Liste der noch zu erledigenden Punkte

Kapitel 1

1.Overview

- „image society“ (webpages: 1995 text-based, 2005 image based, 2015 video based ...)

  - data transfer rates $\uparrow$, compression rates $\uparrow$
  critical shift: reading $\rightarrow$ watching

- „Photoshop“-ing (remove wrinkles, bumps, ...)

- Images in medicine („medical image proscessing“), x-ray, CT, MRI, ultrasound, ... („modalities“).
  different questions:

  1.) Layout!

  align bottom measurments $\overset{?}{\Rightarrow}$ image
  expl: tomography
  $\Rightarrow$ difficult mathematical problems

  2.) Image enhancements

  - denoising
    simple pixels/lines: „sandpaper“ interpolation so richtig?
    global noise: smoothing
  - grayscale
    histogramm balancing (spreading)
  - distortion
    makes straight lines (in real world) straight (in the images)
  - edge detection
    contour enhancement
  - segmentation
    detect and separate parts of the image
  - registration
    sequence of images of the same object $\Rightarrow$ Wort? , compare Skizze
    $\nearrow$ object following in a movie

<u>Our Focus:</u>

- mathematical models/methods/ideas
- (algorthms)
- ((implementation))

skipped: Very fast intro: Matlab and images

Kapitel 2

2.What is an image?

2.1 Discrete and continuous images

There are (at least) two different points of view:

Discrete / Digital

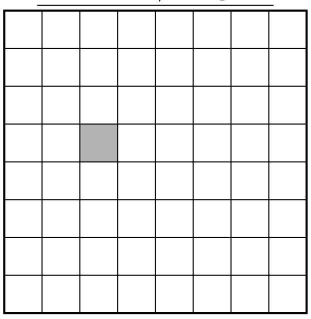


Abbildung 2.1: Discrete Image

Continous / Analogue

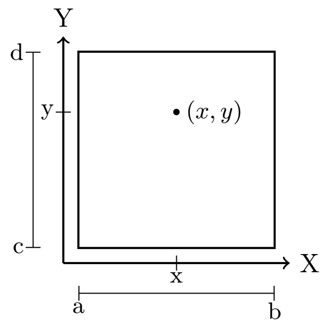


Abbildung 2.2: Continous Image

object:	matrix	function
tools:	linear algebra (SVD, ...)	analysis (differentrage, integrate, ...)
pros:	(finite storage) storage, complexity	freedom, tools, motions?P.4 (e.g. edge discontinuity)
cons:	limitations: zooming, rotations, ...	storage (infinite amout of data)

arguably, one has:

- real life $\Rightarrow$ continuous „images“ (objects)
- digital camers $\Rightarrow$ discrete images

In general we will say:

Definition 2.1 ((mathematical) image)**.** A (mathematical) *image* is a function

$$u : \Omega \to F,$$

where: $\Omega \subset \mathbb{Z}^d$ (discrete) or $\Omega \subset \mathbb{R}^d$ (continuous) ... *domain*
$d = 2$ (typical case 2D), $d = 3$ („3D image“ $=$ body or $\underbrace{\text{2D} + \text{time}}_{\text{movie}}$)

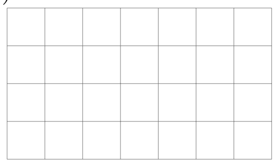
$d = 4$ (3D $+$ time)

F ... *range of colours*

$F = \mathbb{R}$ or $[0, \infty]$ or $[0, 1]$ or $\{0, \dots 255\}$, ... grayscale (light intensity)

$F \subset \mathbb{R}^3$ ... RGB image (colored)

$F = \{0, 1\}$ ... black/white



3 Layers
$\Rightarrow$ colored images:w

Matlab stuff

Large parts of the course: analytical approach (i.e. continuous domain Ω)
Since we want to differentirate, ... the image u.

Still: need to assume that also F ist continuous (not as $\{0, 1\}$, $\{0, 1, \dots, 255\}$ or $\mathbb{N}$) since otherwise the only differentiable (actually, the only continuous) functions $u : \Omega \to F$ are *constant* functions $\Leftrightarrow$ single-colour images

Also: We usually take F one-dimensional ($F \subset \mathbb{R}$). Think of it as either

- gray scaled image, or
- treating R,G & B layer separately

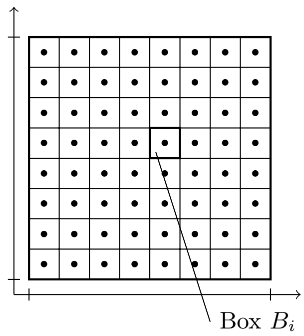
2.2 Switching between discrete and continuous images

continuous $\to$ discrete:

- divide the continuous image in small squared pieces (boxes) (superimpose grid)
- now: represent each box by *one* value
  - strategy 1: take function value $u(x_i)$ for x_i = midpoint of box B_i
  - strategy 2: use mean value

$$\frac{1}{|B_i|} \int_{B_i} u(x) dx$$

$\Omega =$



$\Rightarrow$ discrete image

strategy 1: simple (and quick) but problematic ($u(x_i)$ might represent $u|_{B_i}$ badly; for $u \in L^p$, single point evaluation not even defined)

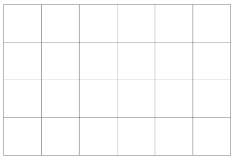
strategy 2: more complex but also more „democratic“ (actually closer to the way how CCD Sensors in digital cameras work)

often the image value of the box B_i gets also digitized, i.e. fitted (by scaling & rounding) into range $\{0, 1, dots, 255\}$

discrete $\to$ continous

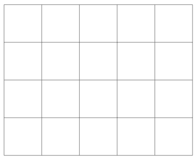
This is of course more tricky ...

- Again: each pixel of the discrete image corresponds to a „box“ of the continuous image (that is still to be constructed)
- Usually: pixel value $\mapsto$ function value at the *midpoint* of the box
- Question: How to get the other function values (in the box)?



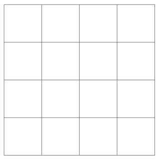
<u>idea 1:</u> just take the function value of the nearest midpoint („nearest neighbour interpolation")

$$\text{For each } x \in B_i : u(x) := u(x_j) \quad \text{where } |x - x_j| = \min_k |x - x_k|$$



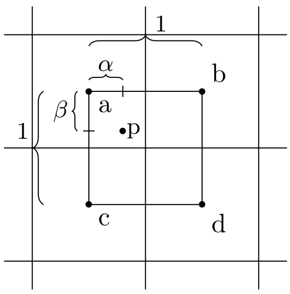
$\Rightarrow$ $u(x) = u(x_i)$ for all $x \in B_i$
$\Rightarrow$ each box is uni-color
$\Rightarrow$ the continuous image is essentially still discrete

<u>idea 2</u>: (bi-) linear interpolation



Let $a, b, c, d \ldots$ function values at 4 surrounding adjacent midpoints ($\nearrow$ figure)
$\alpha, \beta, 1-\alpha, 1-\beta \ldots$ distance to dotted lines ($\nearrow$ figure, w.l.o.g, bob is 1×1)

interpolation (linear) on the dotted line between a and b:



$e := a + \alpha(b - a) = (1-\alpha)a + \alpha b$
(1D - interpolation, convex combination)

Similarly: $f = (1-\alpha)c + \alpha d$

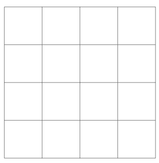
Then: The same 1D-interpolation between e and f

$$\begin{aligned} \Rightarrow u(x) :&= (1-\beta)\cdot e + \beta \cdot f \\ &= (1-\beta)[(1-\alpha)a + \alpha b] + \beta[(1-\alpha)c + \alpha d] \\ &= \underbrace{(1-\alpha)(1-\beta)}\, a + \underbrace{\alpha(1-\beta)}\, b + \underbrace{(1-\alpha)\beta}\, c + \underbrace{\alpha\beta}\, d \end{aligned}$$

$\in [0,1] \wedge \sum = 1$

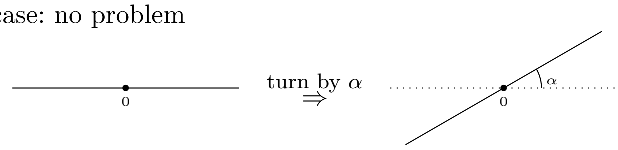
$\Rightarrow$ <u>convex combination</u> of the function values a, b, c, d at the the surrounding 4 midpoints (on which points is the nearest, instead of taking just a, b, c or d - depending)
$\Rightarrow$ 2D linear interpolation, *bi-linear interpolation* (can be interpreted as spline interpolation with bilinear basis splines).

Beispiel 2.2. Rotate image



by angle $\phi \neq k \cdot \frac{\pi}{2}$

- continuous image case: no problem



$$x = D_\varphi\, y \qquad x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix},\ y = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix},\ D_\varphi = \begin{pmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{pmatrix}$$

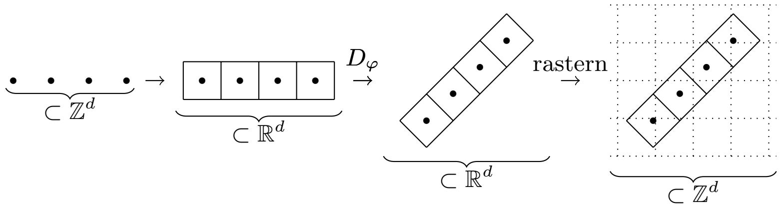
2D rotation matrix

$$y = D_\varphi^{-1}\, x = D_{-\varphi}\, x$$

$$\Rightarrow v(x) := u(y) = u(D_{-\varphi}\, x) \quad \forall x \in \text{domain of the rotated image}$$

- discrete image case: problem !

  For $x \in$ domain of notated image, in general $D_{-\varphi}\, x \notin$ domain of original image[1]
  Way out: $v(x) :=$ *interpolation* between the $u(\cdot)$ of the 4 surrounding pixels of $D_{-\varphi}$



Something to think about:
What happens in the limit (?) if we, starting with an image (discrete or continuous), repeatedly switch between discrete and continuous, non-stop . . . ?
Does the answer depend on the way of switching ? (continuous → discrete: midpoint or average, discrete → continuous: nearest neighbour or bilinear?)

[1]it's not an integer

Kapitel 3

3.Histogramm and first applicatsion

3.1 The histogramm

Definition 3.1 (histogram)**.** Let $\Omega \subset \mathbb{Z}^d$, $F \subset \mathbb{R}$ discrete and $u : \Omega \to F$ a discrete discrete image. The function

$$H_u : F \to \mathbb{N}_0 \; (:= \mathbb{N} \cup \{0\})$$

with

$$H_u(k) := \# \{x \in \Omega : u(x) = k\}, \quad k \in F$$

is called *histogramm* of the image u.

$H_u(k)$ counts how often colour k appears in u.

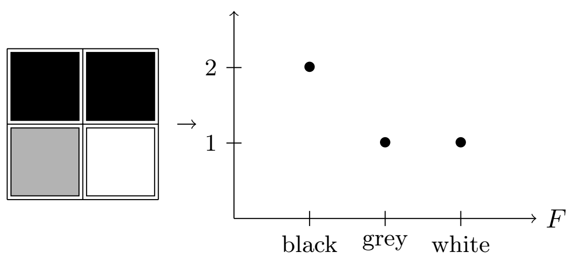
$$\sum_{k \in F} H_u(k) = |\Omega| = \text{number of pixels in the whole image}$$

or

$$\frac{H_u(k)}{|\Omega|} = \begin{array}{l}\text{relative frequence of colour } k \text{ in image } u \\ \text{(relative Häufigkeit)}\end{array}$$

Beispiel 3.2.

$u =$



If u ist a continous image, H_u can be understood as a measure (generalized function)[1]. Another way to write this:

$$H_u(k) = \sum_{x \in \Omega} \delta_{u(x)}(k), \; k \in F \qquad\qquad H_u(k) = \int_\Omega \delta_{u(x)}(k) dx, \; k \in F$$

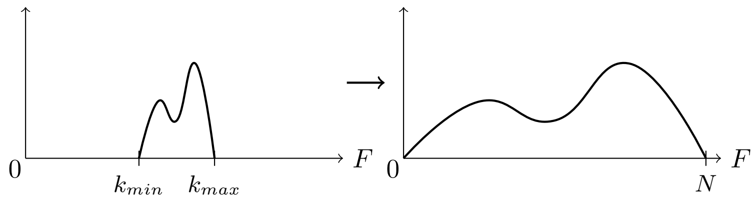
hier fehlt noch das Kronecker underarrow

Matlab-Code

[1] density of a probability distribution

3.2 Application: contrast enhancement

If the image only uses a small part of the available colour/grayscale „palette“ F, then its contrast can be improved by „spreading“ the histogramm over all of F.
Simple idea:



$$\begin{tikzcd} \Omega \arrow[r, bend left, "\text{originales Bild}"] \arrow[rr, bend right, "\text{Bild mit besserem Kontrast}"'] & F \arrow[r, bend left, "\text{Zauberei } T"] & F \end{tikzcd}$$

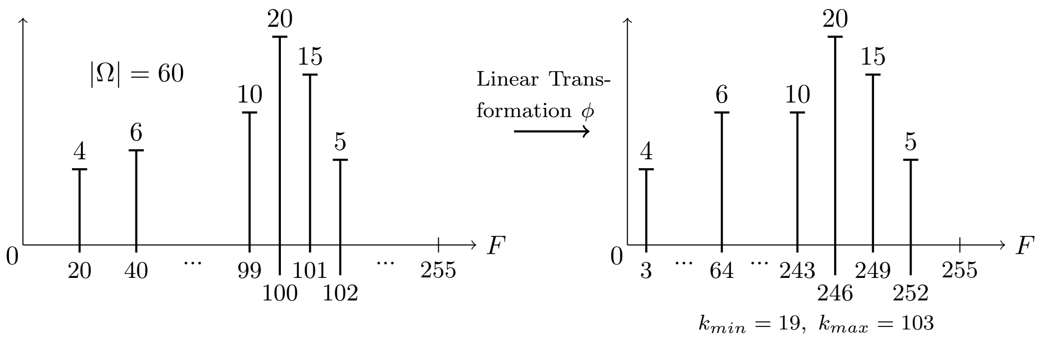
$v = T \circ u$
$v(k) = T(u(k))$

5

The above simple idea („contrast stretching“) corresponds to

$$\begin{aligned} \varphi : k_{\min} &\mapsto 0 \\ k_{\max} &\mapsto N \\ &\text{and linear in between} \\ \text{i.e} \qquad \varphi(k) &= \left[\frac{k - k_{\min}}{k_{\max} - k_{\min}}\right] \end{aligned}$$

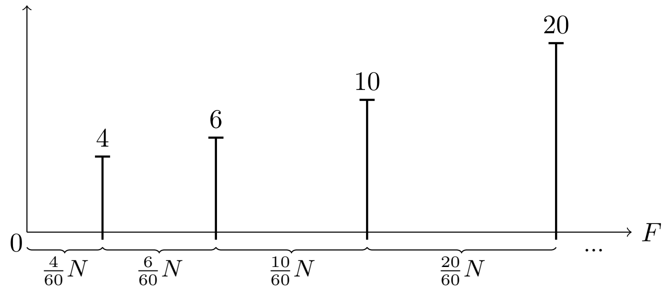
Where [$\cdot$] means ... rounding to the nearest integer (assumuning that $F = \{0, 1, \ldots, N\}$.
Example histogram:



A bit more sophisticated:

$$\begin{aligned} \varphi :\ (k_{\min} &\mapsto 0) \\ k_{\max} &\mapsto N \\ &\text{and } \textbf{non} \text{ linear in between} \end{aligned}$$

such that colour ranges that occur more frequently in u can occupy a larger range of colours in u. ($\Rightarrow$ visibility $\uparrow$)
Example histogramm spread out according to frequency of occurence:



$\Rightarrow$ „density“ is equalized over $F = \{0, \dots, N\}$

Ideal would be:

Layout S.12 u

Note: The new colours (i.e the location of the bars in the histogramm of u) only depend on the frequencies / *height* of the bars in H_u but not on the colours/location of the bars in H_u

Finally: The formula

$$\varphi(k) = \left[\frac{N}{|\Omega|} \sum_{l=0}^{k} H_u(l) \right]$$

This process is called „histogramm equalization“

Exercise ?!

3.3 Another application: conversion to b/w

Task: convert grayscale image to black white
- interesting for object detection/*segmentation* ...!

Idea: Find a threshold $t \in T$ s.t. the histogramm splits into two „characteristic“ parts

For $t \in F$ put

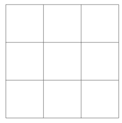
$$\text{black} := \{k \in F : k \leq t\}$$
$$\text{white} := \{k \in F : k > t\}$$

and

$$\tilde{u} := \begin{cases} 0, & u(x) \in \text{black} \\ 1, & u(x) \in \text{white} \end{cases} \qquad \tilde{F} = \{0, 1\}$$

How to find the threshold t:

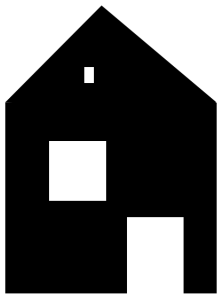
1.) Shape based methods
If the histogramm is „biomodal“
Put $t := \dfrac{k_{\max_1} + k_{\max_2}}{2}$
or $t := k_{\min}$



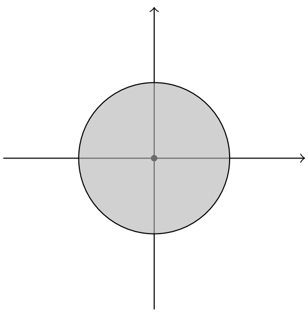
Kapitel 4

4.Basic Morphological Operations

B/W Bild:

$A =$ 

Structural element :

$B =$ 

4.1 Operations on A and B

$$A + B := \{a + b : a \in A, b \in B\}$$

This is called dilation.
You might imagine that at every dark point in the image A the Structurelement is applied.

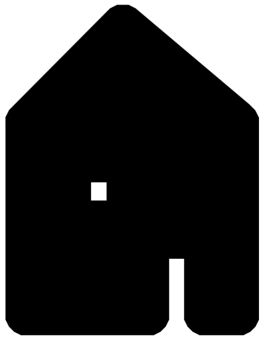
$A + B =$ 

Image created in Matlab through:

```
I=imread('Bild1.png');
se=strel('disk',40,8);
I2=imcomplement(imdilate(imcomplement(I),se));%I am using the complement of the image
    here so that the structural element is applied to the dark parts of the image
imshow(I2);
```

$$A - B := \{a : a + B \subset A\}$$

This is called erosion .
You can imagine that you search for the points in which the structural element fits.

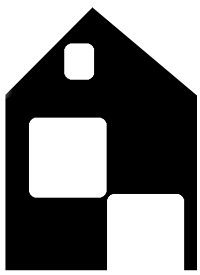
$A - B =$ 

Image created in Matlab thorugh:

```
I=imread('Bild1.png');
se=strel('disk',20,8);
I2=imcomplement(imerode(imcomplement(I),se));
imshow(I2);
```

One may quickly realize that $A \neq (A + B) - B$, so a new Operation is introduced:

$$A \bullet B := (A + B) - B$$

This is called <u>closing</u> and is used to e.g. remove noise. In the example image you might notice that the upper window is missing.

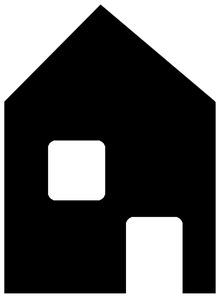
$A \bullet B =$ 

Image created in Matlab thorugh:

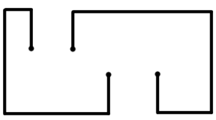
```
I=imread('Bild1.png');
se=strel('disk',20,8);
I2=imcomplement(imdilate(imcomplement(I),se));
I3=imcomplement(imerode(imcomplement(I2),se));
imshow(I3);
```

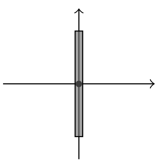
The inverse also exists:

$$A \circ B := (A - B) + B$$

This is called <u>opening</u> .

This time with a new example:

$A =$ 

$B =$ 

$$A \circ B =$$

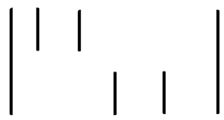


Image created in Matlab thorugh:

```
I=imread('Bild2.png');
se=strel('line',10,90);
I2=imcomplement(imerode(imcomplement(I),se));
I3=imcomplement(imerode(imcomplement(I2),se));
imshow(I3);
```

Kapitel 5

5.Entrauschen: Filter und Co

5.0.1 Noise

Noise = Unwanted disturbances in an image. Mostly becaue of

- point wise
- random
- independent

We consider *noise* to be an additive disturbances (for multiplicative noise use *log*).
Notation:

$$\begin{array}{ccccc} \boxed{f_0} & \xrightarrow{+\text{ noise}} & \boxed{f} & \xrightarrow{\text{denoise}} & \boxed{u} \\ \text{clean} & & \text{disturbed image} & & \text{result} \end{array}$$

The quality of the denoised image u compared to the original image f_0 is described by norms:

$$\begin{aligned} &||f - f_0|| \ldots \text{noise} \\ &||u - f_0|| \ldots \ \underline{\text{absolute error}} \\ &\frac{||u - f_o||}{||f - f_0||} \ldots \ \underline{\text{relative error}} \ \text{ compared to the noise} \\ &\frac{||u - f_o||}{||f_0||} \ldots \text{relative error compared to the signal} \end{aligned}$$

Typically the chosen norm is:

$$||f|| = ||f||_2 = \sqrt{\int_\Omega |f(x)|^2 \, dx}$$

or in the discrete:

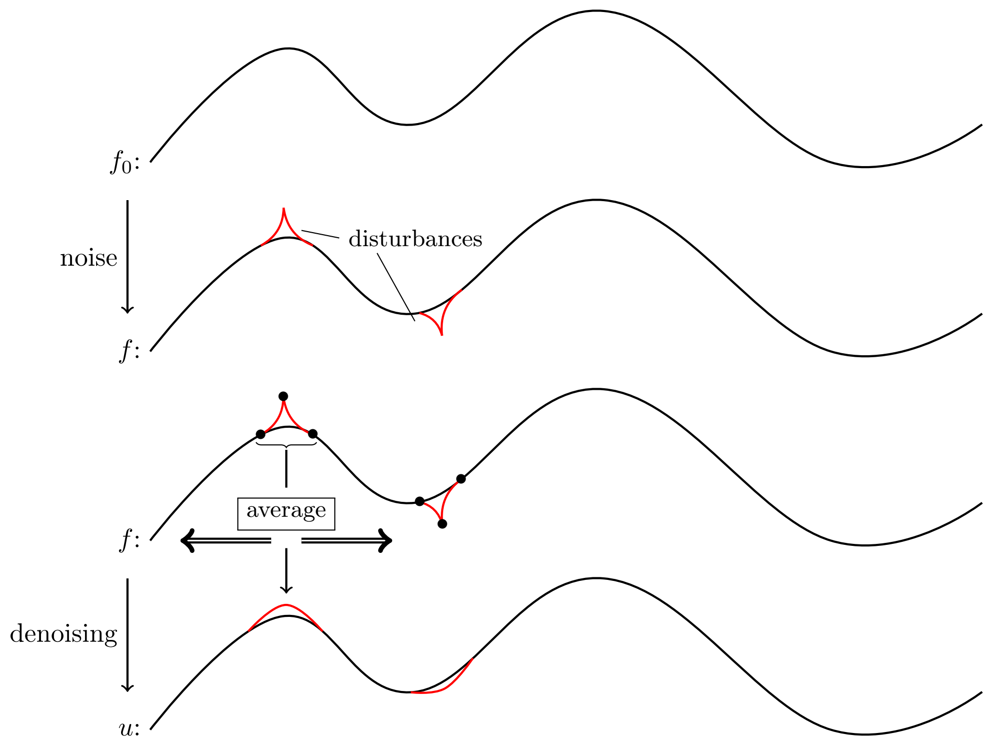
$$||f||_2 = \sqrt{\sum_{x \in \Omega} |f(x)|^2}$$

Closely connected is the Signal to noise ratio (SNR):

$$log(\underbrace{\frac{||f_0||_2}{||u - f_0||_2}}_{\in\ [1,\infty)}) \in [0, +\infty), \text{ where 0 is bad and } +\infty \text{ is good.}$$

5.0.2 smoothing filter

Idea: (to simplify in 1D)

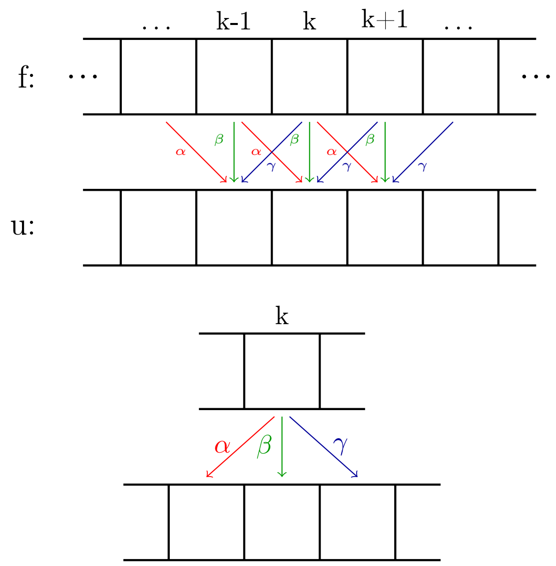


$$u(k) := \alpha \cdot f(k-1) + \beta \cdot f(k) + \gamma \cdot f(k+1) \tag{5.1}$$

where:

$$\alpha + \beta + \gamma = 1 \tag{5.2}$$

More precisely (5.1) means:



With (5.1) there is a mapping $f \mapsto u$, we write

$$u = m \boxast f, \text{ this is called } \underline{\text{Correlation}}.$$

where:

$$\boxed{(m \boxast f)(k) = \sum_{i \in supp(m)} m(i) f(k+i)} \tag{5.3}$$

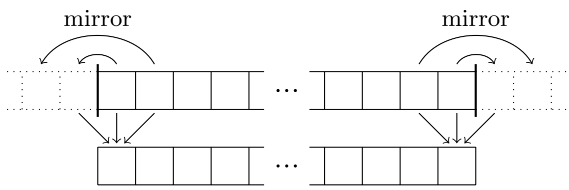
and:

$$m= \begin{array}{c|c|c|c|c} \cdots & -1 & 0 & 1 & \cdots \\ \hline \cdots & \alpha & \beta & \gamma & \cdots \end{array}$$ called <u>mask</u> .

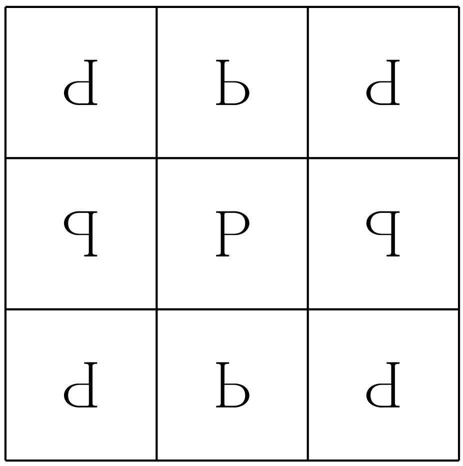
If you set $j := k + i$ in (5.1), then $i = j - k$, which means:

$$\boxed{(m \boxast f)(k) = \sum_{i \in supp(m)} m(j-k) f(j)} \tag{5.4}$$

To apply the mapping onto the boundary the image is reflected, in 1D:



in 2D:



Formula (5.4) might remind one of the <u>convolution</u> :

$$\boxed{(g * f)(k) = \sum_{j \in \mathbb{Z}} g(\underbrace{k-j}_{\text{Difference to (5.4)}}) \cdot f(j)} \tag{5.5}$$

If you set $g(i) := m(-i) =: \tilde{m}(i)$, which corresponds to a reflection of the Mask, then

$$m \boxast f = g * f = \tilde{m} * f$$

Properties of the convolution:

1. $(f * g) * h = f * (g * h)$, Associativity
2. $f * g = g * f$, Commutativity

3. $\tilde{f} * \tilde{g} = \widetilde{f * g}$, Compatibility with reflection

Properties of the correlation:

1. $f \boxast (g \boxast h) = \tilde{f} * (\tilde{g} * h) \overset{\boxed{1}}{=} (\tilde{f} * \tilde{g}) * h \overset{\boxed{3}}{=} (\widetilde{f * g}) * h = (f * g) \boxast h \neq (f \boxast g) \boxast h$, not associative!
2. $f \boxast g = \tilde{f} * g \overset{\boxed{2}}{=} g * \tilde{f} = \tilde{\tilde{g}} * \tilde{f} \overset{\boxed{3}}{=} (\widetilde{\tilde{g} * f}) = \widetilde{g \boxast f} \neq g \boxast f$, not commutative!
3. $\tilde{f} \boxast \tilde{g} = \tilde{\tilde{f}} * \tilde{g} \overset{\boxed{3}}{=} (\widetilde{\tilde{f} * g}) = \widetilde{f \boxast g}$, Compatibility with reflection