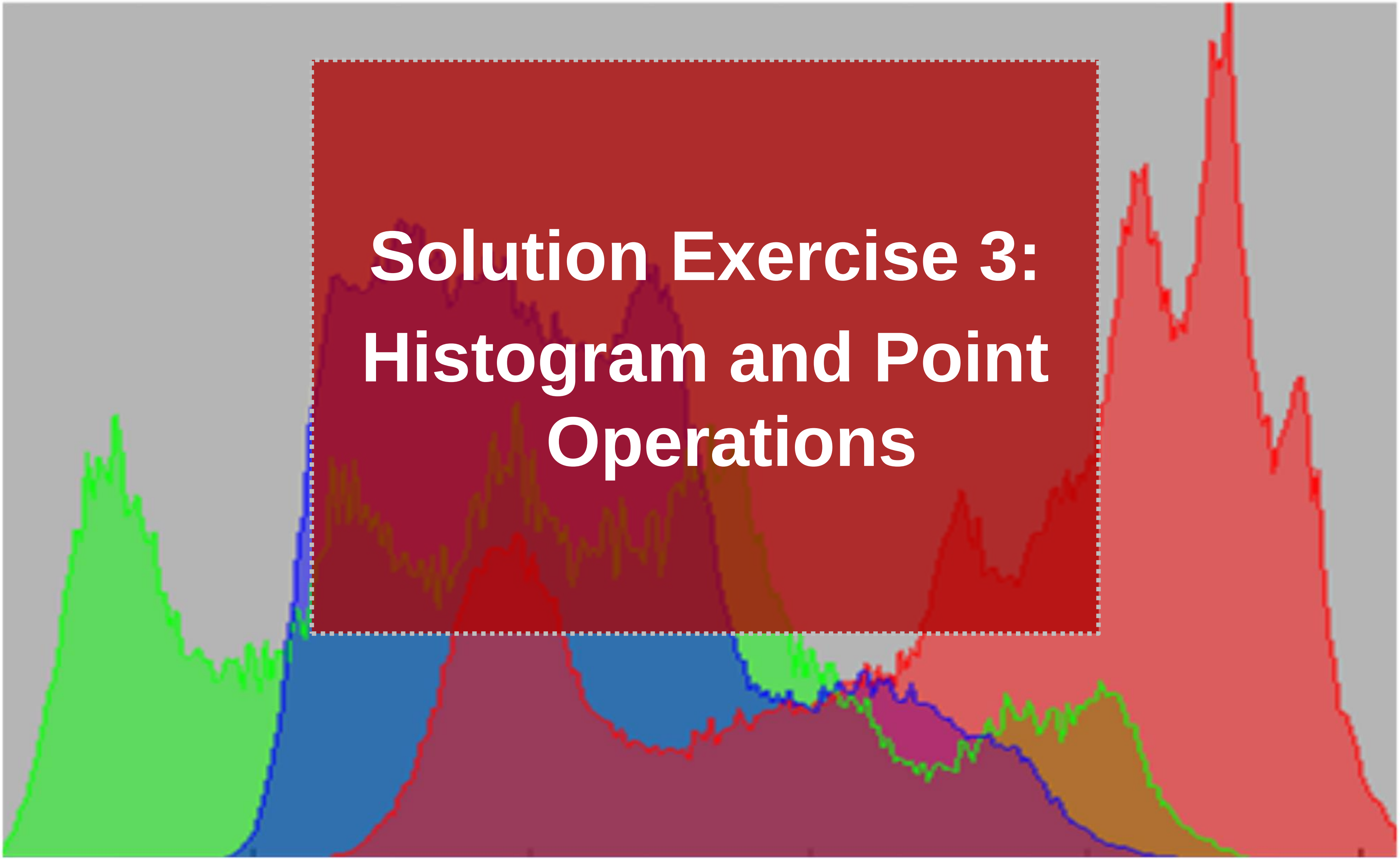
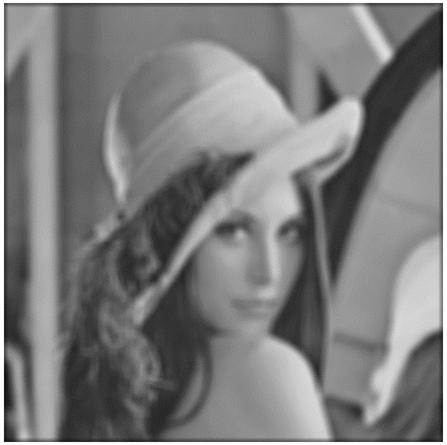
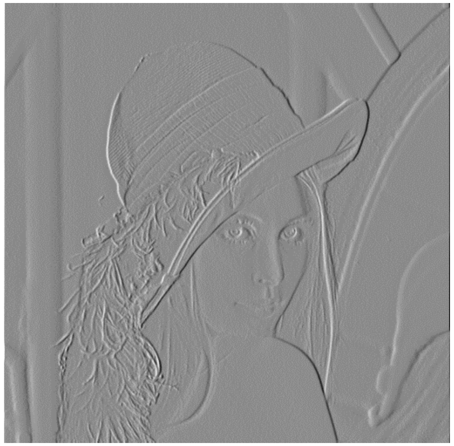


Solution Exercise 3: Histogram and Point Operations

Chapter 4: Spatial Filtering

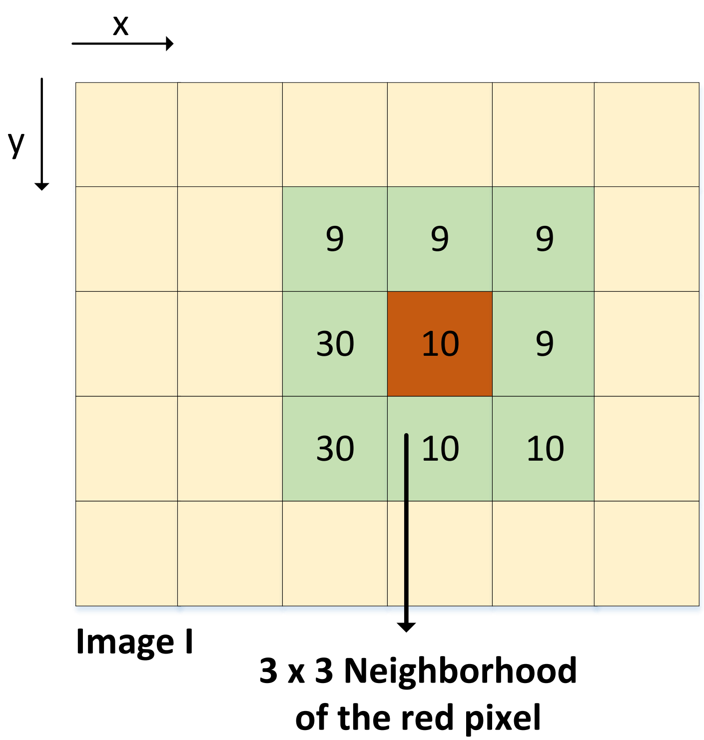




Two dimensional filtering

Filter definition, neighborhood

- A spatial filter is a linear image operation, where the new value of each pixel is influenced by the intensities of the pixels in a defined neighborhood
- The operation can be expressed mathematically as a convolution with a filter mask
- Example of a neighborhood operation:
  Taking the mean in a 3 x 3 neighborhood: What is the mean intensity of the neighborhood of pixel *(y,x)*?



General filter equation

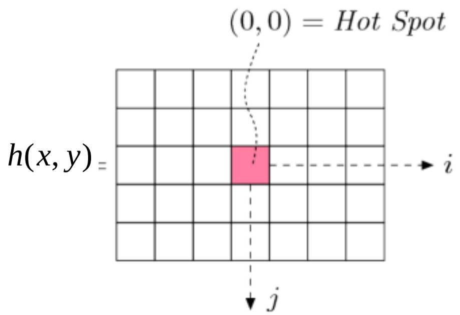
- Convolution is the most important neighborhood operation. It is linear and shift invariant (LSI). If the filter mask is symmetric, convolution is equal to correlation.
- Correlation: The pixel in the output image I'(x,y) is the weighted sum of the input pixel values

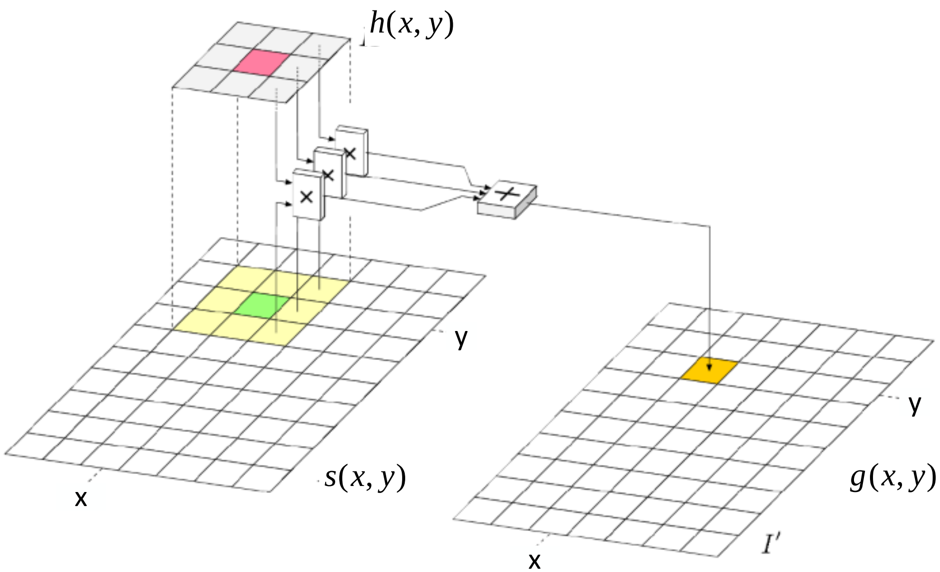
$$g[x,y] = s[x,y] * h[x,y] = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} s[x-i, y-j] * h[i,j]$$

  This can be expressed more compactly as:

  $g = h \otimes s$

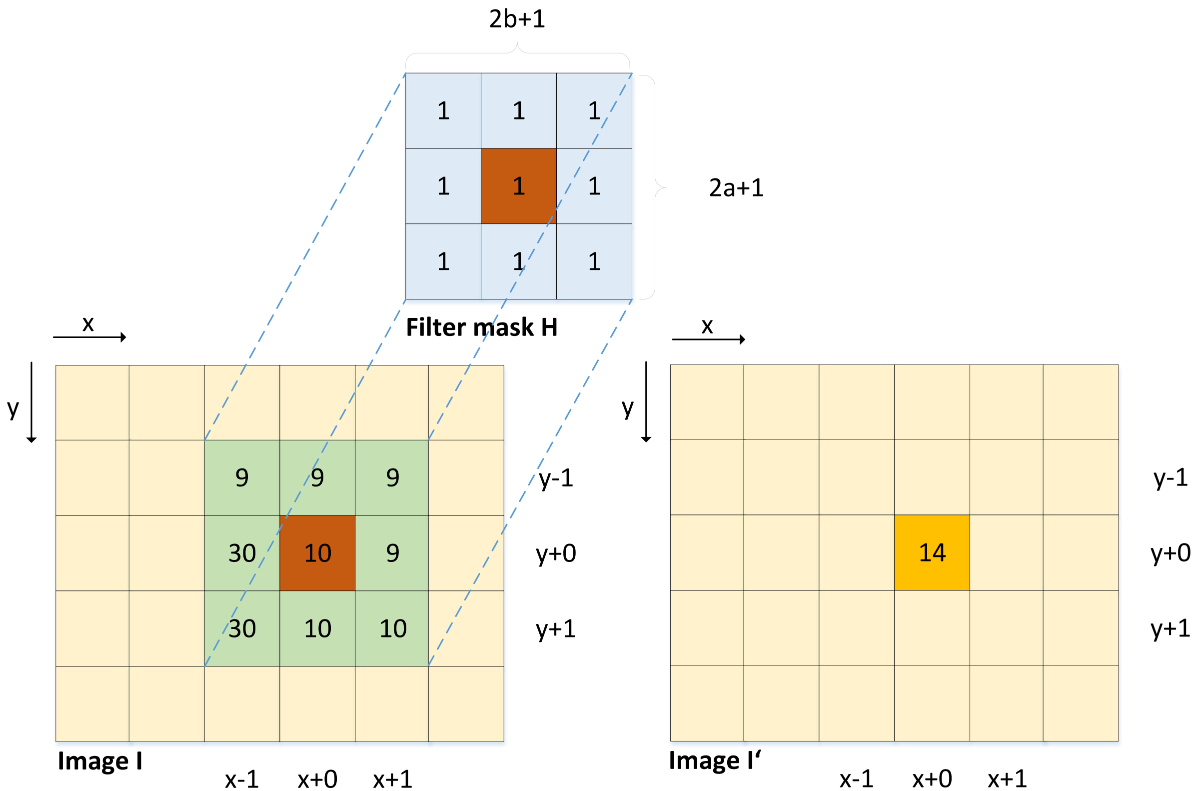
- The number of input pixels is defined by the size of the filter mask
- *h* is the filter mask/matrix/kernel of the height *(2a+1)* and the width *(2b+1)* → usually a = b
- The entries in the filter mask are called filter coefficients





Example

- Filter mask for calculating the mean pixel value

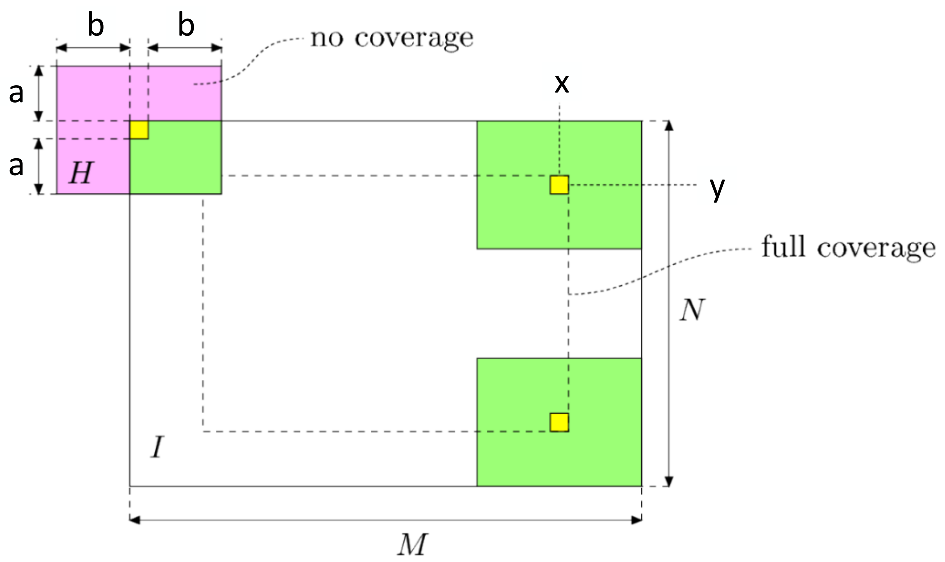


$$g[x,y] = s[x,y] * h[x,y] = \sum_{i=-a}^{a} \sum_{j=-b}^{b} s[x-i, y-j] * h[i,j]$$

$$= \frac{1}{9} \sum_{i=-1}^{1} \sum_{j=-1}^{1} s(x-i, y-j) * 1 = \frac{1}{9}(4 * 9 + 3 * 10 + 2 * 30) = 14$$

In order to not change the average image brightness, the filter matrix has to be multiplied with 1/9 (in this example). This is called normalization.

Boundaries



Solutions:

- Cropping: just apply the filter on pixels where full coverage is guaranteed
- Padding: create a margin around the image and fill it with black pixels, for example
- Mirroring: create a margin around the image and fill it with the pixel values as if a mirror is placed at the edge of the image
- Wrapping: create a margin around the image and restart with the pixel intensities of the image itself
- Extending: create a margin around the image and continue with the intensity of the neighboring pixel
- …

TECHNISCHE UNIVERSITÄT CHEMNITZ

Examples:

Cropping

Padding

Mirroring

Wrapping

Separation of filter mask

- The two-dimensional filtering can be separated into two one-dimensional operations by applying the second operator on the interim result generated by the first operator

- The advantage of this separation: less computing power needed

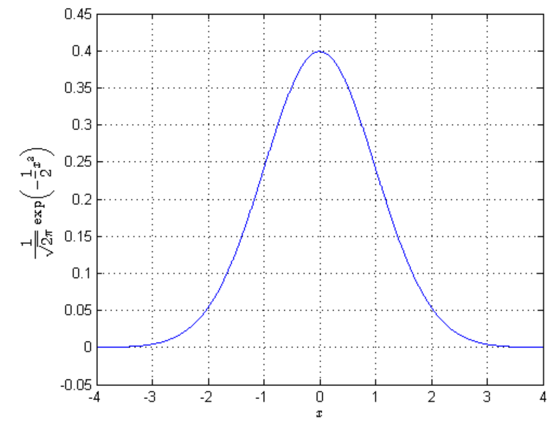
$$h(x,y) = \frac{1}{16} \cdot \begin{bmatrix} 1 \\ 4 \\ 6 \\ 4 \\ 1 \end{bmatrix} * \frac{1}{16} \cdot \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \end{bmatrix} = \frac{1}{256} \cdot \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

- For processing of a 5x5 filter matrix 25 multiplications and 24 summations are necessary

- When the filter is separated:
  - 2 x 5 = 10 multiplications
  - 2 x 4 = 8 summations are needed

Smoothing filters

- Any linear filter with non-negative coefficients
- Compute a weighted average of the image pixels within a certain image region
- Low-pass filter
- Examples:
  - Box
  - Gaussian



2D Gaussian filter

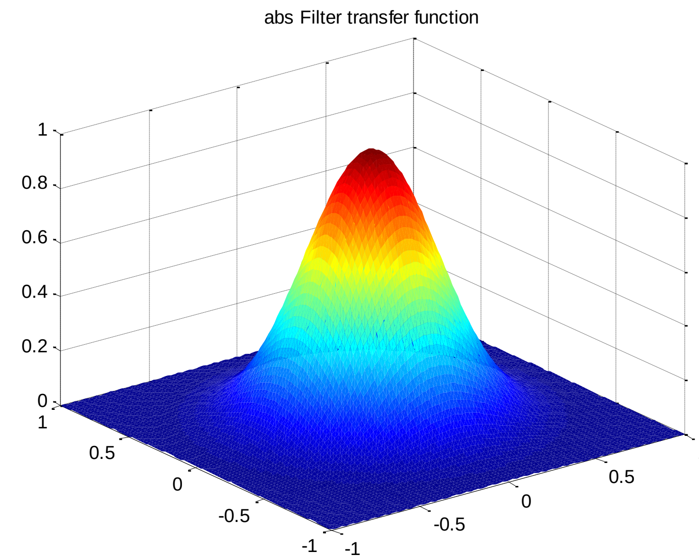
- Corresponding to a discrete, two-dimensional Gaussian function

$$h_\sigma(r) = \exp\left(-\frac{r^2}{2\sigma^2}\right) \quad \text{or} \quad h_\sigma(x,y) = \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

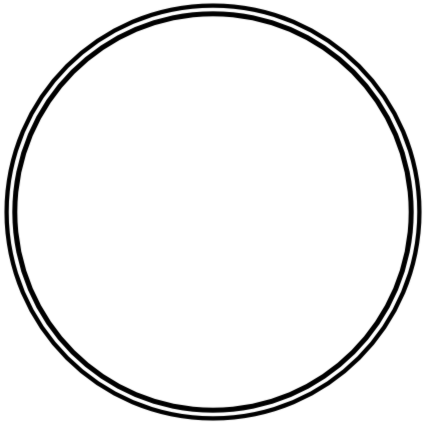
σ - width of the bell-shaped function
r - distance from the center

- Separable into a pair of one-dimensional filters

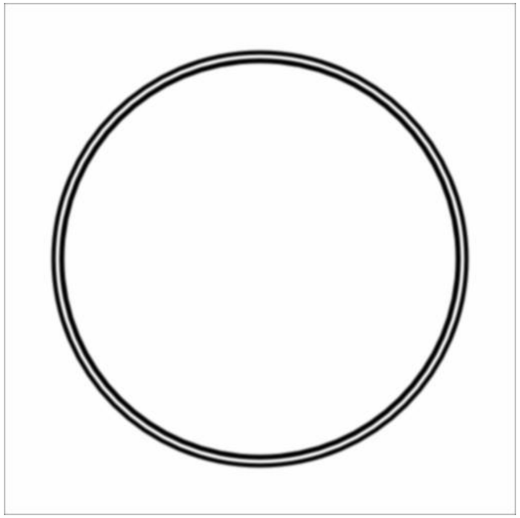
$$h_\sigma(x,y) = \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right) = \exp\left(-\frac{x^2}{2\sigma^2}\right)\cdot\exp\left(-\frac{y^2}{2\sigma^2}\right) = h_\sigma(x)\cdot h_\sigma(y)$$



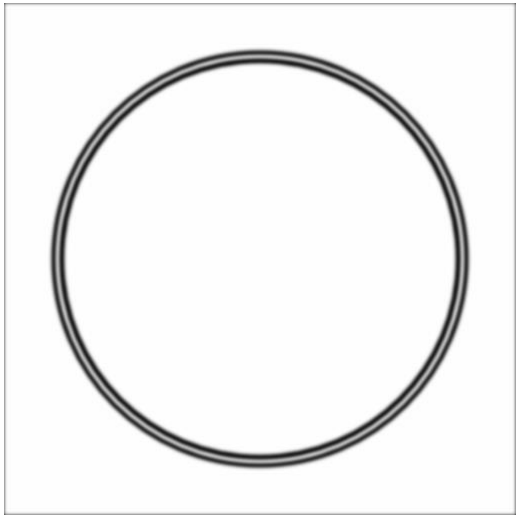
Repetition Filter types



No filter



Gaussian filter size 5



Gaussian filter size 11

Binomial filters

- Discrete approximation to Gaussian functions
- Easy to implement
- 2D Binominal filters can also be expressed as two orthogonal 1D Binominal filters
- 1D Binominal filters follow the binominal distribution, see table below
  - E.g. the 1D Binominal filter of order 2 is specified as:

$$h(x_i) = \frac{1}{4}[1 * x_{i-1} + 2 * x_i + 1 * x_{i+1}]$$

  - While the 1D Binominal filter of order 4 is specified as:

$$h(x_i) = \frac{1}{16}[1 * x_{i-2} + 4 * x_{i-1} + 6 * x_i + 4 * x_{i+1} + 1 * x_{i+2}]$$

Order	Scale factor															
0	1								1							
1	1/2							1		1						
2	1/4						1		2		1					
3	1/8					1		3		3		1				
4	1/16				1		4		6		4		1			
5	1/32			1		5		10		10		5		1		
6	1/64		1		6		15		20		15		6		1	
7	1/128	1		7		21		35		35		21		7		1

2D Binomial filters

- The 2D binominal filter can also be expressed as two orthogonal 1D Binomial filters (Binomial of order 2):

$$h(x,y) = \frac{1}{4}\begin{bmatrix}1\\2\\1\end{bmatrix} * \frac{1}{4}\begin{bmatrix}1 & 2 & 1\end{bmatrix} = \frac{1}{16}\begin{bmatrix}1 & 2 & 1\\2 & 4 & 2\\1 & 2 & 1\end{bmatrix}$$

$\frac{1}{16} \cdot$

1	2	1
2	4	2
1	2	1

- The 2D Binomial filter of 4th order:

$$h(x,y) = \frac{1}{16}\begin{bmatrix}1\\4\\6\\4\\1\end{bmatrix} * \frac{1}{16}\begin{bmatrix}1 & 4 & 6 & 4 & 1\end{bmatrix} = \frac{1}{256}\begin{bmatrix}1 & 4 & 6 & 4 & 1\\4 & 16 & 24 & 16 & 4\\6 & 24 & 36 & 24 & 6\\4 & 16 & 24 & 16 & 4\\1 & 4 & 6 & 4 & 1\end{bmatrix}$$

$\frac{1}{256} \cdot$

1	4	6	4	1
4	16	24	16	4
6	24	36	24	6
4	16	24	16	4
1	4	6	4	1

Derivative filters

General description

- Local changes are highlighted, even noise → edge detection
- High-pass filter
- Examples: Prewitt, Sobel, Roberts

$$\frac{df}{dx}(x) \approx \frac{f(x+1) - f(x-1)}{2}$$

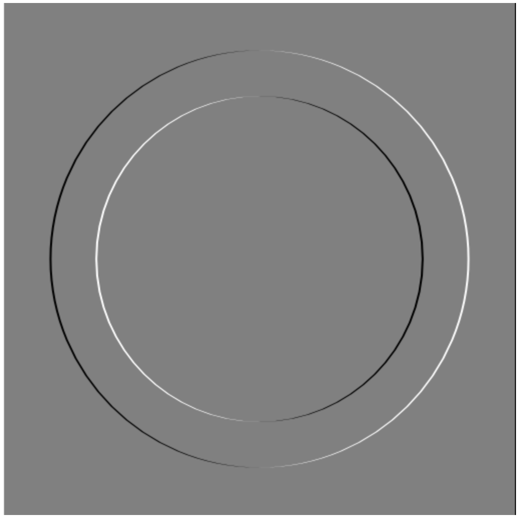
- The gradient vector of function *I* at position *(y,x)*:

$$\nabla I(y,x) = \begin{bmatrix} \frac{\partial I}{\partial x}(y,x) \\ \frac{\partial I}{\partial y}(y,x) \end{bmatrix}$$

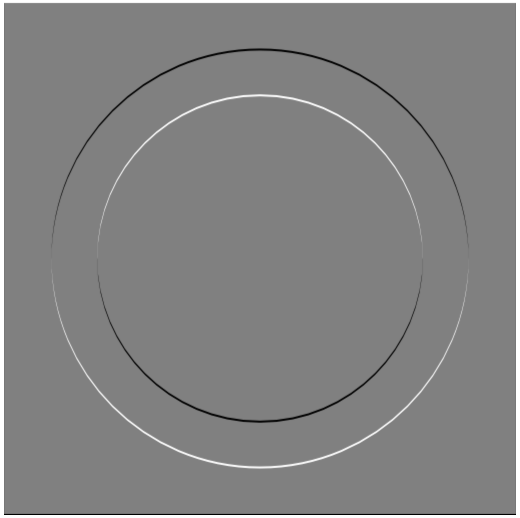
- The magnitude of the gradient:

$$\left|\nabla I(y,x)\right| = \sqrt{\left(\frac{\partial I}{\partial x}(y,x)\right)^2 + \left(\frac{\partial I}{\partial y}(y,x)\right)^2}$$

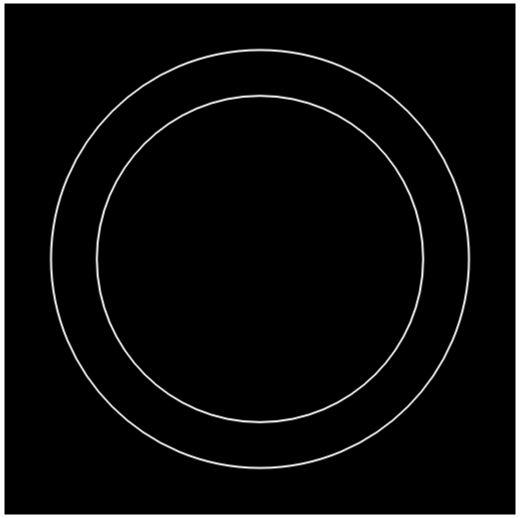
  – invariant under image rotation
  – important for isotropic localization of edges → basis of many edge detection methods



Derivate in *x* (horizontal) direction



Derivate in *y* (vertical) direction



Magnitude of the gradient

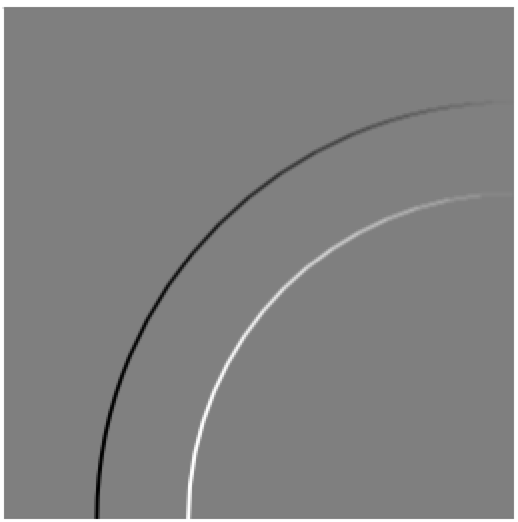
In the two left images:
lowest (negative) values are shown black
maximum (positive) values are white
zero values are gray

Sobel operator

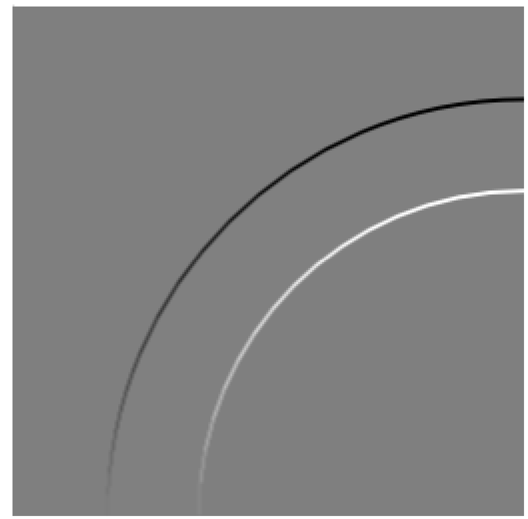
- Combination of derivation and smoothing (Gaussian filter)
- Higher weight to the current center line and column, respectively

$$h = \frac{1}{8}\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

$$h = \frac{1}{8}\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$



x direction - derivation
y direction - smoothing



x direction - smoothing
y direction - derivation

Fourth exercise

- Implement a Gaussian and a Sobel filter

Expected Output (1)

Original Image

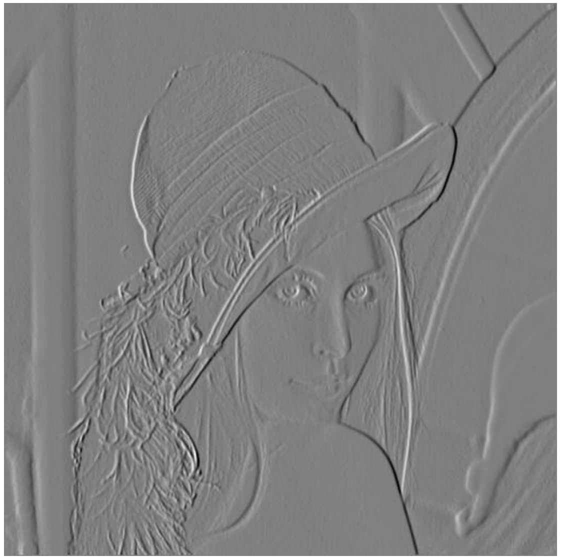
Gaussian Blur with 5x5 Filter

Expected Output (2)

Original Image

Abs of Sobel Images below

X-filtered Image with 3x3 Sobel Kernel



Y-filtered Image with 3x3 Sobel Kernel

That is all for today.