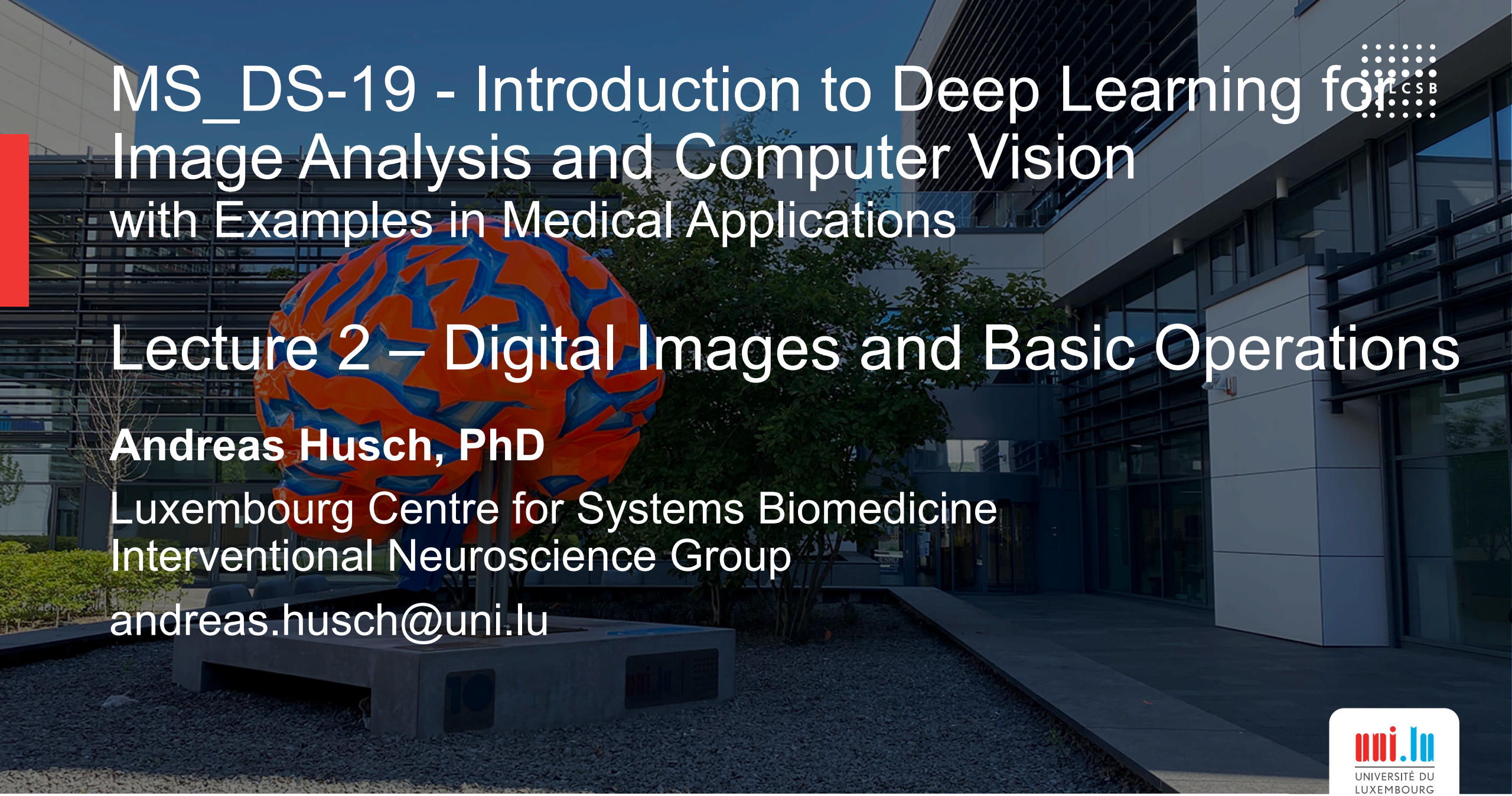


MS_DS-19 - Introduction to Deep Learning for Image Analysis and Computer Vision

with Examples in Medical Applications

Lecture 2 – Digital Images and Basic Operations

Andreas Husch, PhD

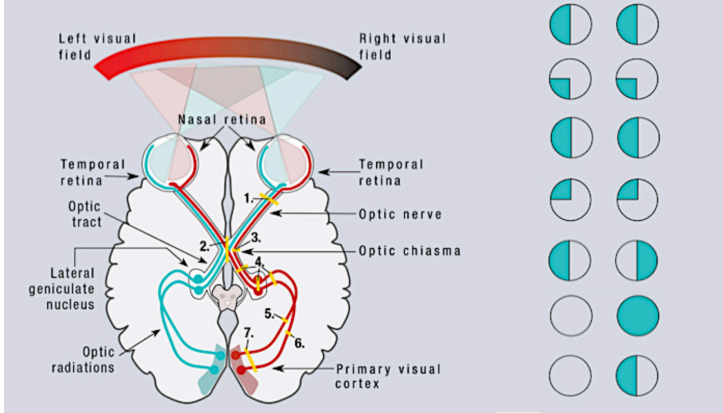
Luxembourg Centre for Systems Biomedicine
Interventional Neuroscience Group

andreas.husch@uni.lu

UNIVERSITÉ DU LUXEMBOURG

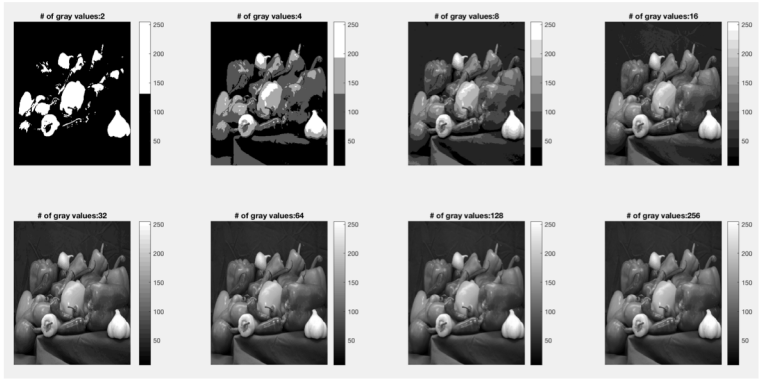
Recap last lecture

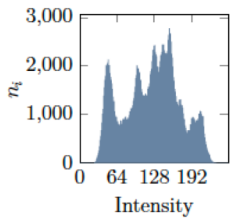
- Human Visual System
- Images as Functions
  - Multiple equivalent discrete representations
- Indexed Images
  - Colormaps (Look-up-table)
- Sampling and Quantization
  - Resolution
  - Bit-Depth
- Histograms
  - Look-up-tables
  - Window/Level

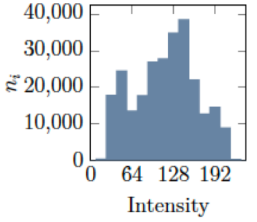


$B_t = (r,g,b)(x,y), \text{with } r,\ g,\ b \in \{0,..,255\}$

$B_c = b(x,y,c) \text{ with } b \in \{0,...,255\} \text{ and } c \in \{1,2,3\}$







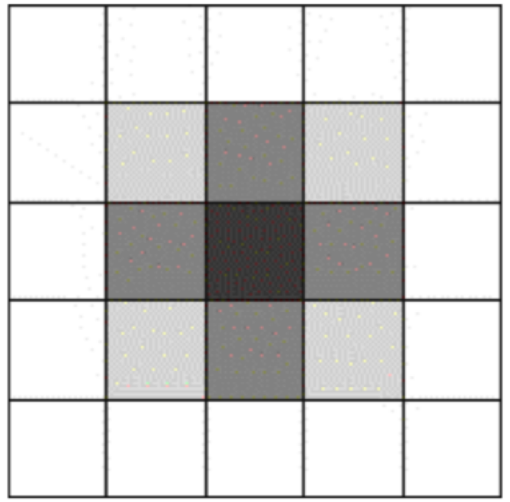
(a) Example grayscale image.

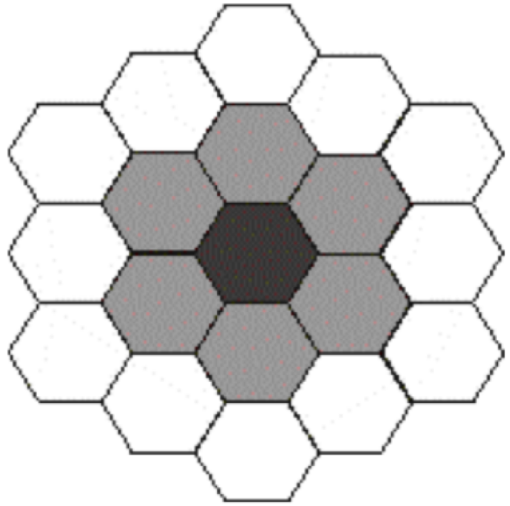
(b) Histogram of the example image with 256 bins.

(c) Histogram of the example image with 16 bins.

Image Grids and Neighborhood relations

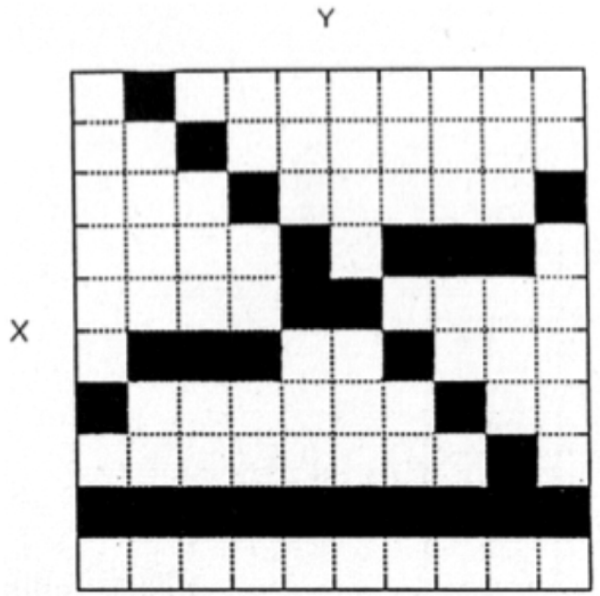
- **Cartesian grids**
  - **Square** grid: simplest and most common form; regular grid over the image.
  - **Non-Square** (rectangular) grid: **anisotropic pixels**: $\Delta x \neq \Delta y$
    - frequent in medical imaging! (why?)
- **Non-Cartesian**, for example **Hexagonal grid**: Each pixel has exactly six neighbors.
  - Neighboring pixels have exactly one common edge.
  - Discrete geometry is simpler (distances, contour length, area of objects, etc.).





Digital Image Grids

- Strengths and weaknesses of discrete geometries: e.g. on the problem of connected components



Neighborhood (in 2D)

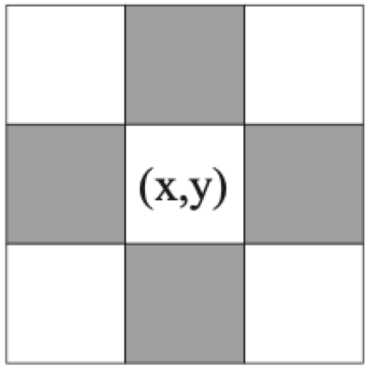
4-neighborhood: The four direct neighbors of a central pixel

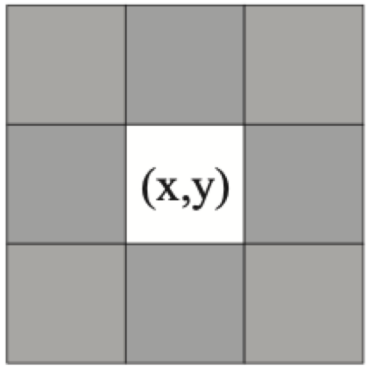
- central pixel (x,y) form the 4-neighborhood $N4(x,y)$. Coordinate list:

$$N4 = \{(x,y\ -1),(x\ -1,y),(x,y\ +1),(x\ +1,y)\}\ .$$

8-neighborhood: The four direct neighbors plus the four diagonal neighbors of a central pixel (x,y) form the 8-neighborhood $N8(x,y)$

$$N8 = \{(x\ -1,y\ -1),(x\ -1,y),(x\ -1,y\ +1),$$
$$(x,y\ -1),(x,y\ +1),(x\ +1,y\ -1),(x\ +1,y),(x\ +1,y\ +1)\}$$

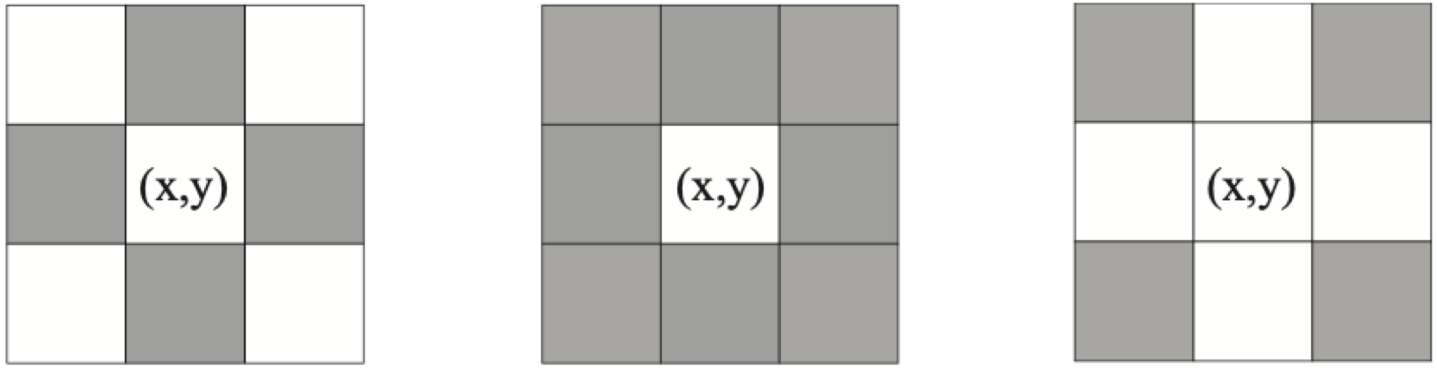




Neighborhood (in 2D)

D-neighborhood: The **diagonal** neighbors of a central pixel (x,y) for the D-neighborhood $\mathrm{ND}(x,y)$

$$ND = \{(x-1,y-1),(x-1,y+1),(x+1,y-1),(x+1,y+1)\}$$



Neighborhood – other dimensions

How does the equivalent of a 4-Neihborhood look

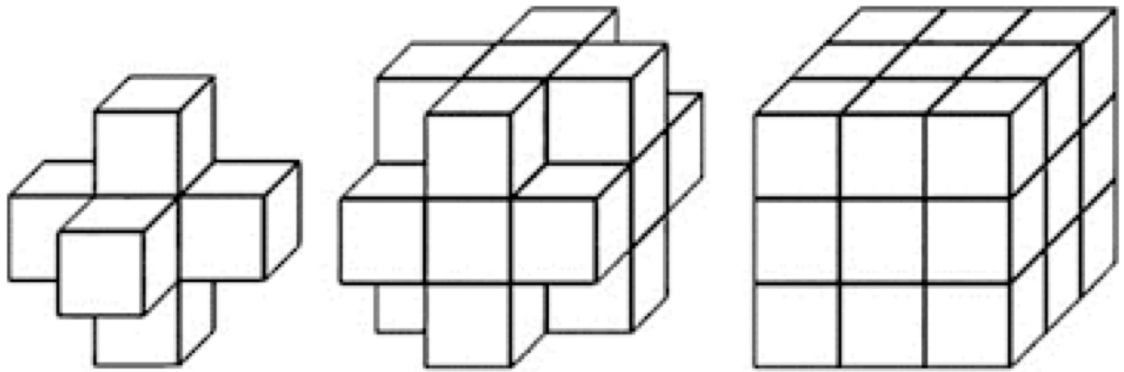
- In 1D?
- In 3D?

How does the equivalent of a 8-Neihborhood look

- In 1D?
- In 3D?

What about applications

in Deep Learning?

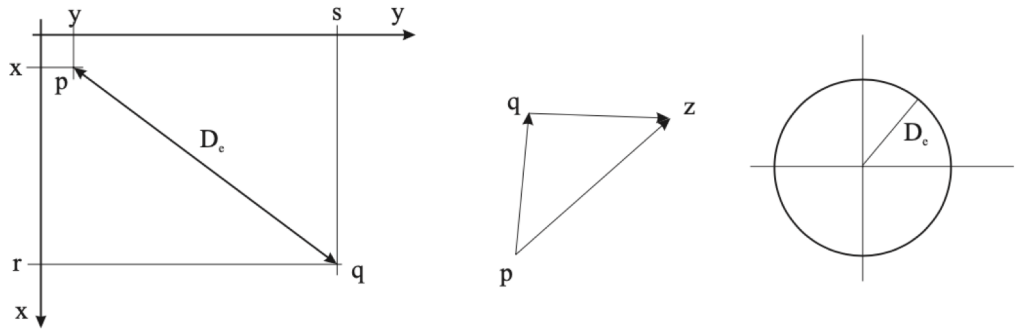


Distanc

- Geometric aspects: Distance measures
  - Measure of the distance $\mathrm{d(p,q)}$ of two points $\mathrm{p=(x,y),q=(r,s)}$ in a square grid.
  - Simple metric: Euclidean distance with

$$d_e(p,q) = \sqrt{(r-x)^2 + (s-y)^2}$$

  - Direct neighbors $\mathrm{q} \in \mathrm{N_4}$ of p have distance $\mathrm{d_e} = 1$ (standardized grid size with edge length of one unit length)
  - Diagonal neighbors q ∈ ND then have Euclidean distance $d_e = \sqrt{2}$.



Distances

- (Discrete) metrics must satisfy the triangle inequality;
- with the image points $\mathrm{p} = (\mathrm{x},\mathrm{y})$, $\mathrm{q} = (\mathrm{r},\mathrm{s})$ and $\mathrm{z} = (\mathrm{u},\mathrm{v})$ we obtain a a proper distance function or metric $\mathrm{D_x}$ when

$$
\begin{aligned}
D_x(p,q) &\geq 0 \\
D_x(p,q) &= 0 \quad \textit{if} \quad p = q \\
D_x(p,q) &= D_x(q,p) \\
D_x(p,z) &\leq D_x(p,q) + D_x(q,z)
\end{aligned}
$$

Distances

- This allows simpler (integer number) metrics to be defined for image processing:
- D_4-distance: D_4 distance or absolute metric of points $p=(x,y), q=(r,s)$ is the shortest N_4 path between p and q:

$$d_4(p,q)=|r-x|+|s-y| \ .$$

Points of the same distance lie on the contour of a rhombus

- D_8-distance: D_8 distance or checkerboard metric **('city block')** of points p,q is shortest N_8 path between p and q:

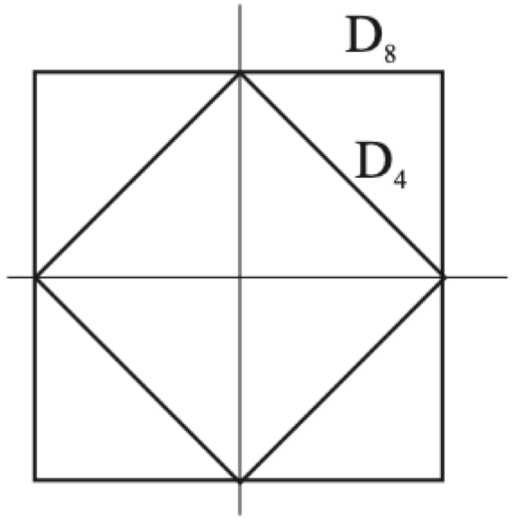
$$d_8(p,q)=\max(|r-x|,|s-y|) \ .$$

Points of equal distance lie on the contour of a square

D_8 and D_4

4	3	2	3	4
3	2	1	2	3
2	1	0	1	2
3	2	1	2	3
4	3	2	3	4

2	2	2	2	2
2	1	1	1	2
2	1	0	1	2
2	1	1	1	2
2	2	2	2	2



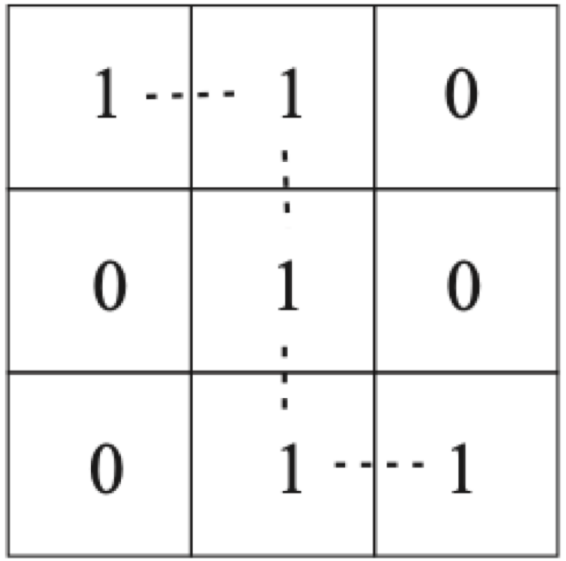
Topological aspects

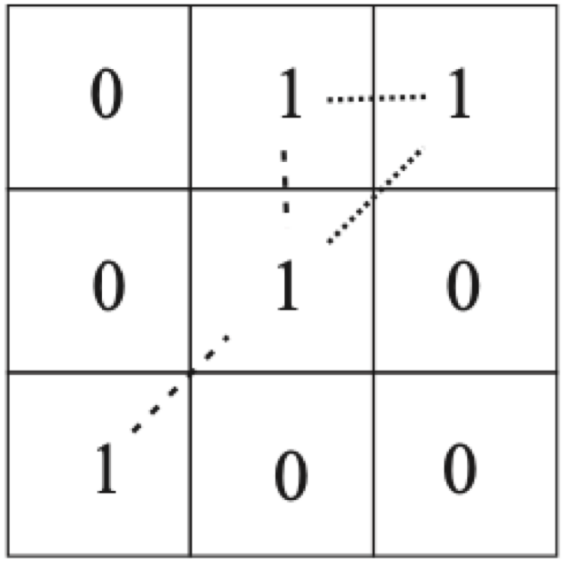
- Topological aspects of the neighborhood of a pixel:
- **Neighboring** pixels are examined to determine **relationships**: *is a pixel “similar” and ”neighboring”?*
- **Similarity conditions** can be defined by a *predicate P* (e.g. *P: gray values f in an interval V =[fmin,...,fmax]).*
  - In practise, one compares a pixel p with the pixels q from the neighborhood N(p):is similarity criterion fulfilled? (P = true |f (p), f (q) ∈ V). If yes, then the found pixels p,q form a *N-connection* (**connected component**).
- Connections can be defined by means of so-called adjacencies (r = 4, 8, m):

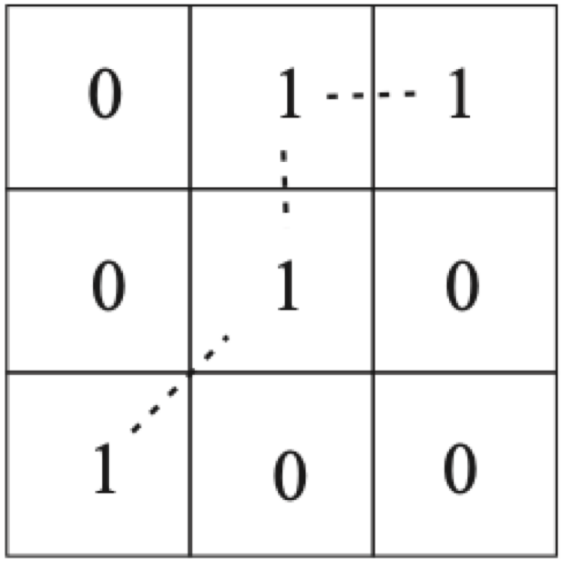
Adjacency

- 4-adjacency: Two pixels p and q with values within V are 4-adjacent if q is in N4(p).
- D-adjacency: Two pixels p and q with values within V are adjacent if q is in ND(p).
- 8-adjacency: Two pixels p and q with values within V are 8-adjacent if q lies in N8(p).
- m-adjacency: Two pixels p and q with values within V are adjecent if
    - 1. q is in N4(p) or
    - 2. q is in ND(p) and the set N4(p)∩N4(q) has no pixels with values from V
      .
- The m adjacency avoids inconsistencies (e.g., multiple assignments in the case of 8 adjacency).

Adjacencies of pixels







Connected components

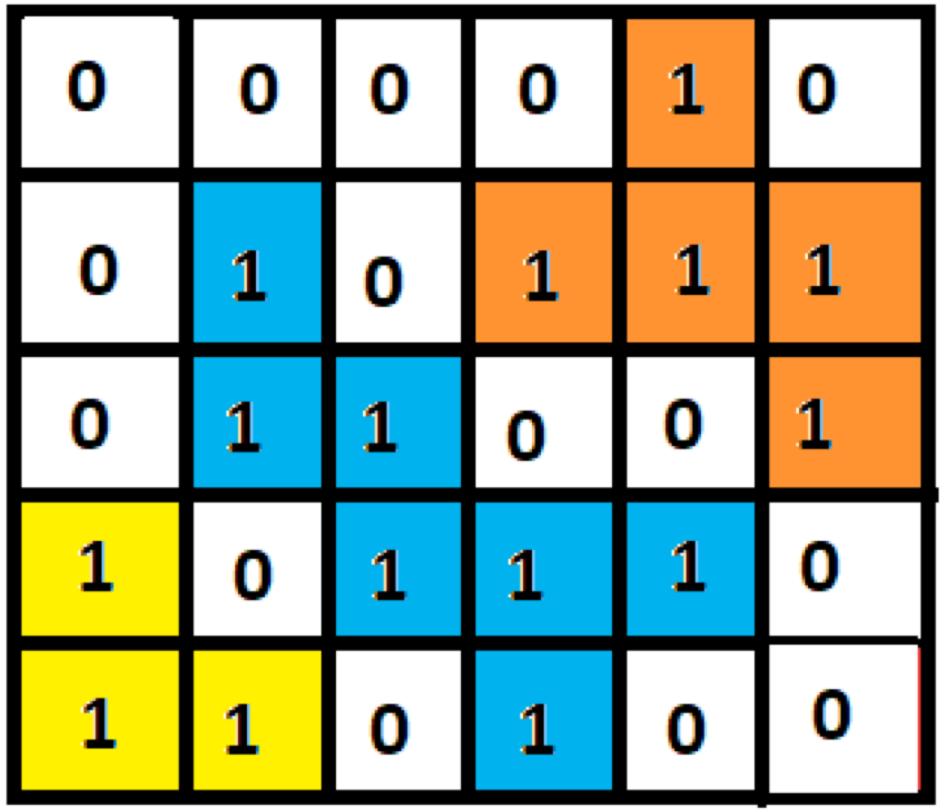
Digital path: A digital path $S = \overline{pq}$ between two points $\mathrm{p}=(\mathrm{x},\mathrm{y})$ and $\mathrm{q} = (\mathrm{r},\mathrm{s})$ is given by the sequence of certain points:

$$S = \overline{pq} = [(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)]$$

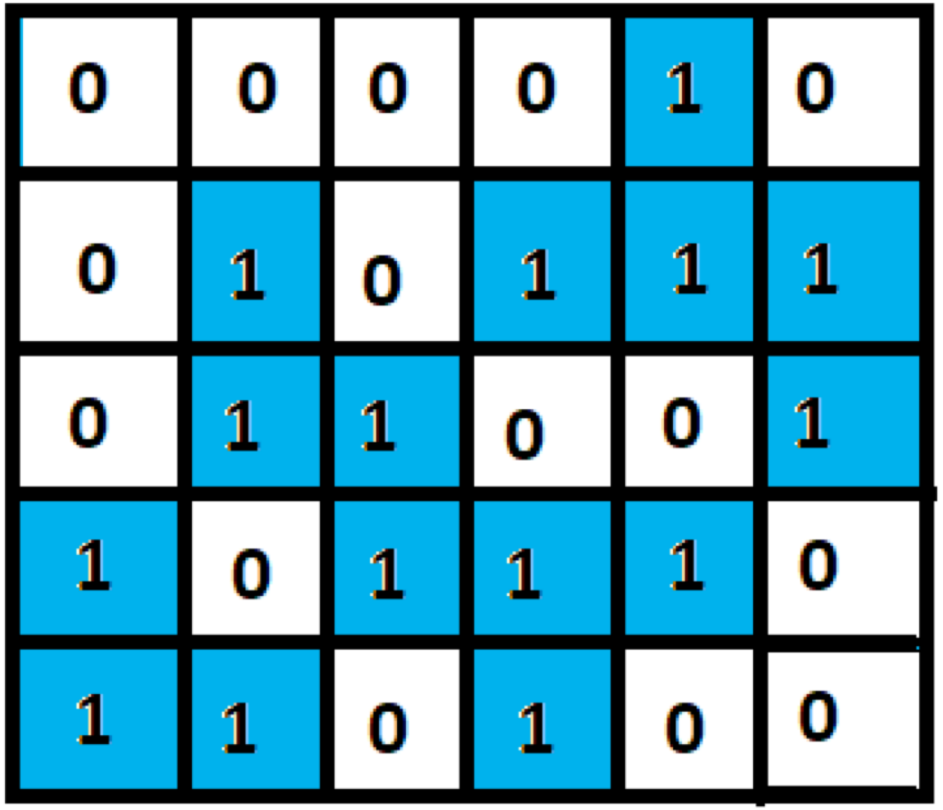
where $\mathrm{p}=(\mathrm{x},\mathrm{y})=(\mathrm{x}_0,\mathrm{y}_0)$ and $\mathrm{q} =(\mathrm{r},\mathrm{s})=(\mathrm{x}_\mathrm{n},\mathrm{y}_\mathrm{n})$ and the pixels $(\mathrm{x}_{\mathrm{i}-1},\mathrm{y}_{\mathrm{i}-1})$ and $(\mathrm{x}_\mathrm{i},\mathrm{y}_\mathrm{i})$ are adjacent (4,8 or m) for $1 \leq \mathrm{i} \leq \mathrm{n}$.

Connected component: let S be a set of pixels of an image, then two pixels p,q are connected in S if there is a path $\overline{pq}$ that is entirely in S. For each pixel p in S, the set of all pixels in S that are connected to it is called a connectivity component of S.

Connectivity induced by a neighborhood relation



4-connectivity map



8-connectivity map

Connected components II

- Connectivity relation Z

$$Z \quad : \quad pZq \Leftrightarrow \exists \overline{pq}$$

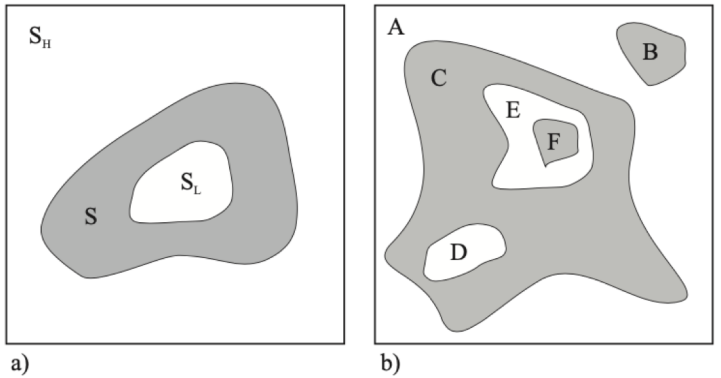
$$S_Z \quad : \quad \{p, q \mid pZq\}$$

$$pZq \quad : \quad \text{if } \exists\ \overline{pq} \text{ with } \overline{pq} \in S$$

- Z is an equivalence relation; thus, pZp (reflexive) holds,
- $\mathrm{pZq} \rightarrow \mathrm{qZp}$ (symmetric) and
- $\mathrm{pZq} \wedge \mathrm{qZu} \rightarrow \mathrm{pZu}$ (transitive).
- If there is only one correlation component, then it is called a correlation set.

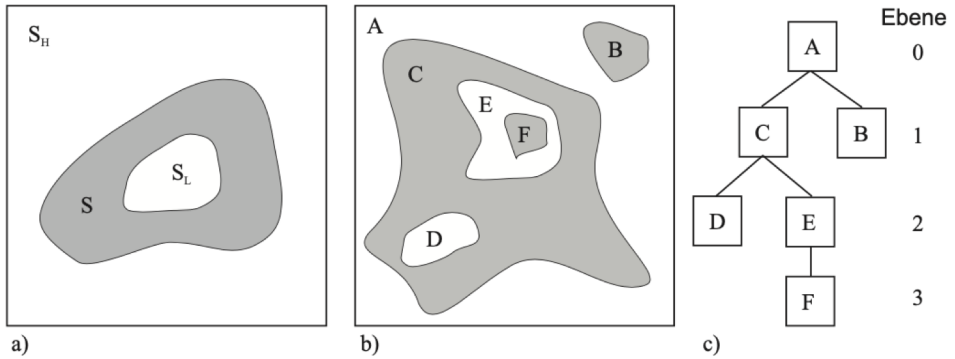
Euler number and adjacency tree I

- Simple description of segmented images: **Objects** and **background** each represented as **connected components**.
- An image then contains the components
  - S (object or silhouette),
  - $\mathrm{S_H}$ (background, touching the edge)
  - $\mathrm{S_L}$ (hole in the object).
- It holds: $\neg\mathrm{S} = \mathrm{S_B} \cup \mathrm{S_{\#l,}}$ image $\mathrm{B} = \mathrm{S} \cup \neg\mathrm{S}$ (segmentation is a *partition*), $\mathrm{S} \subset \mathrm{B}$ and $\mathrm{S} \cap \neg\mathrm{S} = 0$.



Euler number and adjacency tree II

- Surrounding relation U: The relation U says that the silhouette S "surrounds" the hole S_L "surrounds":
  - S U S_L S surrounds S_L ,
    - U is a partial relation,
  - RUS → S¬UR
  - RUS∧SUT → RUT (transitive) .
- Adjacency tree: the relation U can be **described as a tree** with the following level numbering.
  - Level 0 : Image B,
  - odd levels : components S_i
  - even levels : holes S_{L_i}



Euler number and adjacency tree III

- Euler number: A characteristic quantity is the Euler number E, which is the difference of the number C of components minus the number H of holes:

$$\mathrm{E}=\mathrm{C}\text{-}\mathrm{L}$$

- E is a topological property.

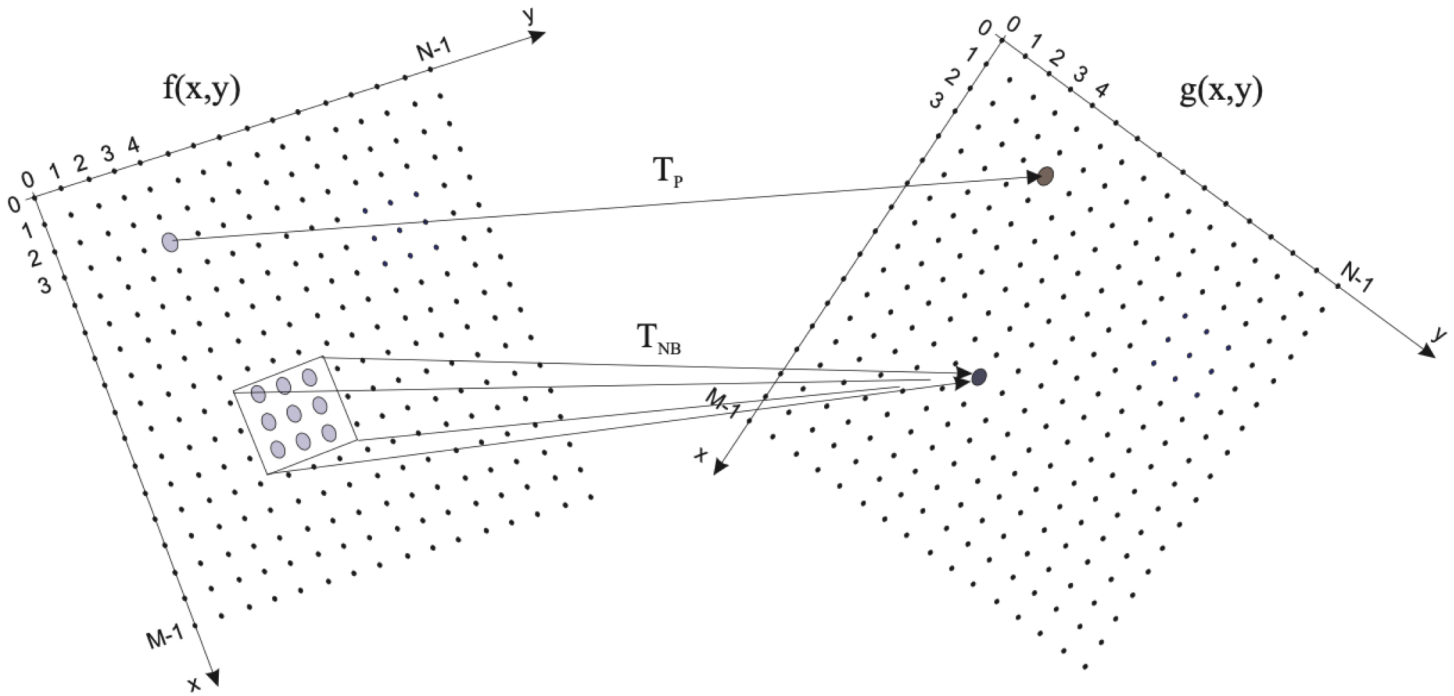
Elementary image operations

- Examples of image operations
  - **point and local** image operations,
    - notice that **local operations** might be applied to the **global image** ☺
      - -> cf. convolutional neuronal networks
  - simple **gray value manipulations** (**intensity** transformations),
  - local **convolution operation** as **generic** image operation
    - -> cf. convolutional neuronal networks
  - simple **geometric transformations**

- Note: Transformations from spatial to frequency domain (FFT) and subsequently application of operators in frequency domain are possible but out of scope of this class. Images are Signals ☺ (and we can consider Signals as Images)

Point and neighborhood operations

- **Point operations** T_p and **neighborhood operations** TN_P are very important operation groups.
  - Mainly linear or nonlinear value range transformations
- **Geometric operations** are often a mixture of the two.

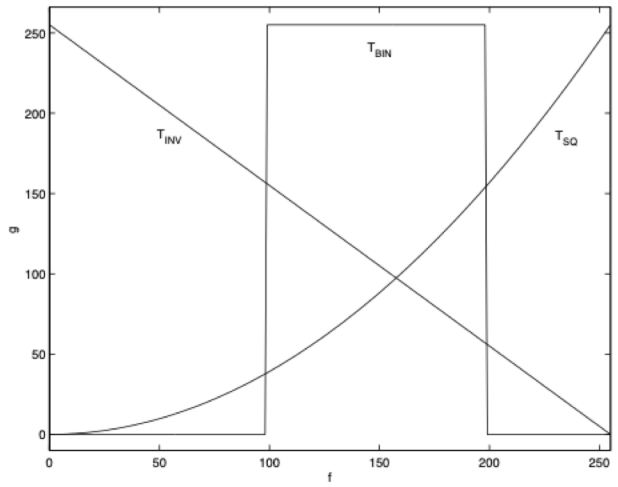


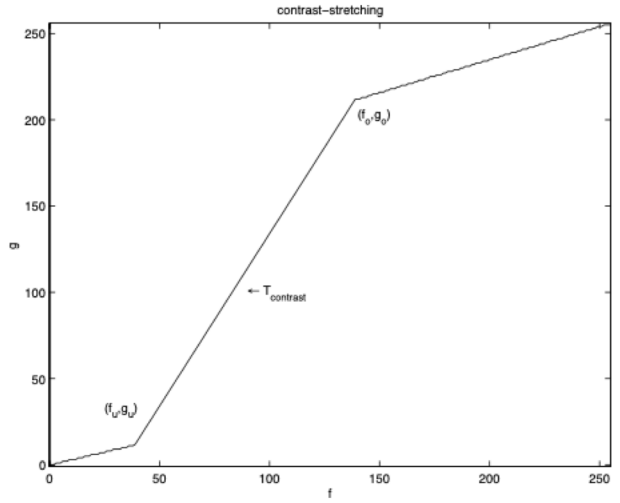
Point operations I

- Point operations: Value **range** (intensity) transformations in the form

$$T_p : f(r,s) \mapsto g(x,y)$$

with $\mathrm{T_p}$: any linear or non-linear transformation g = T_P (f)





- Typical applications are image enhancement (mostly *subjective* but also *objective*) and also segmentation (binary/enumerable image generation).

Point operations II

- Examples:
- Inversion: Brightness values mirrored at the center of the value range:

$$T_{INV} : g := (L-1) - f \quad , f \in [0, L-1]$$

- Binarization: Brightness values in the interval $[f_u, f_o]$ are set to "1" - otherwise to "0".

$$T_{BIN} : g := \begin{cases} 0 & | \quad 0 \leq f < f_u \\ 1 & | \quad f_u \leq f \leq f_o \\ 0 & | \quad f_o < f \leq L-1 \end{cases}$$

Point operations II

- or simply also by means of threshold Θ:

$$T_{BIN} : g := \begin{cases} 1 & | \quad f \geq \Theta \\ 0 & | \quad \text{sonst} \end{cases}$$

- Point-operations consider each “point” of the image range *individually*

Point operations III

- Contrast enhancement: Gray values are stretched (*spread*) or compressed in certain areas.

$$T_{contrast} : g := \begin{cases} c_1 * f & | \quad f \leq f < f_u \\ c * f & | \quad f_u \leq f \leq f_o \\ c_2 * f & | \quad f_o < f \leq L-1 \\ & \text{with } c_1, c_2 < 1,\ c > 1 \end{cases}$$

(general: c > 1: spreading; c < 1: compression)

Local or neighborhood operations I

- Local or so called neighborhood operations consider the function values f (q) of the points q in the *neighborhood* of a pixel $\mathrm{p} = (\mathrm{x},\mathrm{y})$ to generate a new function value g at the point p' (usually: $\mathrm{p}' = \mathrm{p}$).

- for this the definition of a neighborhood $\mathrm{N}(\mathrm{p})$ is needed(e.g.N4,N8 or arbitrary forms for N)

$$T_{NB} : \mathbf{f}(N(x,y)) \mapsto g(x,y)$$

Local or neighborhood operations I

- Example 1: Average over neighborhood $\mathrm{N(p)}$ as N4 neighbor points to the point p

$$
\begin{aligned}
(x,y) &= T_{NB_M}(f(N_4(x,y))) \text{ respectively} \\
g(x,y) &= 1/4 \sum_{\forall(r,s)\in N_4(x,y)} f(r,s) \text{ and tendered to} \\
g(x,y) &= 1/4(f(x,y-1)+f(x,y+1)+f(x-1,y)+f(x+1,y))
\end{aligned}
$$

Local or neighborhood operations II

- Example 2: Pairwise gray value *difference* over the horizontal and vertical neighboring points in N_4, respectively.

$$g(x,y) = T_{NB_D}(f(N_4(x,y)))$$
$$g(x,y) = (f(x,y-1)-f(x,y+1))+(f(x-1,y)-f(x+1,y))$$

- Replacing the signs by factors (±1), we obtain T_{NB_D} in sum form to:

$$g(x,y) = (+1)f(x-1,y)+(-1)f(x+1,y)+$$
$$(+1)f(x,y-1)+(-1)f(x,y+1)$$

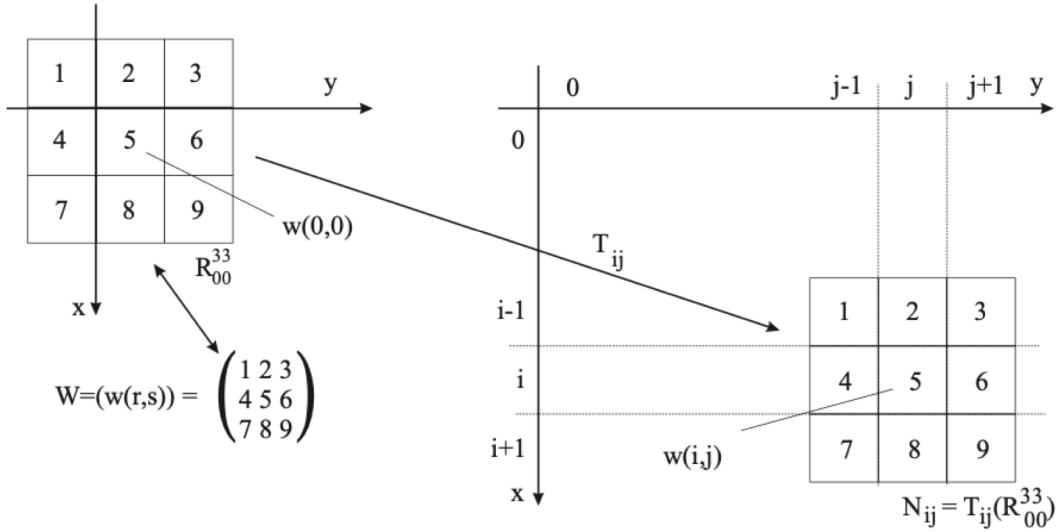
or compact to

$$g(x,y) = \sum_{\forall(r,s)\in N_4(x,y)} w(r,s)\cdot f(r,s)$$

with $w(r,s)$: Weight factor or coefficients for neighboring point $q = (r,s) \in N_4, (p) = N_4(x,y))$

Local or neighborhood operations III

- Base neighborhood with weight values



- Result of the local gray value averaging / difference (see above).

Local or neighborhood operations IV

Typical applications for neighborhood operations

- **local image enhancement**,
  - e.g. *noise reduction* by *local averaging*
  - e.g. *contrast enhancement* based on local gradient
  - e.g. *edge enhancement* by means of Laplace operator
- all forms of **local filtering** (linear, nonlinear) in the spatial domain
- any (special) forms of local operators for segmentation, context analysis etc.

Filtering in local area I

- Operation of the local area filters:

  A filter mask $w(r,s)$ slided over the image $F=f(x,y)$ column by column and row by row.
  For each pixel (x,y) the filter response $g(x,y)$ is calculated linking the filter coefficints $w(r,s)$ with the (underlying) pixels $f(r,s)$

- For linear filters the filter response $G=g(x,y)$ results in

$$g(x,y) = \sum_{\forall(r,s)\in N(x,y)} w(r,s)f(r,s)$$

- This can also be formulated explicitly for a mask W of dimension $m \times n$ via formulate a double sum

$$g(x,y) = \sum_{r=-a}^{a} \sum_{r=-b}^{b} w(r,s)f(x+r,y+s)$$

- with $a = (m - 1)/2$ and $b = (n - 1)/2$ if m, n odd

Filtering in local area II

- The **operation is performed point by point**, i.e. for an image of the dimension $\mathrm{M} \times \mathrm{N}$ there are MN operations (computational time in $O(\mathrm{MN})$!)
- The shifted element of the local neighborhood operation, describing the function $\mathrm{w}(\mathrm{x},\mathrm{y})$ is **referred to as** *filter, mask, kernel, windows or template*.
- The discrete elements describing the filter are called *coefficients*

- *Q: Do you see a connection to convolutional neural networks?*

Filtering in local area III

- Examples:

Contour enhancement with modified Laplace operator (TL):

$$W_{T_L} := \begin{pmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{pmatrix}$$

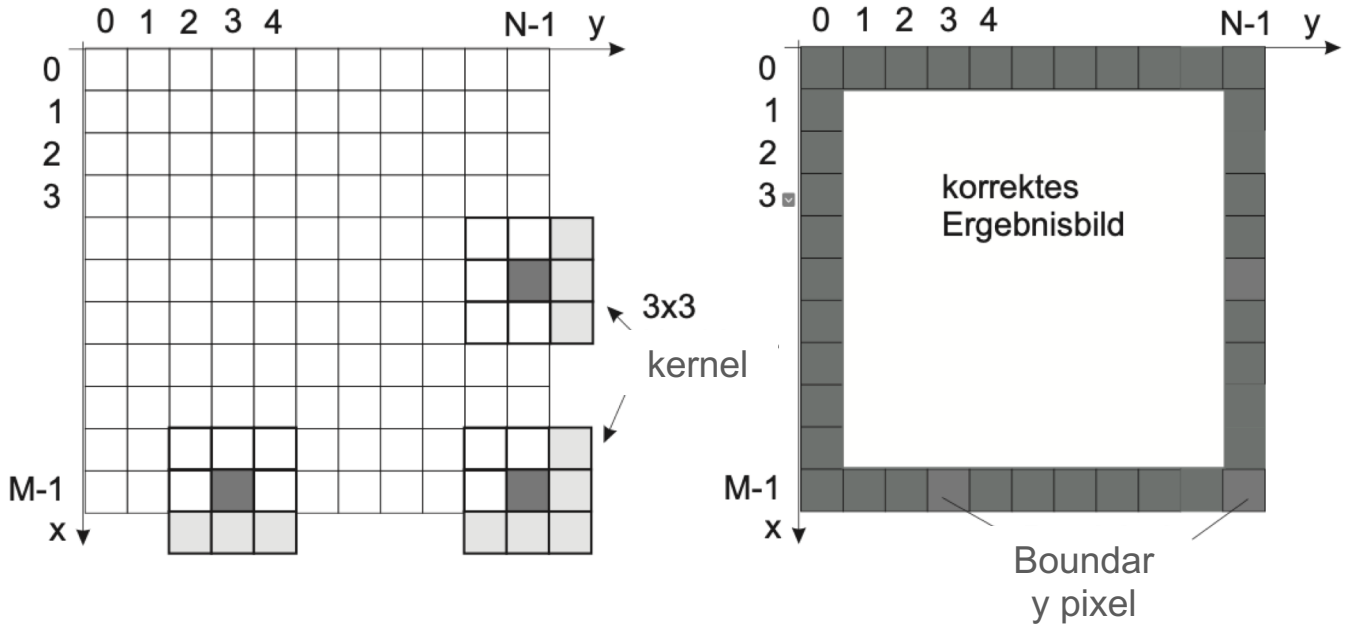
Weighted smoothing (≈Gauss):

Filter masks WTG

$$W_{T_G} := \frac{1}{16} \begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}$$

Filtering in local area IV

- The local operations result in **boundary pixels problems**: overlapping of W with F at all positions q in $\mathrm{M} \times \mathrm{N}$ image not possible!



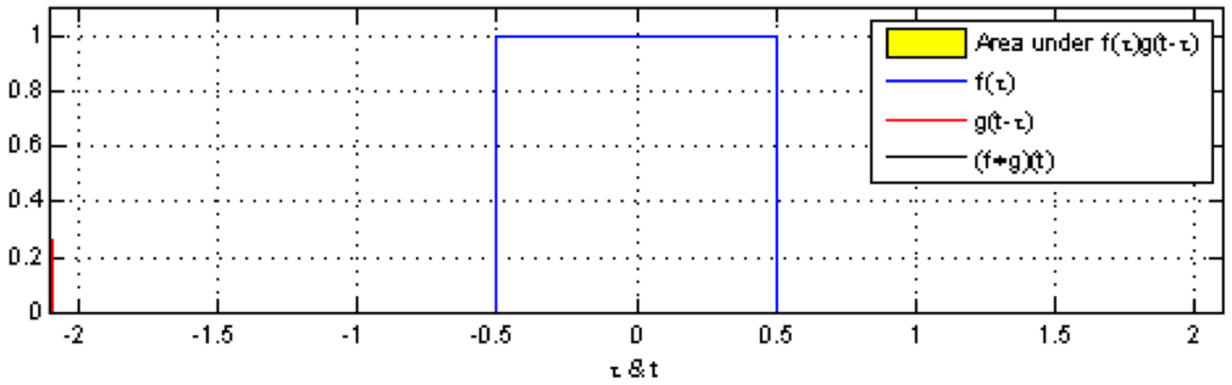
- **Solutions**: *Padding*, *Mirroring* of F at the boundaries, and more. Application dependent!
- *Q: Do you see a connection to convolutional neural networks?*

Convolution theorem I

- Linear filtering, as discussed before, can be describedas the **convolution** of two functions $\mathrm{f(x,y)}$ and $\mathrm{g(x,y)}$ of the dimension $\mathrm{M} \times \mathrm{N}$ with the resulting convolution sum

$$f(x,y) * g(x,y) = \frac{1}{MN} \sum_{r=0}^{M-1} \sum_{s=0}^{N-1} f(r,s) g(x-r, y-s)$$

- this corresponds in the *frequency domain* to the point-wise multiplication of the Frequency functions $\mathrm{F(u,v)}$ with $\mathrm{G(u,v)}$



Convolution theorem I

- Convolution theorem: Convolution of two functions f(x),w(x) in the *spatial domain (time domain)* corresponds to a multiplication of their Fourier transforms F (u), G(u) in the *frequency domain*

$$f(x) \star g(x) \Leftrightarrow F(u) \cdot G(u)$$

- Mirroring of $\mathrm{g(x,y)}$ is often ignored in BV practice, since g(x,y) is mostly symmetric

End Lecture 2 (short) – following slides context was covered in exercise, slides here for reference and self study

Image enhancement (introduction)

- Subjective image enhancement - for the human viewer, so that he or she can, for example, more easily and reliably grasp the image content.
- Objective image enhancement - for machine evaluation, (e.g. noise suppression or contour sharpening).

Example (subjective)

- **Subjective** image enhancement, e.g. gamma correction

$$f(x,y) = c \cdot b^{\gamma}(x,y) \quad c, \gamma \in \mathbf{R} > 0$$

original

LUT–transform γ= 0.5

Q.: How could we implement this in a line of MATLAB code?

Example (objective)

- **Objective** Image enhancement, e.g. nonlinear noise reduction

$$f(x,y) = MEDIAN(b(r,s)_{\forall(r,s)\in N(x,y)}) \quad \text{with} \quad N(x,y) = R_{xy}^{33}$$

original + salt & pepper – noise

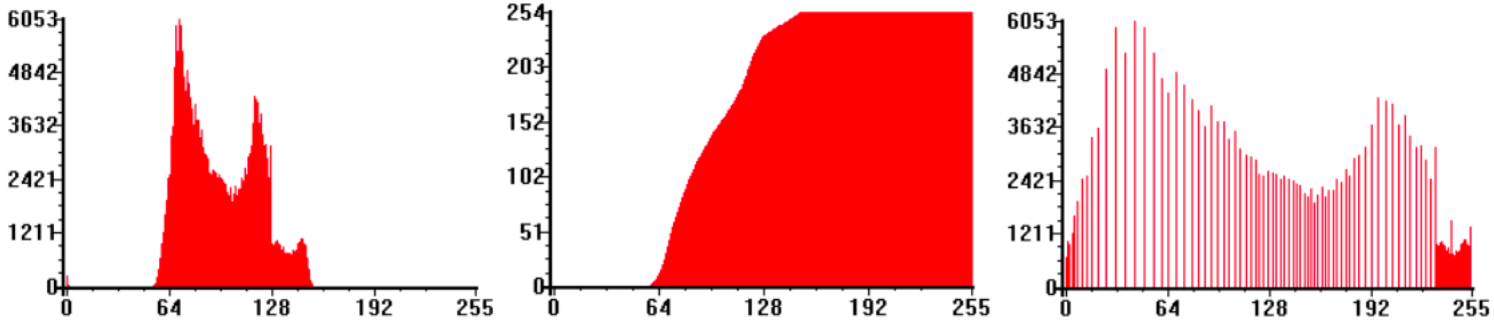
noisy image median filtered

- Figure : Noise reduction (left: image with salt & pepper interference, right: median filtered image).

Histogram Equalization

- Histogram Equalization: Idea - using the full intensity range
- Example: Subjective Effect on Example Image

- Algorithm: Transformation Function from Histogram



Histogram Equalization I

- Given: *probability density* for $\mathrm{f} : \mathrm{p_f}\,(\mathrm{f})$, transformation function T :
  - Let T be unique and monotonically increasing between 0 and 1.
  - Let be $\mathrm{f,g} \in \mathrm{R}$ with $0 \leq \mathrm{g,f} \leq 1$, then g is obtained to:

$$g = T(f) = \int_0^f p_f(\gamma) d\gamma \quad mit \ (0 \leq f \leq 1)$$

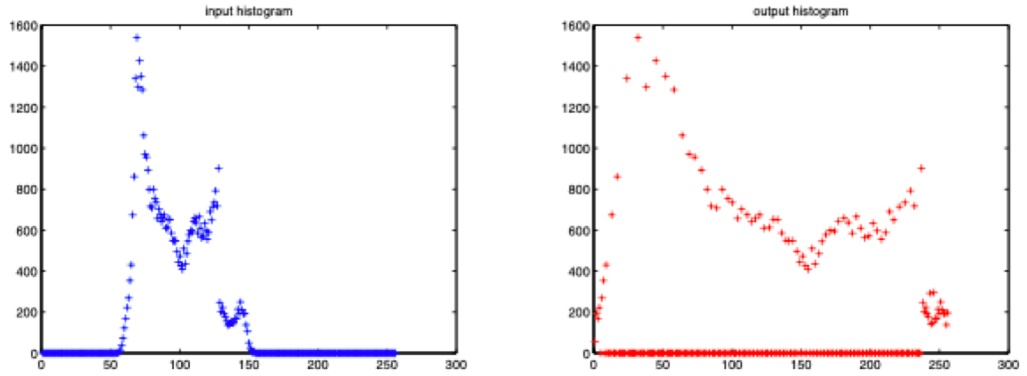
Digital practice I

- Instead of the distribution density, one considers the relative frequency of the gray values f_k :

$$p_f(f_k) = \frac{n_k}{n} \quad k = 0, 1, \ldots, L-1$$

n_k = Frequency of gray values f_k

n = $M * N$ (Number of pixels in image)



Discrete histogram equalization (histo-leveling)

- The discrete gray values $\mathrm{g} \in \mathrm{N}_{+},\ \mathrm{g}_{\min} \leq \mathrm{g} \geq \mathrm{g}_{\max}$ are obtained by means of a unary transformation $\mathrm{T}_{\mathrm{HE}} : \mathrm{f} \rightarrow \mathrm{g}$
- Result image $\mathrm{g}(\mathrm{x},\mathrm{y})$ using point operation over all image coordinates (x,y):

$$T_{HE} : g(x,y) = g(f(x,y)) = ROUND\left[(g_{max} - g_{min}) \cdot \sum_{\gamma=f_{min}}^{f} h_f(\gamma)\right] + f_{min}$$

- With a gray value range $0 \leq \mathrm{b},\mathrm{g} \leq 255$ simplifies to

$$T_{HE} : g(x,y) = g(f(x,y)) = ROUND\left[255 \cdot \sum_{\gamma=0}^{f(x,y)} h_f(\gamma)\right]$$

- Rounding needed to map real values back to integer indexes for the “table lookup” of the CDF!

End Lecture 2 (short) – including topic introduced in exercise