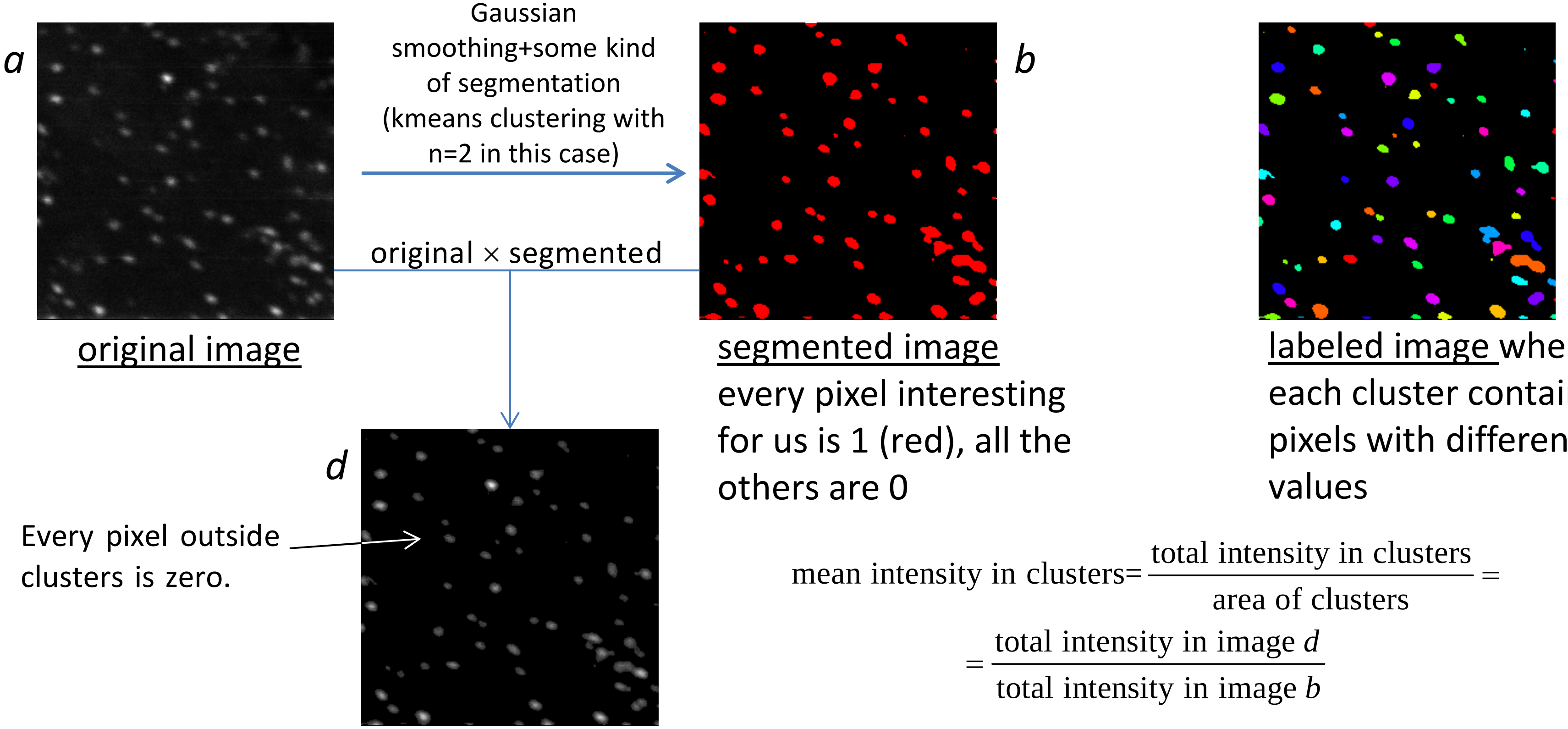


All three estimations are acceptable and used.

Conclusions for length, circumference and area estimations:

- the digitized version of real objects are visible in images
- during the measurements we are interested in the parameters (length, area) of the real objects
- since the digital image is only an approximation of the real object, estimations carried out on the digital image can also only be approximations of the real parameters

Measurements: determination of mean intensities


a
b
c
d
Gaussian smoothing+some kind of segmentation (kmeans clustering with n=2 in this case)
original × segmented

original image

segmented image every pixel interesting for us is 1 (red), all the others are 0

labeled image where each cluster contains pixels with different values

Every pixel outside clusters is zero.

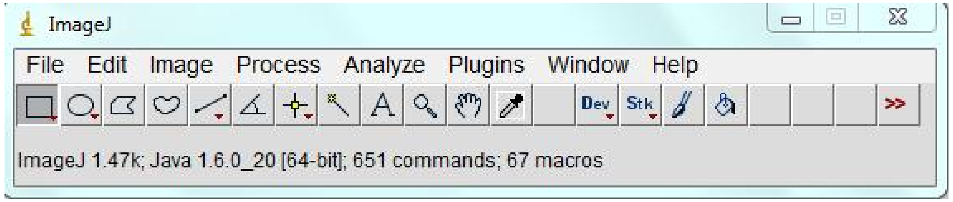
$$\text{mean intensity in clusters} = \frac{\text{total intensity in clusters}}{\text{area of clusters}} = \frac{\text{total intensity in image } d}{\text{total intensity in image } b}$$

- Area of all clusters: the sum of the red pixels (since every red pixel=1; the segmented image is stored in variable „*b*”):
  - Matlab: area=sum(b(:))   DipImage: area=sum(b);
- The total intensity in the original image in areas corresponding to clusters:
  - d=a*b;
  - Matlab: totint=sum(d(:));   DipImage: totint=sum(d);
- Mean intensity in clusters:
  - totint/area
- **The above calculation can be performed for each individual cluster using image “*c*”.**

DipImage: sum(a(b));
Matlab:
sum(a (reshape(b,numel(b),1)==1))/
sum(b(:))

Scientific image analysis programs

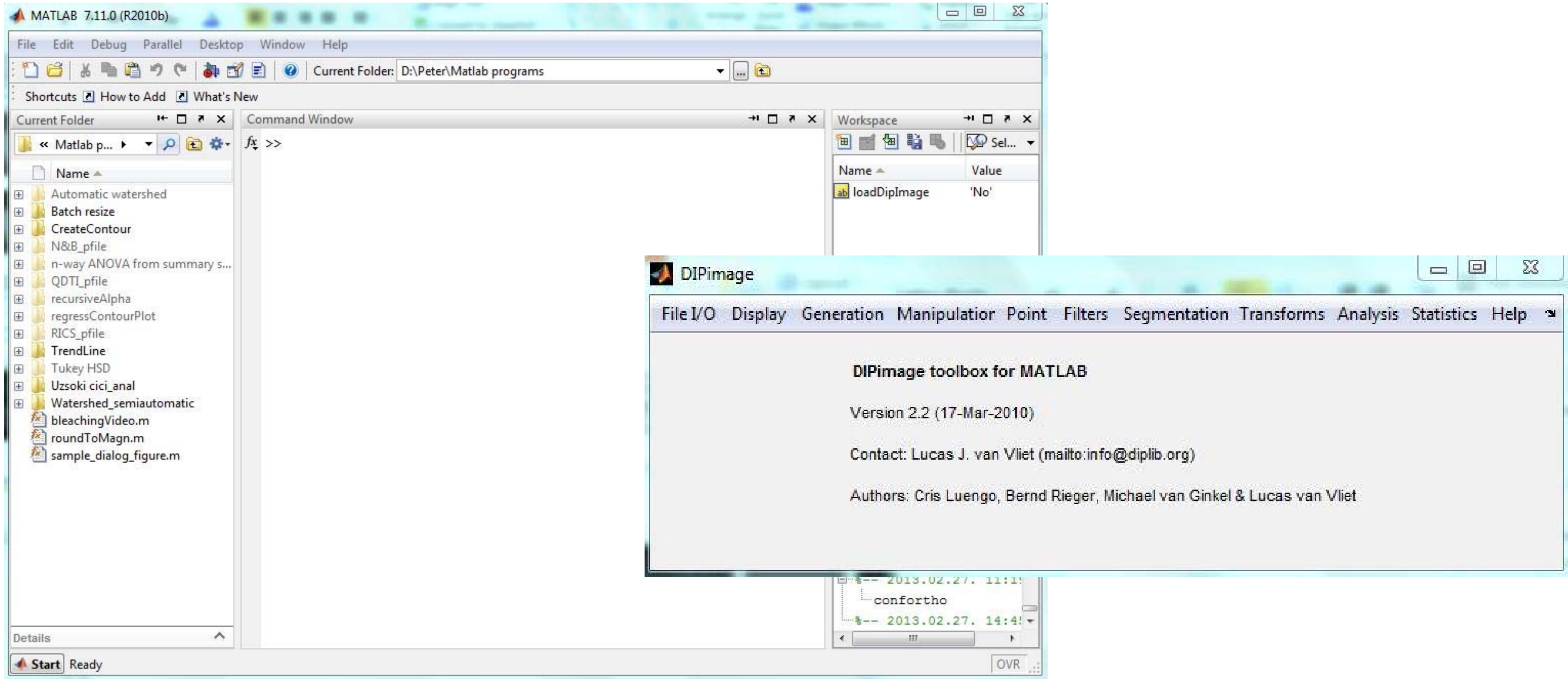
- The most commonly known programs cannot be used for scientific image processing (Powerpoint, Adobe Photoshop, Corel Photopaint)
- The number of scientific image analysis programs is high, here only a few examples are given.



ImageJ 1.47k; Java 1.6.0_20 [64-bit]; 651 commands; 67 macros

- ImageJ (rsbweb.nih.gov/ij/)
  - free, biology-oriented, regularly upgraded, user friendly
  - customizable and extendable with plugins (rsbweb.nih.gov/ij/plugins/index.html)
  - multi-level usage:
    - GUI
    - recording macros (saving a series of commands available from the menu)
    - writing Java programs (plugin)

Scientific image analysis programs


MATLAB 7.11.0 (R2010b)
File Edit Debug Parallel Desktop Window Help
Shortcuts How to Add What's New
Current Folder
Command Window
Workspace
Details
Start Ready
OVR
DIPimage
File I/O Display Generation Manipulation Point Filters Segmentation Transforms Analysis Statistics Help

Current Folder: D:\Peter\Matlab programs

```
fx >>
```

Name
- Automatic watershed
- Batch resize
- CreateContour
- N&B_pfile
- n-way ANOVA from summary s...
- QDTI_pfile
- recursiveAlpha
- regressContourPlot
- RICS_pfile
- TrendLine
- Tukey HSD
- Uzsoki cici_anal
- Watershed_semiautomatic
- bleachingVideo.m
- roundToMagn.m
- sample_dialog_figure.m

Name	Value
loadDipImage	'No'

```
%-- 2013.02.27. 11:1
  confortho
%-- 2013.02.27. 14:4
```

DIPimage toolbox for MATLAB

Version 2.2 (17-Mar-2010)

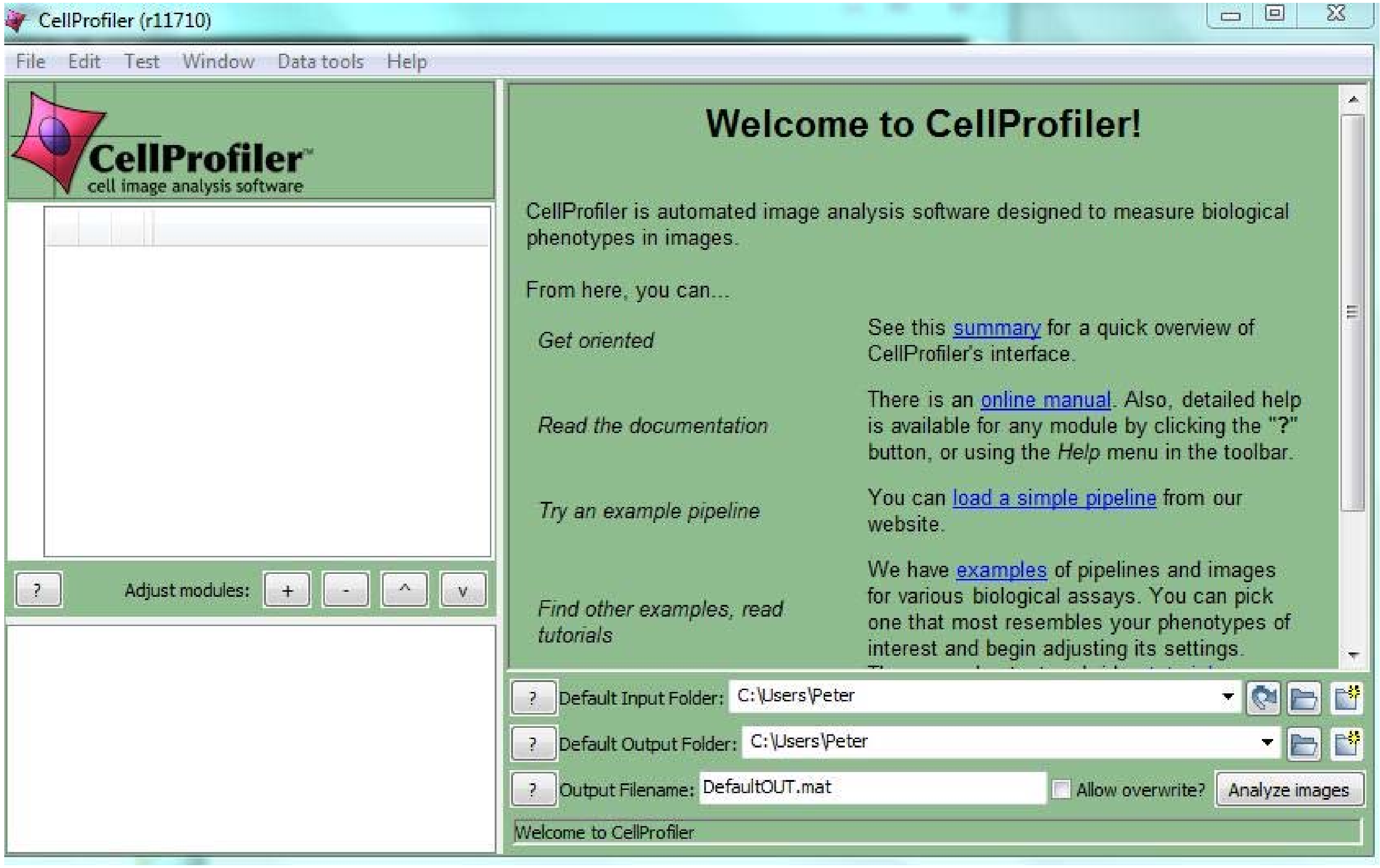
Contact: Lucas J. van Vliet (mailto:info@diplib.org)

Authors: Cris Luengo, Bernd Rieger, Michael van Ginkel & Lucas van Vliet

- Matlab (www.mathworks.com/products/matlab/)
  - not free
  - command-line input → not user friendly
  - extendable with programs written in its highly efficient programming language (M-files) many of which are downloadable: www.mathworks.com/matlabcentral/
  - impossible is nothing with Matlab…
- DipImage (http://www.diplib.org/)
  - a toolbox installed under Matlab
  - GUI-controlled + extends the command collection of Matlab with special image analysis functions

Scientific image analysis programs

- Cell Profiler (www.cellprofiler.org/)
  - free
  - an algorithm consisting of commonly used image processing tools can be built using an menu-controlled system with which an arbitrary number of images can be analyzed automatically (high-throughput image analysis)



Welcome to CellProfiler!

CellProfiler is automated image analysis software designed to measure biological phenotypes in images.

From here, you can...

Get oriented	See this summary for a quick overview of CellProfiler's interface.
Read the documentation	There is an online manual. Also, detailed help is available for any module by clicking the "?" button, or using the *Help* menu in the toolbar.
Try an example pipeline	You can load a simple pipeline from our website.
Find other examples, read tutorials	We have examples of pipelines and images for various biological assays. You can pick one that most resembles your phenotypes of interest and begin adjusting its settings.

Default Input Folder: C:\Users\Peter

Default Output Folder: C:\Users\Peter

Output Filename: DefaultOUT.mat

□ Allow overwrite?

Welcome to CellProfiler

Scientific image analysis programs

- Metamorph (www.moleculardevices.com/products/software/meta-imaging-series/metamorph.html)
  - not free
  - microscope control and image analysis
  - user friendly
  - extensive capabilities for image analysis and 3D visualization
- Image Pro (www.mediacy.com)
  - not free
  - microscope control and image analysis
  - user friendly
  - extensive capabilities for image analysis and 3D visualization
- Usually every confocal microscope has its own software with which simple image processing can be carried out.
  - The freely downloadable version of these programs is simplified → only the simples operations can be carried out, but efficiently and in a user friendly way.